DIGITAL SIGNATURE SCHEME ENABLING PRE-CONTROL OF CONTENT EDITING FOR SECONDARY USE

Tatsuhiko Sano, Yoshio Kakizaki, Masaki Inamura, and Keiichi Iwamura.
Department of Electrical Engineering, Graduate School of Engineering, Tokyo University of Science.
1-14-6 Kudankita, Chiyoda-ku, Tokyo, 102-0073 Japan.
{sano, kakizaki, minamura, iwamura}@sec.ee.kagu.tus.ac.jp

ABSTRACT

In this study, we propose a digital signature scheme that enables the precontrol of content editing for secondary use. In the proposed scheme, the original content is divided into several parts, a unique signature is defined for each part, and aggregate signatures are created. The unique signatures are exhibited as individual signatures according to the first author's scope of permission, and a secondary author can edit the content within these permission boundaries. For data insertion control, empty data is placed between the parts and controlled by a pairing operation using the aggregate signatures, whereby the verifier determines whether the editing is allowable. A sanitizable signature scheme is limited to deletions (sanitization). In the proposed scheme, not only deletions but also alterations to existing data and insertions of new data are possible.

KEYWORDS

Digital signature, Sanitizable signature, Copyright, Secondary use

1. INTRODUCTION

Until now, general content distribution services have targeted commercial content data, e.g., digital versatile discs for video or video-on-demand service. Content makers and distributors providing these services need to protect their investments. Therefore, the following functions are needed for content protection:

1. Access Control: Allowing legal users users to access content while excluding illegal users.
2. Copy Protection: Restricting users from making copies of data without express permission of the copyright holders.

For these reasons, many types of traditional digital rights management (DRM) technologies have been utilized.

However, in recent years, the Internet connection environment has become more widespread and new content distribution services are prevalent. User-generated content (UGC) distribution services, which are network services, are widespread, e.g., YouTube and Facebook. The content distribution in UGC services has a very different form from traditional content distribution; users expect other users to copy and modify the content. However, traditional DRM technologies for access control and copyright protection cannot be intrinsic to this process because of the denial of copying and editing. Therefore, a new scheme is needed for controlling versions that also enables inheritance.

To explore the potential of this new scheme, we investigate a system that employs a digital signature scheme. In digital signature schemes, generally, if users modify data or its digital signature, the signature cannot be verified, making it difficult for users to edit content data after signing with these schemes. Therefore, sanitizable signature schemes, which can verify the authenticity of content data without interacting with the signer, have been proposed. [Remark 1] To consider existent sanitizable signature schemes suitable for adaption to a content editing system, we must resolve the following problem; existent sanitizable signature schemes must be able to verify the signature only by sanitizing, in other words, by deleting content data. However, these schemes can verify the signature of data modified multiple times. In UGC services, users reuse and modify content data frequently and redistribute it. Therefore, we consider that the new scheme could be adapted to UGC services.

To incorporate the above techniques, consider a system that applies a digital signature. In general, because the verification cannot be performed when the content or electronic signature is changed after creating the signature, it is difficult to edit the content after signing. Sanitizable signature schemes have been proposed that secure the legitimacy of the content after creating the signature if concealment and deletion are performed, and these schemes are applicable to editing. However, because conventional sanitizing signature schemes are intended to be used for confidential information, such as in government documents, their purpose has only been to remove and change (sanitizing) some of the data.

Therefore, facilities to precontrol changes, deletions, or additions while still guaranteeing the original content are largely ignored.

In this study, we propose a digital signature scheme that can control the partial editing of content for secondary use for the solution of the above problem. In the case of editing content data on UGC services, users want to control edits such as alteration, deletion, and insertion. Furthermore, users want to preset permissions or denials of alteration, deletion, and insertion before content distribution. Users using the proposed system can generate a sanitizable signature corresponding to three types of editing control with presetting and can enable verification of a signature after modifying content data. Therefore, we can realize this precontrol system. In addition, the system allows the user to verify not only who makes or edits content data but also the process used to create this data with an order-specified multi-signature scheme. As a result, users who edit content data can reuse/remake this data within the permissions granted by copyright holders, and verifiers can certify all members involved in the making and editing of content.

