Sanitizable signatures with strong transparency in the standard model

Shivank Agrawal, Swarun Kumar, Amjed Shareef and C. Pandu Rangan

Theoretical Computer Science Lab,
Department of Computer Science and Engineering,
Indian Institute of Technology Madras, India.
{shinku100,swarun.s,amjedshareef}@gmail.com, prangan@iitm.ac.in

Abstract. Sanitizable signatures provide several security features which are useful in many scenarios including military and medical applications. Sanitizable signatures allow a semi-trusted party to update some part of the digitally signed document without interacting with the original signer. Such schemes, where the verifier cannot identify whether the message has been sanitized, are said to possess strong transparency. In this paper, we have described the first efficient and provably secure sanitizable signature scheme having strong transparency under the standard model.

Key words: sanitizable signatures, strong transparency, standard model

1 Introduction

Applications like e-government and e-tax payment systems require appropriate alteration of digitally signed documents in order to hide personal information. Sanitizable signatures came into much attention when recently, government entities were forced to disclose documents owing to disclosure laws. In the past, when secret paper documents were made declassified, hiding of sensitive or personal information in the document was done by blackening-out (sanitizing) relevant sections of the documents. A digital signature, however, prohibits any alteration of the original message once it is signed. So, in the world of digital signatures, sanitization cannot be done.

A sanitizable signatures protects the confidentiality of a specified part of the document while ensuring the integrity of the document. A solution for this problem was proposed earlier in [1] as *content extract signatures*. In 2005, Ateniese et al. [2] introduced *sanitizable signatures* which can alter the signed document instead of hiding it. A sanitizable signature scheme is a signature scheme which allows a designated party, called the *sanitizer*, to hide certain parts of the original message after the message is signed, without interacting with the signer. The verifier confirms the integrity of disclosed parts of the sanitized document from the signature and sanitized document. In other words, a sanitizable signature scheme allows a semi-trusted *sanitizer* to modify designated portions of the document and produce a valid signature on the legitimately modified document without any interaction with the original signer. These designated portions of the document are blocks or segments explicitly indicated as mutable under prior agreement between the signer and the sanitizer. The sanitizer can produce a valid signature only if it modifies these portions and no other parts of the message. Following these works several authors [3–8] proposed various sanitizable signature schemes with different properties.

There are different types of sanitizable schemes present in literature. In some schemes, sanitization on any part of the message can be performed by the sanitizer, while in other schemes, sanitization on some parts of the messages can be restricted by the signer, and this decision can be made even after the signing of the message performed by a signer or anyone else. Transparency is a another property of sanitizable signature schemes [2, 9]. If the verifier knows which part of the document is sanitized, then the scheme has no transparency. If he does not know whether the message is sanitized, then the scheme has weak transparency. If he also does not know whether the message can be sanitized, then the scheme is said to have *strong transparency.*

1.1 Our contributions

In this paper, we have provided two protocols for strong transparency in the *standard model* using bilinear pairing. Our construction is based on Waters's scheme [10]. These are the *first* efficient and secure schemes which provide strong transparency under the standard model. In our first protocol we achieve strong transparency by providing some secret information to the sanitizer, where the portions of the message to be sanitized are specified by the signer. In our second protocol we remove the need to send secret information to the sanitizer on a per-message basis, provided that the blocks of message which need to be sanitized are fixed beforehand. This requirement may hold in several kinds of documents such as forms, databases, etc. The length of our sanitized signature is equivalent to the length of Waters' [10] signature, hence shorter than the signatures produced by other protocols. Our scheme uses techniques similar to the sanitizable signature scheme discussed in [9] and provides additional properties. The scheme in [9] does not provide transparency. We also compare our scheme with other sanitizable signature schemes proposed in literature.

1.2 Applications

As described in [2], sanitizable signatures have several applications.

Multicast and Database Applications. Sanitizable signatures are quite well-suited for customizing authenticated multicast transmissions. For example, in a subscription-based internet multimedia database, sponsors may wish to insert personalized commercials into messages at various points of the broadcast.

