Deployment of Contents Protection Scheme using Digital Signature

Daichi Tanaka[1], Keiichi Iwamura[1], Masaki Inamura[2] and Kitahiro Kaneda[3]
[1]*Tokyo University of Science, 6-3-1 Niijuku, Katsushika-ku, Tokyo, 125-8585, Japan*
[2]*Tokyo Denki University, Ishizaka, Hatoyama-machi, Hikigun, Saitama, 350-0394, Japan*
[3]*Osaka Prefecture University, 1-1 Gakuencho, Naka-ku, Sakai, Osaka, 599-8531, Japan*

Keywords: Copyright Protection, Edit Control, Digital Signature, BLS Signature, Aggregate Signature, Mounting.

Abstract: With the recent progress in social media and Internet technology, it is easy for consumers to edit contents uploaded on the network and create new contents. In this situation, copyright protection related to the secondary use of uploaded content is important. In our previous works, three schemes were proposed for controlling the secondary use of content according to the author's intentions by using digital signatures. Using these schemes, an author can control the changes, deletions, additions, and diversions in each portion of his/her contents, as well as the composition of the contents. The objective of this study was to verify the practicality of this technology by mounting it. Thus, we applied this technology to contents created using "MikuMikuDance," a contents editing tool for 3D CG movies, and mounted a contents protection system. We show the manner in which this system was mounted, and describe the evaluation of the processing speed in a simulation environment.

1 INTRODUCTION

In conventional mainstream content distribution services, the service, which is a specific addresser, such as a television station or a publishing company, provides contents to consumers. Recently, however, an environment where consumers can create content has become available, and they can publish their content easily on the Internet. Platforms for the circulation of content are called consumer generation media (CGM) services. As Web services that provide CGM services, YouTube, Niko Niko, etc. are well known. Within these CGM services, contents circulation, where not only are new contents created, but also the contents that other authors have created are edited and published as new contents, prospers.

In this situation, viewing control and copy control technologies, which are the current mainstreaming copyright protection technologies, are not suitable for CGM contents, because they constrain the circulation and editing of contents. In content circulation, new copyright protection technologies, such as rights succession, where the original author's rights are inherited, and edit control, which allows an edit of contents only in alignment with the author's intentions, even if the contents are used secondarily, are required.

In our previous works, three copyright protection schemes that realize rights succession and edit control by using digital signatures were proposed. Using these schemes, a contents protection system that can control the secondary use of contents to meet the author's intentions was realized. It allows an author to control changes, deletions, additions, and diversions in each portion of his/her contents, and to control also the composition of the contents themselves using signatures.

This study was aimed at verifying the validity of this contents protection system by mounting it. We applied it to contents created by "MikuMikuDance," which is a content edit tooling for 3D CG movies. The mounted program covers only the change function using a Boneh-Lynn-Shacham (BLS) signature, which constitutes the foundations of the system. We demonstrate signature creation and signature verification for changing the contents created by "MikuMikuDance" in an actual content creation environment. We show how this function is mounted, and present an evaluation of the processing speed in a simulation environment.

In this paper, Section 2 presents the principles of the edit control schemes presented in our previous works and the basic knowledge required to use them. Section 3 describes "MikuMikuDance," which has been mounted, and Section 4 shows the restrictions

Tanaka, D., Iwamura, K., Inamura, M. and Kaneda, K.
Deployment of Contents Protection Scheme using Digital Signature.
DOI: 10.5220/0006472501300137
In *Proceedings of the 14th International Joint Conference on e-Business and Telecommunications (ICETE 2017) - Volume 2: ICE-B*, pages 130-137
ISBN: 978-989-758-257-8

on mounting and an algorithm mounted in the program, and explains the application of the algorithm to the contents in "MikuMikuDance." Section 5 describes the simulation environment and the evaluation results, and in Section 6 a summary is presented.