2. RELATED WORKS

2.1 Aggregate Signature

Okamoto and Pointcheval defined a Gap-Diffie-Hellman (GDH) class[14]. Consider a $\mathbb{G}'$ with prime order p. They defined two problems for Diffie-Hellman (DH) as follows:

The computational DH (CDH) problem:

For $a, b \in \mathbb{Z}_p^*$ and $g \in \mathbb{G}'$, given (g, g^a, g^b), compute g^{ab}.

The decisional DH (DDH) problem:

For $a, b, c \in \mathbb{Z}_p^*$ and $g \in \mathbb{G}'$, given (g, g^a, g^b, g^c), determine whether $c = ab$.

For the GDH class, it is defined that the DDH problem is easy whereas the CDH problem is difficult.

The first example of such groups was given by Joux and Nguyen [15,16], and Boneh, Lynn, and Shacham showed that a new signature scheme based on the GDH class using pairing, which is one of the functions on specified elliptic curves, could be structured[4]. Consider two groups $\mathbb{G}_1$ and $\mathbb{G}_2$on elliptic curves enabling pairing, and $\mathbb{G}_\tau$ as a cyclic groups. Let e denote the symbol for pairing function $(\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_\tau)$. Pairing has the following characteristics:

- For $P_1, P_2 \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$, then $e(P_1P_2, Q) = e(P_1, Q)e(P_2, Q)$.
- For $P \in \mathbb{G}_1$ and $Q_1, Q_2 \in \mathbb{G}_2$, then $e(P, Q_1Q_2) = e(P, Q_1)e(P, Q_2)$.
- For $a, b \in \mathbb{Z}_p^*$, $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$, then $e(P^a, Q^b) = e(P^b, Q^a) = e(P^{ab}, Q) = e(P, Q^{ab}) = e(P, Q)^{ab}$.

As a result, a signature scheme based on a group on which pairing can be executed can be structured as follows:

Key Generation: The term g is a generator of a group $\mathbb{G}_2$. The secret key of the signer is a random element $x \in \mathbb{Z}_p^*$, whereas his/her public key is $v = xg$.

Signing: H is a one-way hash function that outputs a random element in the whole group $\mathbb{G}_1$, e.g., *MapToGroup*[4]. The term m is both a plain message and a signing target. The signer computes $h = H(m)$ and returns $\sigma = xh$.

Verification: When the verifier is given g, v, m and σ, he/she computes $h = H(m)$ and verifies $e(\sigma, g) = e(h, v)$.

While verifying, if a signer accurately generates a signature accurately, a verifier can verify it using the pairing result as follows:

$e(\sigma, g) = e(h^x, g) = e(h, g)^x$.

$e(h, v) = e(h, g^x) = e(h, g)^x$.

Therefore, if $e(\sigma, g) = e(h, v)$, then σ has been generated accurately. Furthermore, it has

been shown that the above scheme is secure against chosen-message attacks [4]. In this study, we call this signature scheme *the BLS signature scheme.*

After proposing the concrete GDH signature scheme, Boneh et al. also proposed also a prototype of the aggregate signature scheme on the basis of the BLS signature scheme [17]. Consider a group $\mathbb{U} = \{u_1, \cdots, u_n\}$, which is a group of n signers, and $\mathbb{L} = \{u_{i_1}, \cdots, u_{i_l}\} \subseteq \mathbb{U}$, which is a group of l members who actually participate in an aggregate signature. Let $\mathbb{J} = \{i_1, \cdots, i_l\}$ denote the set of indices of such members. Then, the proposers showed that one could structure the aggregate signature scheme on the basis of *the BLS signature scheme* as follows:

Key Generation: The term g is a generator of a group $\mathbb{G}_2$. The secret key of the signer $u_i \in \mathbb{U}$ is a random element $x_i \in \mathbb{Z}_p^*$, whereas his/her public key is $v_i = x_i g$.