It is desirable to authenticate these messages to allow the subscribers to distinguish legitimate contents from spam. Since real-time authentication may be too costly, one solution is for each vendor to sign the commercial once and allow the database administrator to customize the individual commercials by replacing the generic identity field with the actual subscriber's identity, at various points of the commercial. This way, the subscriber can verify that the commercial comes from a legitimate source (i.e., it is not spam) and the sponsors do not have to sign each customized broadcast. Furthermore, the database administrator is not forced to divulge personal information of its subscribers without their consent. A related application of sanitized signatures is editing movie content. Depending on the age of the subscriber, the administrator can replace offensive language with watered-down substitutes rather than bleep out the words. Again, sanitized signatures provide the desired benefits.

Medical Applications. Sanitizable signatures can be used to ensure the integrity, authenticity, and anonymity of public health information in medical records. In general, sanitizable signatures can accommodate different levels of data de-identification, supporting the minimum necessary disclosure standard of the existing privacy laws. This provides flexibility not available in redactable signatures.

Similarly, sanitizable signatures may be applied in several other scenarios such as secure routing, e-governance, etc.

2 Preliminaries

2.1 Bilinear Pairing

Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be a multiplicative groups of prime order p. The elements $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. A bilinear pairing is a map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow G_T$ with the following properties:

1. **Bilinear**: $e({g_1}^a, {g_2}^b) = e(g_1, g_2)^{ab}$ for all $g_1 \in \mathbb{G}_1$,$g_2 \in \mathbb{G}_2$, where $a, b \in \mathbb{Z}_p$.
2. **Non-degenerate**: There exists $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ such that $e(g_1, g_2) \neq 1$; in other words, the map does not send all pairs in $\mathbb{G}_1 \times \mathbb{G}_2$ to the identity in $\mathbb{G}_T$.
3. **Computability**: There is an efficient algorithm to compute $e(g_1, g_2)$ for all $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$.

2.2 Security Assumptions

Definition 1. The Computational Diffie-Hellman (CDH) problem is that, given g, g^x, $g^y \in \mathbb{G}$ for unknown x,$y \in Z_p*$, to compute g^{xy}.
We say that the *ϵ-CDH assumption* holds in $\mathbb{G}$ if no polynomial-time algorithm has non-negligible probability ϵ in solving the CDH problem.

2.3 Sanitizable Signature

A sanitizable signature scheme is a signature scheme that allows the sanitizer to sanitize certain portions of the document and to generate the valid signature of the resulting document with no interaction with the signer. A sanitizable signature is processed by three parties consisting of a signer, a sanitizer, and a verifier. The signer generates the signature assuring the authenticity of the document. The sanitizer receives the document and its signature from the signer. The sanitizer generates the sanitized document and its signature without any help of the signer. The verifier receives the sanitized document and its signature from the sanitizer. The verifier accepts the signature only if he verifies the authenticity of the disclosed document.

2.4 Transparency

A sanitizable signature scheme may have various levels of *transparency*, which we define below:

1. **No transparency.** The verifier knows which part of the document is sanitized.
2. **Weak transparency.** The verifier does not know if the message is sanitized. The verifier only knows if the message can be disclosed and sanitizing is prohibited or not.
3. **Strong transparency.** The verifier does not know if the message has been sanitized. In this model no extra information is sent to the verifier other than message and a signature.

2.5 State information

In our first scheme, the signer can control the states of the bits of the document, i.e., whether sanitization is allowed or sanitization is prohibited. This state information is kept secret to achieve strong transparency. The signer generates the secret information for each message of the document. The secret information is necessary to generate the signature of the sanitized document. The signer sends the secret information of the message to the sanitizer if he allows the sanitizer to sanitize the message. Otherwise he does not send the secret information of the message to the sanitizer.

In our second scheme, the sanitizer is given the power to control certain bits of the message a priori in the set-up phase. In this case, no secret information needs to be sent to the sanitizer by the signer.

2.6 Scheme Outline

- **Key Generation.** Algorithm *KeyGen*, executed by the PKG, takes as input a security parameter 1^k and outputs public parameters *param*, public key and secret key pair for the signer (PK, SK) and secret information of the sanitizer SK', if any.