2 RELATED WORK

We have proposed three schemes which provide rights succession control and edit control. First, a fusion of rights succession and edit control is realized(Masaki, 2012); however, the scheme addresses the edit of only one content. Next, rights succession and edit control involving two or more contents is realized using a BLS signature (Katsuma, 2015); however, this paper proposed only scheme without how to realize it with contents. latest, correspondence to an ID based signature is realized(Tatsuya F. 2016), so that the collection and verification of a public key certificate is unnecessary. In this study, we mounted the change function proposed in second study using the BLS signature for fundamental functional verification. Therefore, in the following we describe the BLS signature and the outline of the scheme proposed in Katsuma's study.

2.1 Aggregate Signature Scheme based on Boneh-Lynn-Shacham Signature

Boneh, Lynn, and Shacham proposed an aggregate signature scheme (Boneh, 2003) based on the Boneh-Lynn-Shacham (BLS) signature scheme (Boneh, 2001) using an operation on an elliptic curve and pairing. This scheme aggregates two or more different signatures for every message into one signature, and makes the signature size a steady length that is not dependent on the number of signers.

We denote by $L = \{u_{i\_1}, \cdots, u_{i\_t}\}$ a set of signers in a group who participate in generating an aggregate signature, and by $J = \{i\_1, \cdots, i\_t\}$ a set of symbols of these signers. Then, the construction of the aggregate signature is as follows.

[Key Generation]
where g is a generator of $\mathbb{G}_1$. x_i is the value of Z_p. The key generation center calculates

$$v_i = x_i g \tag{1}$$

where x_i represents a private key of $u_i \in L$ and v_i a public key of u_i.

[Signing]
We denote by $H: \{0,1\}^* \rightarrow \mathbb{G}_2$ a one-way hash function. m_j is a message of a signer u_j. Then, signer u_j calculates

$$h_j = H(m_j) \tag{2}$$

$$\sigma_j = x_j h_j \tag{3}$$

as his/her signature corresponding to m_j. After the signers have set their signatures, all the signatures are collected and calculated

$$\sigma = \sum \sigma_j \ (j \in J) \tag{4}$$

as an aggregate signature.

[Verification]
The verifier collects $m_{i\_1}, \cdots, m_{i\_t}, \sigma, g$ and the verification keys $v_j (j \in J)$, and then, calculates $h_j = H(m_j)$ from all m_j and determines whether

$$e(g, \sigma) = \prod e(v_j, h_j) \ (j \in J) \tag{5}$$

is realized using pairing. If the aggregate signature is created correctly, the above equation is realized.

2.2 Concept of Edit Control using Signature

2.2.1 Edit Control

In the scheme presented in Katsuma's study (Katsuma, 2015), an author divides his/her content into partial contents, sets the signatures of each partial content, and aggregates these signatures to one signature for the content. Hereafter, we call the signature of the partial content the edit control signature. If an author permits editing of the partial content, he/she exhibits the edit control signature. When an editor changes the partial content, the edit control signature is deleted from the aggregate signature and a new signature of the editor's partial content is added to the aggregate signature. If the author does not permit editing of the partial contents, he/she keeps the edit control signature secret. In this case, an editor cannot edit the partial contents because he/she cannot change the edit control signature in the aggregate signature.

In this scheme, as the edit control signature to control the changes, deletions, and additions of partial content two types of signature are set: change control and deletion control. Addition control is realized by change control, as the addition of content is realized by changing empty data to real data, as discussed later. Deletion refers to changing real data to empty data. However, deletion control needs to be independent of change control. For example, the control of a fixed form, such as a four-frame cartoon, allows each frame to be changed; however, in order to prevent breaks in the fixed form the control does not allow the deletion of frames. More specifically, in a movie, credit title deletion is allowed, but a change is not.

This scheme enables not only an edit in one of the contents, such as a change, addition or deletion of partial contents, but also an edit involving two or more contents, using the diversion control signature and composition control signature. The diversion control signature controls whether a partial content can be used in other contents, and the composition control signature controls whether two contents can be compounded. However, in the case of this mounting, since an edit involving more than one content is not addressed, the explanation is omitted.

2.2.2 Entities

In this scheme(Katsuma, 2015) the *i*-th author is introduced without using the word "editor" and two entities called the *i*-th author and a verifier are defined as follows.