Signing: H is a one-way hash function that outputs a random element in the whole group $\mathbb{G}_1$. The term m_{i_α} is both a plain message and a signing target of the signer $u_{i_\alpha} \in \mathbb{L}$. The signer computes $h_{i_\alpha} = H(m_{i_\alpha})$ and returns $\sigma_{i_\alpha} = x_{i_\alpha} h_{i_\alpha}$.

Aggregation: The issuer of an aggregate signature finally collects all σ_{i_α} generated by u_{i_α}, computes $\sigma = \Sigma_{j \in \mathbb{J}} \sigma_j$ and returns $(m_{i_1}, \cdots, m_{i_l}, \mathbb{L}, \sigma)$.

Verification: When the verifier is provided values of $g, m_{i_1}, \cdots, m_{i_l}, \mathbb{L}$ and σ, he/she collects all v_{i_α} using $\mathbb{L}$, computes all $h_{i_\alpha} = H(m_{i_\alpha})$, and verifies $e(\sigma, g) = \Pi_{j \in \mathbb{J}} e(h_j, v_j)$.

While verifying, if all signers accurately generate an aggregate signature, a verifier can verify this signature using the pairing result as follows:

$e(\sigma, g) = e\left(\Pi_{j \in \mathbb{J}} h_j^{x_j}, g\right) = \Pi_{j \in \mathbb{J}} e\left(h_j^{x_j}, g\right) = \Pi_{j \in \mathbb{J}}((h_j, g)^{x_j})$.

$\Pi_{j \in \mathbb{J}} e\left(h, v_j\right) = \Pi_{j \in \mathbb{J}} e\left(h, g_j^{x_j}\right) = \Pi_{j \in \mathbb{J}}((h_j, g)^{x_j})$.

Therefore, if $e(\sigma, g) = \Pi_{j \in \mathbb{J}} e(h_j, v_j)$, then all σ_{i_α} have been generated accurately. Furthermore, it has been proven that the above scheme is secure even with the reduced security of *the BLS signature scheme* [17].

2.2 Sanitizable Signature

A sanitizable signature scheme is a signature which allows a semi-trusted party called a sanitizer to hide parts of the original message after the message is signed, without interacting with the signer. A verifier can confirm the integrity of disclosed parts of the sanitized document from the signature. Sanitizable signatures are quite useful in government or military offices, where there is a dilemma between the disclosure requirements of documents and private secrets. In this section, we explain IIKO, which is a sanitizable signature scheme; in the following section, we compare it with our scheme [5].

In the IIKO scheme, a signer establishes the following control statuses for each partial document:

- "Accept Sanitizations" and "Accept Deletions" (ASAD).
- "Accept Sanitizations" and "Prohibit Deletions" (ASPD).
- "Prohibit Sanitizations" and "Accept Deletions" (PSAD).
- "Prohibit Sanitizations" and "Prohibit Deletions" (PSPD).
- "Sanitized" and "Accept Deletions" (SAD).
- "Sanitized" and "Prohibit Deletions" (SPD).
- "Deleted" (D).

We show each status in Figure 1.

The term M_α^* is the partial document in the whole document, and the above status is established to M_α^*. The term h_α is the hash value from M_α^*. Furthermore, if the status is SAD or SPD, M_α^* is not expressed because this partial document has already been sanitized, and only the hash value, i.e., h_β^*, from this partial document is expressed. The term σ_i is the signature of M_α^* that can be sanitized or has already been sanitized, and the term σ is an aggregate signature of all σ_i. If sanitization is prohibited for a partial document, the signature of this document is not generated and not aggregated to σ. Therefore, when a partial

document prohibiting sanitization is sanitized, the verifier can detect illegal sanitization.

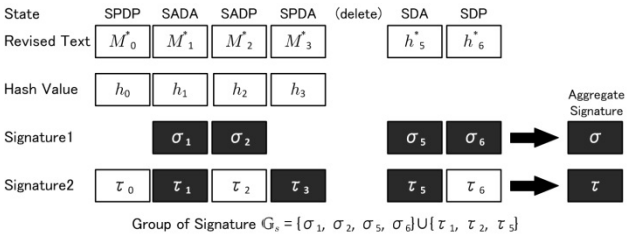

Figure 1. Statuses in IIKO Scheme.