- **Signing.** Algorithm *Sign*, executed by the signer, takes as input a document M, public parameters *param* and secret key SK. Let $M = m_0m_1 \cdots m_n \in \{0,1\}^n$, where m_i is defined as the bit at index i of message M. Let $I_{\mathcal{S}} \subseteq \{1, \cdots, n\}$ denote the set of indices that the sanitizer is allowed to modify. The signing algorithm outputs a document M, two signatures(σ_1, σ_2) of M and secret information SI for the sanitizer, if any.
- **Sanitization.** Algorithm *Sanitize*, executed by the sanitizer, takes message M, public parameters *param*, signature σ on M, sanitizer's secret key SK', if any, secret information from the signer SI, if any, and outputs a message M' and sanitized signature σ'.
- **Verification.** Algorithm *Verify*, executed by the verifier, takes as input an unsanitized document and signature (M, σ) or a sanitized document and signature (M',σ'), public parameters *param*, and public key PK of signer, outputs *accept* or *reject*. The strong transparency property requires that the verifier not be able to find out whether the document is sanitized or not. Hence the verification procedure remains the same for both sanitized and unsanitized documents.

3 Security Model

3.1 Correctness

We require that $\mathit{Verify}(\sigma, M, PK, \mathit{param}) = \mathit{accept}$, for an unsanitized message M if :

1. $(PK, SK, SK', \mathit{param}) \leftarrow \mathit{KeyGen}(1^k)$,
2. $(\sigma, SI) \leftarrow \mathit{Sign}(M, SK, \mathit{param})$,

We additionally require that $\mathit{Verify}(\sigma', M', PK, \mathit{param}) = \mathit{accept}$, for an sanitized message M' if:

1. $(PK, SK, SK', \mathit{param}) \leftarrow \mathit{KeyGen}(1^k)$,
2. $(\sigma, SI) \leftarrow \mathit{Sign}(M, SK, \mathit{param})$,
3. $(M', \sigma') \leftarrow \mathit{Sanitize}(M, \sigma, PK, SK', SI, \mathit{param})$

3.2 Unforgeability

We have the following game $\mathsf{Exp}_{\mathrm{unf}}$ for unforgeability:

1. The simulator S gives *param* and PK to the adversary A.
2. A is allowed to query the signing oracle q_s times adaptively. During the j^{th} query, on inputing a document $M_j = m_{j,1} \cdots m_{j,n}$, the oracle returns the corresponding signature σ_j on M_j.
3. Finally A outputs a document M^* , a signature σ^*.

A wins if $\mathit{Verify}(\sigma^*, M^*PK, \mathit{param}) = \mathit{accept}$ and the message M^* is not equal to any query message M_j for $1 \leq j \leq q_s$.

Note that the adversary is *not provided a sanitization oracle*, as a sanitized signature is indistinguishable from a normal signature by the signer on the same message. This follows from strong transparency. This security model for unforgeability is also present in [11].

Definition 2. A sanitizable signature scheme is (ϵ, q_s)- unforgeable if there is no randomized polynomial time adversary winning the above game with probability at least ϵ with at most q_s queries to the signing oracle.

3.3 Indistinguishability

We have the following game $\mathsf{Exp}_{\text{ind}}$ for indistinguishability:

1. The simulator S gives *param* and PK to the adversary A.
2. A is allowed to query the signing oracle q_s times adaptively. The oracle is the same as the one in the game for unforgeability.
3. A sends two different signatures σ_0, σ_1 on M_0, M_1 respectively and a sanitized message M', where M' differs from M_0 and M_1 only at bits that are allowed to be sanitized.
4. S picks a random bit b and sends σ'_b to A which is the signature obtained from the sanitization of message M_b.
5. Finally, A outputs bit b'

A wins the game if $b = b'$. The advantage of A is $|\Pr[b = b'] - 1/2]|$.

Definition 3. A sanitizable signature scheme is said to be unconditionally indistinguishable if there is no adversary winning the above game with advantage greater than 0 with any number of queries to the signing oracle.

3.4 Immutability

We have the following game $\mathsf{Exp}_{\text{imm}}$ for immutability. Let $I_{\mathcal{S}}$ be the set of positions of the bits in the message that the sanitizer is allowed to modify. Here the adversary is a sanitizer who attempts to sanitize bits outside his permissible set $I_{\mathcal{S}}$.

1. A sends a challenge set $I_{\mathcal{S}}$, the set of bit positions where sanitization is allowed.
2. The simulator S gives the public parameters *param* and PK to the adversary A.
3. In scheme-2, the one-time secret information corresponding to the set $I_{\mathcal{S}}$ is also given to the adversary.