- The *i*-th author

The *i*-th author is involved in a work, and can set up edit control signatures for the partial contents and update the aggregate signatures. For simplicity, we express the work using a tree structure, as shown in figure 1. In this scheme, each author who is in the deepest portion of the tree is called a 1st author, and an author who is in a portion of the tree route is called the *n*-th author when the tree height is *n*-1. Therefore, *i* is defined as the author’s position in the work. The *i*-th author can set the edit control signature for the partial contents that he/she has produced or edited only when an edit is permitted by the edit control signatures defined by the (*i*-1)-th authors. In figure 1, A11 to A16 are the primary contents created by two or more 1st authors; the 2nd authors created the secondary contents A21 and A22 using the primary contents of the 1st authors. Finally, the 3rd author produces the final content A31. Here, the 2nd authors can edit partial contents according to the setting of the edit control signature by each 1st author, and the 3rd author follows the setting of the entire edit control signature by the 1st and 2nd authors.

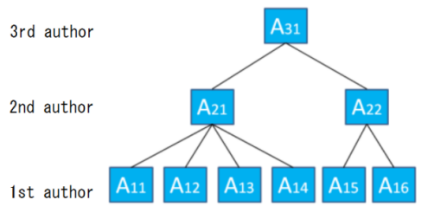


Figure 1: Examples of entities in the tree structure of a work.

- Verifier

The verifier verifies whether a content has a valid signature. If this function is available in a program that reproduces contents, we can construct a system such that the content cannot be reproduced if it does not have a valid signature.

2.2.3 Contents and Partial Contents

In this scheme, the partial contents consist of two types of data: empty and real. The empty data are placed in the portion that is to be added or deleted, and the real data constitute the displayed contents. The empty data are treated as control data for controlling addition and deletion, and the control data are not included in the displayed contents.

An author produces one or more partial contents and makes them available to the public in the following form as a content. A content comprises the start data, one or more partial contents, and the last data. The start data and the last data are the control data. Each data item is identified by an identifier.

This scheme detects edits that are contrary to an author's intention, but does not prevent just contents from turning into unjust contents by means of a processing violation, such as an overwrite or a change in the parameters. Since only the just copied contents are edited, the original contents are not affected by a processing violation. Therefore, even if an attacker does violate processing, he/she gains no advantage only because the edited contents become unjust contents.

2.2.4 Algorithm

In this section, we explain the specific algorithm presented by other work (Katsuma, 2015). However, we omit the explanation of the processing procedures other than changes, because of the page limit. In this algorithm, it is assumed that the binding between the signers and the verification keys is guaranteed by a certification authority, and the information that is being prepared is not obtained by a third party.

[Key Generation]
ID$_{ij}$ is the author ID defined according to the location of a work. For example, ID$_{11}$ is the author of contents A11 in Figure 1. Each ID$_{ij}$ has a secret key $s_{ij} \in Z_p$, and exhibits the public key $V_{ij} = s_{ij}g$. All of the signing keys are different.

[Signing]
The author always performs this signature generation process before publishing the original content.

1. Author ID$_{ij}$ determines the control permissions for changes for each partial content. Author ID$_{ij}$ determines the content ID of his/her created contents as IC_{ij}.
2. Author ID$_{ij}$ sets the start data as A_{ij0}* and the last data as A_{ijm+1}* for m partial contents. Here, d is the message of the control data. Then, author ID$_{ij}$ generates the start signature α_{ij} and the last signature β_{ij}. The constant r varies according to the edit, but here only the value over change is used.

$$A_{ij0}{}^{*} = IC_{ij} \parallel I_{ij0} \parallel da \; = \; b \; + \; c \qquad (6)$$

$$A_{ijm+1}{}^{*} = IC_{ij} \parallel I_{ijm+1} \parallel d \qquad (7)$$

$$\alpha_{ij} = s_{ij} H\big(IC_{ij} \parallel I_{ij0} \parallel H\big(A_{ij0}{}^{*}\big) \parallel r\big) \qquad (8)$$

$$\beta_{ij} = \; s_{ij} H\big(IC_{ij} \parallel I_{ijm+1} \parallel H\big(A_{ijm+1}{}^{*}\big) \parallel r\big) \qquad (9)$$