On the other hand, the term τ_j is the signature of M_j^* targeting to all partial documents (however, $\sigma_j \neq \tau_j$), and the term τ is an aggregate signature of all τ_j. The term τ_{j_0} is the signature of a partial document that accepts deletions and is open to the public. The term τ_{j_1} is the signature of a partial document that prohibits deletions and is not open to the public. If a partial document that accepts deletions is deleted, τ_j can be divided into τ and a new aggregate signature can be generated. If a partial document that prohibits deletions is deleted, τ_j cannot be divided into τ and the verifier can detect an illegal deletion.

3. SCHEME DESCRIPTION

This method assumes secondary content, such as a situation in which you add new content or edit existing content. We need to allow secondary content when permitted by the author but restrict it when denied by the author. In such cases, copy protection and limits on the number of plays are insufficient measures. It is necessary to establish a mechanism that can authorize the editing of content.

For such a request, the PIAT signature scheme [5] and sanitizable signature scheme [6] are proposed for a mechanism that it can pre-control edits. However, these schemes do not permit the addition of new data or the change of existing data. Our method allows control of changes, deletions, and additions to partial contents by setting aggregate and individual signatures.

3.1 Control Status of Each Part

One of seven control statuses is defined for each part. Two types of parts are assumed: parts with existing data and parts with empty data. Existing data is data that can actually be played back and such data can be altered and deleted. Empty data cannot be played back. Instead, it is used for controlling the insertion of data.

(a) Accept Alterations and Accept Deletions (AAAD)
(b) Accept Alterations and Prohibit Deletions (AAPD)
(c) Prohibit Alterations and Accept Deletions (PAAD)
(d) Prohibit Alterations and Prohibit Deletions (PAPD)
(e) Deleted (D)
(f) Accept Insertion (AI)
(g) Prohibit Insertion (PI)

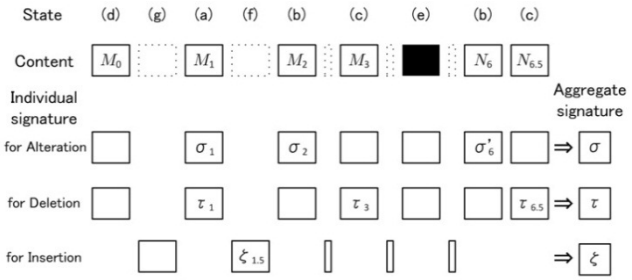

Figure 2: Example control statuses

3.2 Entity

For the proposed digital signature system in general content, three entities are defined as follows:

Author - Creating content and setting the edit control license.

Editor - Content editing with edit rights.

Verifier - Verifying whether the contents have a valid signature.

3.3 Author

The author function divides the original content into several parts and sets the status to one of (a)–(e) for parts with existing data and to (f) or (g) for parts with empty data, as shown in Figure 2. The author generates a content identifier (ID) and parts identifiers (ID_i) and then generates individual signatures for alteration, deletion, and

insertion from the ID_i and hash values for each part. Aggregate signatures for alteration, deletion, and insertion are each generated by convoluting the individual signatures for alteration, deletion, and insertion. Next, the author outputs only the signatures of the parts for which alteration, deletion, and insertion are allowed in each signature group.

As a result of these processes, the author can control the originality of the data in each part without including individual signatures in the signature groups and can allow editing of the data in each part by using the existing individual signatures in the signature groups.

3.4 Editor

The editor function receives the aggregate signatures and signature groups when receiving the original content. To alter or delete data, the editor separates the individual signatures for the existing data from the aggregate signatures for alteration or deletion. To alter the data, the editor generates a signature for new data and convolutes the signature with the aggregate signatures. To insert new data between existing data, the editor separates the individual signatures for the parts with empty data from the aggregate signatures for insertion. If there are no individual signatures for the parts containing data to be edited, the aggregate signatures cannot be renewed.

3.5 Verifier

The verifier function determines whether the editor is allowed to edit the content. With the pairing operation using the aggregate signatures, the hash values calculated for the content, and a public key, the verifier determines whether the editing is allowable. Playback devices and applications are assumed to have a verification capability.