3. Author ID$_{ij}$ creates data A_{ijk}* for the message of each partial content A_{ijk}:

$$A_{ijk}{}^{*} = IC_{ij} \parallel I_{ijk} \parallel A_{ijk} \qquad (10)$$

4. Author ID$_{ij}$ generates a hash value:

$$h_{ijk} = H\big(IC_{ij} \parallel I_{ijk} \parallel H\big(A_{ijk}{}^{*}\big) \parallel r\big) \qquad (11)$$

5. Author ID$_{ij}$ generates an edit control signature for changes for every the partial content:

$$\sigma_{ijk} = s_{ijk} h_{ijk} \qquad (12)$$

6. Author ID$_{ij}$ creates an aggregate signature:

$$\sigma_{ij} = \alpha_{ij} + \sum \sigma_{ijk} + \beta_{ij} \qquad (13)$$

7. Author ID$_{ij}$ keeps σ_{ij} secret, if he/she does not allow changing.

[**Updating Signatures for Editing**]

Let us consider the case where Author ID$_{ab}$ changes the partial content A_{ijk} in the content A_{ij}, which is created by Author ID$_{ij}$, to A_{abk}, which he/she created.

8. Author ID$_{ab}$ confirms whether editing of contents A_{ij} is allowed using the signature verification. If editing of contents A_{ij} is not allowed, then the edit is stopped.
9. When a change is permitted, author ID$_{ab}$ can substitute A_{ijk} with A_{abk}, and decides the edit permission of A_{abk}.
10. Author ID$_{ab}$ generates the data A_{abk}* and the hash value for the substituted partial contents.

$$A_{abk}{}^{*} = IC_{ij} \parallel I_{ijk} \parallel A_{abk} \qquad (14)$$

$$h_{abk} = H\big(IC_{ij} \parallel I_{ijk} \parallel H(A_{abk}{}^{*}) \parallel p \parallel r\big) \qquad (15)$$

11. Author ID$_{ab}$ generates a new edit control signature σ_{abk} of A_{abk}, as in Step 5 during the signing process by using his/her signing key.
12. Author ID$_{ab}$ updates the aggregate signature as follows. Here, the author cannot update of the aggregate signature, if σ_{ij} is not disclosed.

$$\sigma_{ij} = \sigma_{ij} - \sigma_{ijk} + \sigma_{abk} \qquad (16)$$

[Verification]

The verification of the content is performed by the reproduction machine before viewing or the secondary use of the content by the viewer. Here, the entity that performs the signature verification of the content is called the verifier. The verifier performs the following processing.

First, the verifier generates the hash value of each partial content for each edit. The verifier prepares the key of the authors and verifies the following equations. If the results of the examinations are correct, the content is accepted as valid.

$$e\big(g, \sigma_{ij}\big) = \prod e\big(v_{ijk}, h_{ijk}\big) \qquad (17)$$

3 "MikuMikuDance" AND ITS CONTENTS

3.1 About "Miku Miku Dance"

"MikuMikuDance" is a 3D video production tool released by Yu Higuchi as a free program. "MikuMikuDance" was first used as a tool for creating dance movies featuring "Hatsune Miku." The most recent version allows the editing of other 3D models and many types of 3D movies. Therefore, it is very easy to start to produce a 3D CG video using "MikuMikuDance," since a considerable amount of material is available on the Internet. Figure 2 shows the creation of 3D CG movies using "MikuMikuDance."

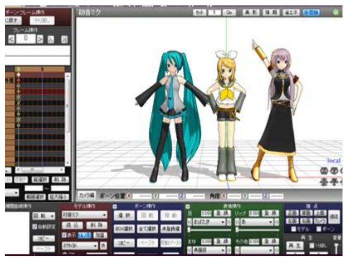


Figure 2: Creating contents in "MikuMikuDance".

3.2 Composition of Contents

In figure 2, each girl is a 3D model. A 3D CG movie comprises a set of 3D models, music, and so on. First, “MikuMikuDance” loads 3D models, music files, and voice files as material of 3DCG videos and creates a 3DCG movie by editing them. The correspondence with the contents of "MikuMikuDance," shown in Table 1, was defined in Katsuma’s paper (Katsuma, 2015) and the signature creation tool and signature verification program were created for them.