First, the verifier determines whether the insertions are allowable. It then determines whether IDi is ascending. If the first verification is successful, the verifier determines whether the alterations and deletions are allowable. If all verifications are successful, the verifier confirms the integrity of the content.

4. PROTOCOL

First, we explain the symbols come out below.

g_2 : generator of G_2
p : prime number
$sk0$: author's secret key
$pk0$: author's public key
$sk1$: editor's secret key
$pk1$: editor's public key
M :original content's message
M_i :i-th parts of original content's message
ID : content identifier
ID_i: content identifier of i-th parts of original content's message
M^*: message concatenated ID
B : empty data
d : dummy data
h : hash value
σ_i : individual signature for change
τ_i : individual signature for deletion
$\xi_{i.5}$: individual signature for addition
σ : aggregate signature for change
τ : aggregate signature for deletion
ξ : aggregate signature for addition
$\mathbb{G}_c$: signature group of individual signatures for change
$\mathbb{G}_d$: signature group of individual signatures for deletion
$\mathbb{G}_a$: signature group of individual signatures for addition
N : data changed by editor
$e(,)$: pairing function
ε_i : individual signature for difference value of hash
ε : aggregate signature for difference value of hash
$z_{i.5}$: individual signature for addition (verification)
z : aggregate signature for addition (verification)
w_i : individual signature for difference value of hash (verification)
w : aggregate signature for difference value of hash (verification)
x_i , X : data for change (verification)
y_i , Y : data for deletion (verification)

Generating a Secret/ Public Key Pair

Author: A secret and public key pair $(sk0, pk0) \in Z/pZ \times G_2$

$$pk0 \leftarrow g_2^{sk0}$$

Editor: A secret and public key pair $(sk1, pk1) \in Z/pZ \times G_2$

$$pk1 \leftarrow g_2^{sk1}$$

Setting Individual Signatures and Aggregate Signatures

An author performs the following process.

A) Divide original content M into n parts.

$$M \rightarrow M_1, M_2, \ldots, M_n$$

B) Generate a content identifier ID and partial content identifiers ID_i, and process as follows:

$$M_0^* \leftarrow ID, M_i^* \leftarrow ID \,||\, ID_i \,||\, M_i$$

C) Assume that the empty data $B_{i.5}$ exists between divided data, use dummy data d, and process as follows:

$$B_{i.5}^* \leftarrow ID \,||\, ID_{i.5} \,||\, d$$

D) Determine one of the statuses (a)~(e) for each partial content, one of the statuses (f)~(g) for each empty data.

E) Generate hash values from $M_0^*, M_i^*, B_{i.5}^*$ $(i = 0, \ldots, n)$.

$$h_i \leftarrow H(M_i^*), \quad h_{i.5} \leftarrow H(B_{i.5}^*)$$

Set individual signatures for change, deletion, and addition.

$$\sigma_i \leftarrow H(ID \,||\, ID_i \,||\, h_i \,||\, 0^c)^{sk0}$$
$$\tau_i \leftarrow H(ID \,||\, ID_i \,||\, h_i \,||\, 1^c)^{sk0}$$
$$\xi_{i.5} \leftarrow H(ID \,||\, ID_{i.5} \,||\, h_{i.5} \,||\, 0^c)^{sk0}$$

F) Set aggregate signatures for change, deletion, and addition.

$$\sigma \leftarrow \Pi_i^n \sigma_i, \quad \tau \leftarrow \Pi_i^n \tau_i, \quad \xi \leftarrow \Pi_i^n \xi_{i.5}$$

G) Output individual signatures of partial contents permitted to change, delete, and add into each signature aggregation $\mathbb{G}_c, \mathbb{G}_d, \mathbb{G}_a$

H) Output the content, signature group $\mathbb{G}_c, \mathbb{G}_d, \mathbb{G}_a$, aggregate signatures, σ, τ, ξ.

$M^* = \{M_0^*, M_1^*, \ldots, M_n^*\}$

Update the Content and Signatures

For simplicity, we define the following processes as (i),… ,(v).