Table 1: Relationships in our implementation.

Define in Katsuma’s study	Define in "MikuMikuDance"
A set of contents	3DCG movie
A content	3D model

Partial contents	Bone(shape)
	Morph(face emotion)
	Motion(physical motion)

In "MikuMikuDance," the operation of each part of a 3D model is not defined frame by frame. "MikuMikuDance" controls each partial content of a 3D model through operation directions such as "A specific part is moved at a specific time to a specific position." For example, “arm is at upper left in Frame 5, lower left in Frame 15, and returns to this point in Frame 30.” The motion in the meantime is added by "MikuMikudance." The contents created by "MikuMikuDance" specify the appearance, motion, etc. using a pmm file. Figures 3 and 4 illustrate this. The description of one motion is defined as one partial content, and a set of descriptions comprising the whole motion of a 3D model is defined as one content. Finally, two or more contents are compounded and saved as one 3D CG movie.

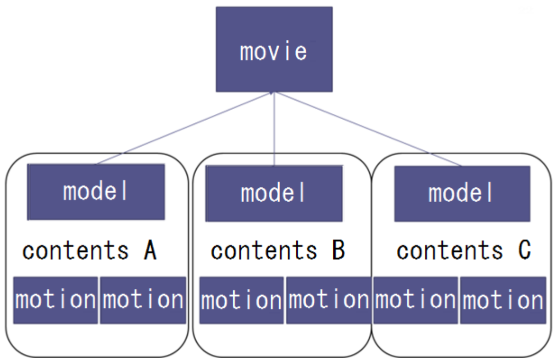


Figure 3: Construction in .pmm files.

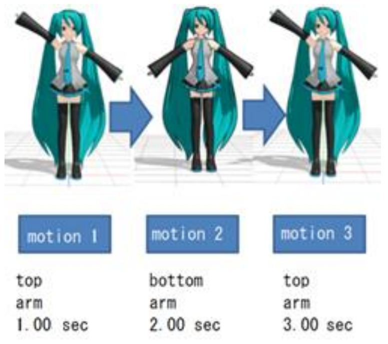


Figure 4: Direction of 3D models.

4 RESTRICTION IN IMPLEMENTATION

We aimed at mounting all the functions presented in our past work. However, it is difficult to mount all of them in time and functionally. For example, empty data, which are not displayed although they exist to control addition and deletion, cannot be prepared in “MikuMikuDance.” Therefore, addition and deletion cannot be realized. However, the aim of this mounting was to demonstrate the new edit control, and to show that the processing speed for signature creation and verification is practical. Therefore, we focused on the change processing procedure described in 2.2.4, which is a basic function, and evaluated the performance of the system.

In addition, when edits that are not allowed are detected, the program displays a processing violation detection message immediately. However, since this program is not unified with “MikuMikuDance,” the actual processing of “MikuMikuDance” cannot be stopped.

Below, we show the mounted algorithm and an example of processing the signature creation and verification using the mounted program.

4.1 Mounted Algorithm

The key generation described in 2.2.4 is performed a priori and set up. The process shown in figure 5(a) includes the pretreatment before the signature process described in 2.2.4. The pretreatment consists of the extraction of each partial content. The pmm data of original content consist of various types of data, one of which constitutes information about the 3D model, for example, “shape,” “move,” and “emotion.” Another type of data consists of temporary information used for editing. The data used for the signature are only the bone, morph, and motion of the

3D model in Table 1, and they are extracted as the target file. The signing described in 2.2.4 is performed for each partial content in the target file.

The process shown in figure 5(b) is the signature update process after an edit. This program saves the hash value of the partial content before the edit. After an editor edits a content, a hash value is created for the edited partial content and the program compares the two hash values. If there are some differences, a change is detected. After this detection, the updating signature described in 2.2.4 is performed and the signatures are updated.

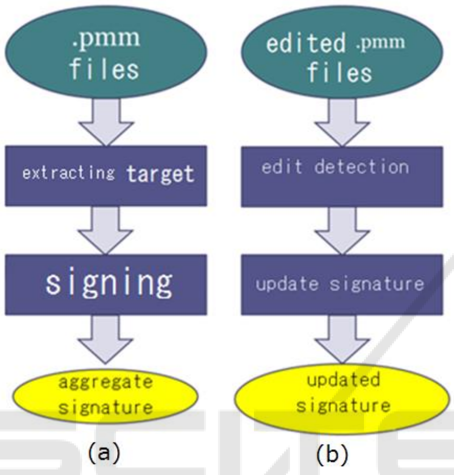
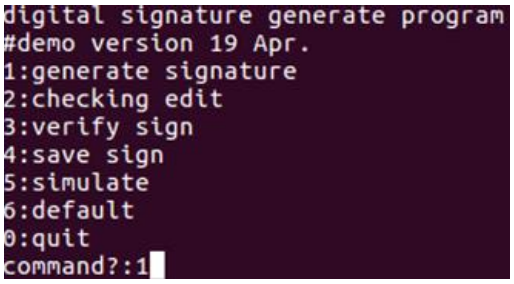


Figure 5: Overview of copyright protect algorithm.

4.2 Concrete Simulation Sample

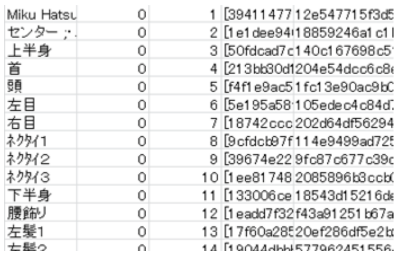
4.2.1 Extract Partial Contents

Our program runs with the command lines shown in figure 6; the created target file described in 4.1 is shown in figure 7. Figure 6 shows the title of the program and the completion date of the latest version. Then, the following steps are executed: 1. signing, 2. extraction of edited parts, 3. signature verification, 4. saving of the signature, 5. verification of signatures, 0. end; the input of a processing number.



```
digital signature generate program
#demo version 19 Apr.
1:generate signature
2:checking edit
3:verify sign
4:save sign
5:simulate
6:default
0:quit
command?:1
```

Figure 6: Running copyright protection program.



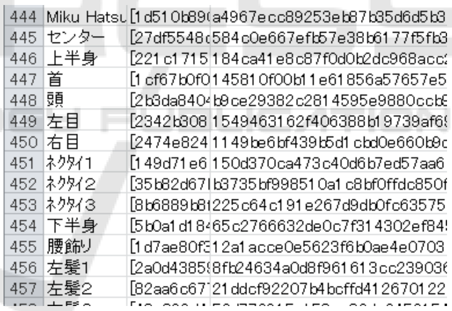
Miku Hatsu	0	1	[39411477	12e547715f3d5
センター ;・.	0	2	[1e1dee94	18859246a1c1
上半身	0	3	[50fdcad7c	140c167698c5
首	0	4	[213bb30d1	204e54dcc6c8
頭	0	5	[f4f1e9ac5	1fc13e90ac9b0
左目	0	6	[5e195a58	105edec4c84d
右目	0	7	[18742ccc	202d64df56294
ﾈｸﾀｲ1	0	8	[9cfdcb97f	114e9499ad72
ﾈｸﾀｲ2	0	9	[39674e22	9fc87c677c39c
ﾈｸﾀｲ3	0	10	[1ee81748	2085896b3ccb
下半身	0	11	[133006ce	18543d15216d
腰飾り	0	12	[1eadd7f32	f43a91251b67a
左髪1	0	13	[17f60a28	20ef286df5e2b

Figure 7: Excel view of generated target file.

In figure 7, the Japanese strings represent the names of the parts of the 3D model. The second column indicates whether the signature can be changed, where 1 means enable and 0 means disable. The numbers 1, 2, 3… are the serial numbers of the partial contents. The hexadecimal strings are the hash values of the partial contents.