(i) $M_i^* \leftarrow N_i^*$, $h_i' \leftarrow H(N_i^*)$, σ/σ_i , $\mathbb{G}_c \setminus \{\sigma_i\}$, $\sigma_i' \leftarrow H(ID \,||\, ID_i \,||\, h_i' \,||\, 0^c)^{sk1}$, $\sigma \leftarrow \sigma \times \sigma_i'$

(ii) τ/τ_i , $\mathbb{G}_d \setminus \{\tau_i\}$, $\tau_i' \leftarrow H(ID \,||\, ID_i \,||\, h_i' \,||\, 1^c)^{sk1}$, $\tau \leftarrow \tau \times \tau_i'$

(iii) $\xi_{i.5} \leftarrow H(ID \,||\, ID_{i.5} \,||\, h_{i.5} - h_{i.5}' \,||\, 0^c)^{sk1}$

(iv) $B_{i.5}^* \leftarrow N_{i.5}^*$, $h_{i.5}' \leftarrow H(N_{i.5}^*)$,$\xi/\xi_{i.5}$,$\mathbb{G}_a \setminus \{\xi_{i.5}\}$, $\sigma_{i.5}' \leftarrow H(ID \,||\, ID_{i.5} \,||\, h_{i.5}' \,||\, 0^c)^{sk1}$,$\sigma \leftarrow \sigma \times \sigma_{i.5}'$

(v) $\tau_{i.5}' \leftarrow H(ID \,||\, ID_{i.5} \,||\, h_{i.j}' \,||\, 1^c)^{sku}$, $\tau \leftarrow \tau \times \tau_{i.5}'$

A) Choose the partial content M_i^* or the empty data $B_{i.5}^*$ that you want to edit from the content M^*.

B) The process depends on the type of edit as follows:

・(a)→(a) :

(i) , $\mathbb{G}_c \leftarrow \mathbb{G}_c \cup \sigma_i'$, (ii) ,$\mathbb{G}_d \leftarrow \mathbb{G}_d \cup \tau_i'$

・(a)→(b) :

(i) , $\mathbb{G}_c \leftarrow \mathbb{G}_c \cup \sigma_i'$, (ii)

If you change only the status , $\mathbb{G}_d \setminus \{\tau_i\}$.

・(a)→(c) :

(i) , (ii) , $\mathbb{G}_d \leftarrow \mathbb{G}_d \cup \tau'_i$

If you change only the status , $\mathbb{G}_c \setminus \{\sigma_i\}$.

・(a)→(d) :

(i) , (ii)

If you change only the status $\mathbb{G}_d \setminus \{\tau_i\}$,$\mathbb{G}_c \setminus \{\sigma_i\}$.

・(a)→(e) :

$M^* \setminus M_i^*$, σ/σ_i , $\mathbb{G}_c \setminus \{\sigma_i\}$, τ/τ_i , $\mathbb{G}_d \setminus \{\tau_i\}$

・(b)→(b) :

(i), $\mathbb{G}_c \leftarrow \mathbb{G}_c \cup \sigma_i'$, (iii), $\mathbb{G}_{bc} \leftarrow \mathbb{G}_{bc} \cup (h_i - h_i')$

・(b)→(d) :

(i), (iii) , $\mathbb{G}_{bd} \leftarrow \mathbb{G}_{bd} \cup (h_i - h_i')$

If you change only the status $\mathbb{G}_c \setminus \{\sigma_i\}$.

・(c)→(d) :

$\mathbb{G}_d \setminus \{\tau_i\}$

・(c)→(e) : $(h_i' = 0)$.

$M^* \setminus M_i^*$, (iii) , $\mathbb{G}_{bc} \leftarrow \mathbb{G}_{bc} \cup (h_i)$, τ/τ_i , $\mathbb{G}_d \setminus \{\tau_i\}$

・(f)→(a) :

(iv), $\mathbb{G}_c \leftarrow \mathbb{G}_c \cup \sigma_{i.5}'$, (v) ,$\mathbb{G}_d \leftarrow \mathbb{G}_d \cup \tau_{i.5}'$

・(f)→(b) :

(iv), $\mathbb{G}_c \leftarrow \mathbb{G}_c \cup \sigma_{i.5}'$, (v)

・(f)→(c) :

(iv) ,(v), $\mathbb{G}_d \leftarrow \mathbb{G}_d \cup \tau_{i.5}'$

・(f)→(d) :

(iv), (v)

・(f)→(g) :

$\mathbb{G}_a \setminus \{\xi_{i.j}\}$

C) Aggregate the calculated value ε_i, generate historical data of edit (r) and the signature of historical data σ_r.

$\varepsilon = \Pi \varepsilon_i, \quad r, \quad \sigma_r$

Table 1. Comparisons between the former scheme and our scheme

	Change	Deletion	Addition	Number of pre-signed	Number of verification
IIKO [5]		✓		2n	2
Proposed scheme	✓	✓	✓	3n+1	4

D) Output edited content, signature groups $\mathbb{G}_c$, $\mathbb{G}_d$, $\mathbb{G}_a$, aggregate signatures $\sigma, \tau, \xi, \varepsilon$, groups of the difference value of hash $\mathbb{G}_{\mathrm{bd}}$, $\mathbb{G}_{\mathrm{bc}}$、historical data of edit r, the signature of historical data σ_r .

Verification

A) Verify whether the content identifier ID of each partial data is equal to M_0^*.

B) Process as follows and verify these data.

$$z_{i.5} \leftarrow H(ID \,||\, ID_{i.5} \,||\, H(B_{i.5}^*) \,||\, 0^c)$$
$$z \leftarrow \Pi z_{i.j}, \;\; e(\xi, g_2) = e(z, pk0)$$

C) If A&B verification is correct, verify whether the partial content identifier ID_i in all data M_i^* is ascending.

D) Check existing of data with the difference value of hash in $\mathbb{G}_{\mathrm{bd}}$, and absence of data with the difference value of hash in $\mathbb{G}_{\mathrm{bc}}$.

E) If D verification is correct, using the difference value of hash in $\mathbb{G}_{\mathrm{bd}}$, $\mathbb{G}_{\mathrm{bc}}$, process as follows and verify signature ε.

$$w_i \leftarrow H(ID \,||\, ID_i \,||\, (h_i - h_i') \,||\, 1^c)$$
$$w \leftarrow \Pi w_i, \;\; e(\xi, g_2) = e(w, pk1)$$

F) If E verification is correct, generate the hash value of existing content $h_i' = H(M_i^*)$ and calculate the hash value of the original content.

$$h_i' + (h_i - h_i') = h_i$$

G) If there are the difference values of hash in $\mathbb{G}_{\mathrm{bd}}$ or $\mathbb{G}_{\mathrm{bc}}$, use calculated hash values in F verification. Otherwise, using the hash value of the existing data, you process as follows:

data to be changed : $x_i \leftarrow H(ID \,||\, ID_i \,||\, h_i \,||\, 0^c)$

data for deletion : $y_i \leftarrow H(ID \,||\, ID_i \,||\, h_i \,||\, 1^c)$

H) Using historical data of edit r, process as follows and verify these data.

for author : $X_0 \leftarrow \Pi x_i, \;\; Y_0 \leftarrow \Pi y_i,$

for editor : $X_1 \leftarrow \Pi x_i, \;\; Y_1 \leftarrow \Pi y_i,$

$$e(\sigma, g_2) = \Pi_{i=1}^{1} e(X_i, pki)$$
$$e(\tau, g_2) = \Pi_{i=0}^{1} e(Y_i, pki)$$

5. CONSIDERATION

5.1 Number of Signatures

Compared to the sanitizable signature scheme, the results of our precontrol scheme for change, deletion, addition, the number of presigned, and the number of verifications are shown in Table 1. The proposed scheme can achieve precontrol of all three statuses. One limitation of the proposed scheme is the necessity to generate the aggregate and individual signatures for all three statuses. Hence, the time needed for signature generation is increased in proportion to the number of divisions set by the content author. However, in practical use, the time taken is too short to be problematic.