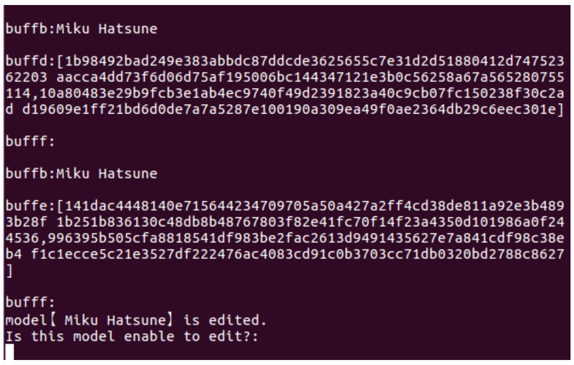
4.2.2 Create Signatures

Using the target file, the digital signatures as edit control signatures are calculated, as described in 4.1. The signatures are also stored in an Excel signature file. In this program, all the signatures are managed in an Excel file, although each signature is attached to each partial content in the scheme.



444	Miku Hatsu	[1d510b89	a4967ecc89253eb87b35d6d5b3
445	センター	[27df5548	584c0e667efb57e38b6177f5fb3
446	上半身	[221c1715	184ca41e8c87f0d0b2dc968acc
447	首	[1cf67b0f0	14581 0f00b11e61856a57657e5
448	頭	[2b3da840	b9ce29382c2814595e9880ccb
449	左目	[2342b308	154946316 2f406388b19739af6
450	右目	[2474e824	1149be6bf439b5d1 cbd0e660b9c
451	ﾈｸﾀｲ1	[149d71e6	150d370ca473c40d6b7ed57aa6
452	ﾈｸﾀｲ2	[35b82d67	b3735bf998510a1 c8bf0ffdc850f
453	ﾈｸﾀｲ3	[8b6889b8	225c64c191e267d9db0fc63575
454	下半身	[5b0a1d18	65c2766632de0c7f314302ef84
455	腰飾り	[1d7ae80f	12a1acce0e5623f6b0ae4e0703
456	左髪1	[2a0d4385	8fb24634a0d8f961613cc23903
457	左髪2	[82aa6c67	21ddcf92207b4bcffd4126701 22

Figure 8: Excel view of generated signature file.



```
buffb:Miku Hatsune

buffd:[1b98492bad249e383abbdc87ddcde3625655c7e31d2d51880412d747523
62203 aacca4dd73f6d06d75af195006bc144347121e3b0c56258a67a565280755
114,10a80483e29b9fcb3e1ab4ec9740f49d2391823a40c9cb07fc150238f30c2a
d d19609e1ff21bd6d0de7a7a5287e100190a309ea49f0ae2364db29c6eec301e]

bufff:

buffb:Miku Hatsune

buffe:[141dac4448140e715644234709705a50a427a2ff4cd38de811a92e3b489
3b28f 1b251b836130c48db8b48767803f82e41fc70f14f23a4350d101986a0f24
4536,996395b505cfa8818541df983be2fac2613d9491435627e7a841cdf98c38e
b4 f1c1ecce5c21e3527df222476ac4083cd91c0b3703cc71db0320bd2788c8627
]

bufff:
model【 Miku Hatsune】 is edited.
Is this model enable to edit?:
```

Figure 9: Signature update program.

4.2.3 Verification of Signature

Figure 9 shows the screen for verifying whether the updated signature and the signature for the edited contents are in agreement.

5 SIMULATION METHOD AND EVALUATION

5.1 Simulation Environment

The simulation was executed in a computer running the Oracle Virtual Box and a virtual OS. Table 2 shows the details of the simulation environment. The simulation compiled the program written in C language using GCC (version 5.4.0), and ran it on the Ubuntu terminal. For each operation described in 2.2.4, TEPLA was used for the pairing operation, elliptic curve calculation, and operation in the finite field on the computer. TEPLA was updated to the latest version on 20 Dec. 2015.

Table 2: System environment in simulation.

Main machine	Details
OS	Windows 7 Home Premium SP1 64bit
CPU	Intel® Core™ i5-2450 M CPU 2.5 GHz
RAM	8.00 GB
Virtual machine	Oracle Virtual Box
Version	Version 5.0.26
Virtual OS	Ubuntu16.04 LTS
CPU	Same as main machine
RAM	3.00 GB

This environment is limited as compared with the most recently developed computers. However, if this simulation was executed at a speed permissible in practice, it would show that the content protection scheme is effective and practical.

5.2 Outline of Simulation

To verify the operation of the mounted program and evaluate it, we changed the number of the 3D models and the number of partial contents comprising each model, measured the time spent on processing two or more times, and recorded the average time.