5.2 Limitation of Contorol Status

It is possible to remain exposed to fraud unless care is taken in permission setting. For example, when you set the permissions to add empty data next to a part of the content with prohibit alteration and accept deletion (PAAD), if you remove the content part and add to the data in the region of the additional authorization, you change the data that prohibited change as a result. To eliminate this circumstance, it is necessary to limit the combinations of control statuses.

6. CONCLUSION

We have developed a digital signature scheme for controlling the editing of content for secondary use. This scheme can be used for moving and still images, music, and other media. In addition, this method can be applied equally well to the concept of UGC, even in apps and services. Future work includes redefinition and inheritance structure of the control status for multiple edits. Moreover, in case of dividing a lot of parts, it is necessary to optimize the process for generating signatures effectively. We intend to implement and evaluate the usefulness of this method.

REFERENCES

[1] YouTube, http://www.youtube.com/

[2] Facebook, http://www.facebook.com/

[3] Boneh, D., Gentry, C., Lynn, B. and Shacham, H.: Aggregate and Verifiably Encrypted Signatures from Bilinear Maps: EUROCRYPT 2003, LNCS2656, pp.416–432 (2003).

[4] Boneh, D., Lynn, B. and Shacham, H.: Short signatures from the Weil pairing: ASIACRYPT 2001, LNCS 2248, pp.514–532, (2001).

[5] Izumi, M., Izu, T., Kunihiro, N. and Ohta, K.: An Extension of Sanitizable and Deletable Signature: 2007-CSEC-38, pp. 355–361 (2007).

[6] Izu, T., Kunihiro, N., Ohta, K., Takenaka, M. and Yoshioka, T.: A Sanitizable Signature Scheme with Aggregation: ISPEC 2007, LNCS 4464, pp.51–64 (2007).

[7] PBC Library, http://crypto.stanford.edu/pbc/

[8] Miyazaki, K., Iwamura, M., Matsumoto, T., Sasaki, R., Yoshiura, H. and Imai, H.: Digitally signed document sanitizing scheme with disclosure condition control: IEICE Trans. Fundamentals E88-A, pp.239–247 (2005).

[9] Miyazaki, K., Susaki, S., Iwamura, M., Matsumoto, T., Sasaki, R. and Yoshiura, H.: Digital documents sanitizing problem: Tech. Rep. ISEC2003-20, IEICE (2003).

[10] Steinfeld, R., Bull, L. and Zheng. Y.: Content extraction signatures: ICISC, pp.285-304 (2001).

[11] Klonowski, M. and Lauks, A.: Extended sanitizable signatures: ICISC, pp.343-355 (2006).

[12] Suzuki, M., Isshiki, T. and Tanaka, K.: Sanitizable Signature with Secret Information: Research Reports on Mathematical and Computing Sciences, C-215, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Available at http: //www.is.titech.ac.jp/research/research-report/C/C-215.pdf (2006).

[13] Steinfeld, R., Bull, L. and Zheng, Y.: Content Extraction Signatures: Kim, K.-c. (ed.) ICISC 2001. LNCS, vol. 2288, pp.285–304. Springer, Heidelberg (2002).

[14] Okamoto, T., Pointcheval, D.: The gapproblems: A new class of problems for the security of cryptographic schemes: Public Key Cryptography - PKC 2001, LNCS 1992, pp.104–118, Springer-Verlag (2001).

[15] Joux, A. and Nguyen, K.:.Separating Decision Diffie-Hellman from Diffie-Hellman in Cryptographic Groups: Cryptology ePrint Archive, Report 2001/003 (2001).

[16] Joux, A. and Nguyen, K.: Separating Decision Diffie-Hellman from Computational Diffie-Hellman in Cryptographic Groups: Springer J. of Cryptology, 16(4), pp.239–247 (2003).

[17] Boneh, D., Gentry, C., Lynn, B. and Shacham, H.: Aggregate and Verifiably Encrypted Signatures from Bilinear Maps: Advances in Cryptology - EUROCRYPT 2003, LNCS 2656, pp.416–432, Springer-Verlag (2003).