Version 9.26 of "MikuMikuDance" was used. The number of 3D models and of their partial contents of the MikuMikuDance standard attachment used in this simulation are shown in Table 3.

Table 3: Number of target 3D model(s) and partial contents.

Models	Partial contents
1	4052
2	6999
3	10070
4	12739
5	15800

5.3 Simulation Results

5.3.1 Generate Signature

We measured the time taken by the signature generation process five times for every number of models and partial contents. The total required time and the time required per 1 partial content are shown in Table 4.

The signature targets include the aggregate signatures.

Table 4: Time spent on signature generation.

Models	Number of signature targets	Required time (ms)	Time per 1 target (ms)
1	4052	2951.01	0.7283
2	6999	5683.37	0.8120
3	10070	7583.61	0.7531
4	12739	9912.55	0.7781
5	15800	11677.51	0.7391

As seen in Table 4, the processing speed per one target is satisfactory and practical: 0.7 ms–0.8 ms, and all the throughputs increase in proportion to the total time for the targets. In general, a movie of about 25 sec comprising one model needs approximately 3000 signatures. In this case, the signature generation time of one movie is about 3 sec. If the signature generation process is performed whenever some contents are created, a signature generation time of several seconds can be considered satisfactory and practical.

5.3.2 Signature Verification

We measured the time required for signature verification 10 times for every number of models and partial contents. The total required time and the time required per 1 partial content are shown in Table 5.

The signature targets include the aggregate signatures.

Table 5: Time spent on signature verification.

Models	Number of signature targets	Required time (ms)	Time per 1 target (ms)
1	4052	2150.18	0.5306
2	6999	4174.38	0.5964
3	10070	5967.22	0.5926
4	12739	7661.56	0.6014
5	15800	11336.42	0.7175

When first verifying the signatures of a content, all the signatures for all the content must be verified, and the time varies according to the number of signatures. However, during editing the signature verification process is performed only for the edited portion and can be performed for each edit in several seconds. This can also be considered satisfactory and practical.

6 SUMMARY

In this study, we developed a program to generate and verify signatures and mounted a contents protection system for"MikuMikuDance," which is a content editing tool for 3D CG movies. Through this mounting, we showed that the scheme can be applied not only in theory but also in an actual application, and that the processing speed achieved is satisfactory and practical.

In future work, we will extend the functions to handle addition, deletion, and diversion of partial contents and the composition of contents, which includes all the functions in the scheme presented in Katsuma's work (Katsuma, 2015). In addition, we will also examine mounting using the ID-based signature (Tatsuya, 2016) and the fusion of edit control and rights succession control(Masaki, 2012) to realize many functions. Improvement in the speed of processing and the system's application to various applications should also be considered.

REFERENCES

Youtube, https://www.youtube.com/

Niko Niko Douga, www.nicovideo.jp/

Yu H. ,VPVP http://www.geocities.jp/higuchuu4/

Laboratory of Cryptography and Information Security, University of Tsukuba, "TEPLA" http://www.cipher.risk.tsukuba.ac.jp/tepla/index.html

Masaki I., Keiichi I., 2012. An Expression of a Quotation Process in Contents with a New Tree-structure-specified Aggregate Signature Scheme. *Information Processing Society of Japan.* Vol. 53 No. 9 pp.2267-2278, Sep.

Katsuma K., Masaki I., Kitahiro K., Keiichi I., 2015. Content Control Scheme to Realize Right Succession and Edit Control, *International joint Conference on e-Business and Telecommunications*.

Tatsuya F., Masaki I., Keiichi I., 2016. Copyright Protection Scheme to Realize Edit Control Based on ID-Based Signature, *ISSN 1882-0840*, pp.1124-1129, Oct

Boneh, D., Lynn, B., Shacham, H., 2001. Short Signatures from the Weil Pairing. *Advances in Cryptology – ASIACRYPT.* LNCS 2248, pp.514–532, Springer.

Boneh, D., Gentry, C., Lynn, B., Shacham, H., 2003. Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. *Advances in Cryptology – EUROCRYPT.* LNCS 2656,pp.416–432, Springer.