4. A is allowed to query the signing oracle q_s times adaptively. During the j^{th} query, on input a document $M_j = m_{j,1} \cdots m_{j,n}$, the oracle returns the corresponding signature σ_j on M_j.
5. In scheme-1, A additionally obtains secret values SK_j along with σ_j. This enables A to sanitize bits at positions $I_{\mathcal{S}}$.
6. Finally, A outputs a document $M^* = m_1^* \cdots m_n^*$, a signature σ^*, where $\forall j \in \{1, \cdots, q_s\}\ \exists i \notin I_{\mathcal{S}} : m_{j,i} \neq m_i^*$.

A wins the game if signature σ^* on M^* verifies successfully. The advantage of A is the probability that A succeeds. This security model is in accordance with [12]. Note that accountability is not required in our model as this property compromises the unconditional indistinguishability of our scheme.

Definition 4. A sanitizable signature scheme is ϵ- immutable if there is no randomized polynomial time adversary winning the above game with probability at least ϵ.

4 Scheme 1

4.1 Outline

This scheme provides a strong transparent sanitizable signature protocol where the signer is proactive in deciding which bits need to be sanitized and by which sanitizer. The signature protocol is based on the Water's signature scheme [10]. The signer sends the indices which are permitted to be sanitized as well as one time secret information that enables sanitization of the relevant portions of the message to the sanitizer, in a secure fashion. The sanitizer may replace those portions of the message by an appropriate message of his choice. In this section, we describe this scheme, provide security proofs based on the CDH assumption, as well as show extensions by which the signer can exert more control on how the message is sanitized.

4.2 Scheme Description

KeyGen. Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be groups of prime order p. Given a pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. We denote by n the number of bits of the message m. Let $g \in \mathbb{G}_1$ and $g_2, u', u_1, \cdots, u_n \in \mathbb{G}_2$.

Public parameters: $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, p, g_2, u', u_1, \cdots, u_n$. Public key of the signer is $g_1 = g^\alpha$, where $\alpha \in \mathbb{Z}_p^*$.

Private parameters: Private key of the signer is $\alpha \in_R \mathbb{Z}_p^*$.

Sign. Let m be the n-bit message $m_1 m_2 \cdots m_n \in \{0,1\}^n$. Signer randomly picks $r \in \mathbb{Z}_p^*$ and outputs the following values (σ_1, σ_2):

$$(\sigma_1 = g_2^{\alpha}(u' \prod_{i=1}^{n} u_i^{m_i})^r, \sigma_2 = g^r)$$

Let $I_{\mathcal{S}}$ be the set of indices that the sanitizer is permitted to modify. Then the signer sends the values u_i^r $\forall i \in I_{\mathcal{S}}$ to the sanitizer in a secure channel. Alternately, these values may be encrypted by the public key of the sanitizer and sent across.

Sanitize. The sanitizer obtains the values (σ_1, σ_2), and the secret information u_i^r $\forall i \in I_{\mathcal{S}}$ from the signer. It runs the verification protocol to check if the signature is valid. Let m' be the message whose signature is sought, which differs from m at positions $I \subseteq I_{\mathcal{S}}$. Define $I_1 = \{i \in I : m_i = 0, m'_i = 1\}$, $I_2 = \{i \in I : m_i = 1, m'_i = 0\}$. The sanitizer chooses $\tilde{r} \in_R \mathbb{Z}_p^*$. Then the required sanitized signature is:

$$(\sigma'_1 = \sigma_1 \frac{\prod_{i \in I_1} u_i^r}{\prod_{i \in I_2} u_i^r} u'^{\tilde{r}} \prod_{i=1}^{n} u_i^{m'_i \tilde{r}}, \sigma'_2 = \sigma_2 g^{\tilde{r}})$$

Verify. The verifier receives the tuple: (σ_1, σ_2) on a message m,
Verifier checks if the following relation holds from public parameters:

$$e(g, \sigma_1) \stackrel{?}{=} e(g_1, g_2) e(\sigma_2, u' \prod_{i=1}^{n} u_i^{m_i})$$

Note that the verification protocol is same for a sanitized and non-sanitized message.

4.3 Security

Correctness. To show correctness, we need to show that any valid normal signature, as well as sanitized signature verifies successfully.

Verification: The signature σ, on a given message m is given by the two-tuple $(\sigma_1 = g_2^{\alpha}(u' \prod_{i=1}^{n} u_i^{m_i})^r \ , \sigma_2 = g^r)$. If valid, then clearly:

$e(g_1, g_2) e(\sigma_2, u' \prod_{i=1}^{n} u_i^{m_i})$
$= e(g^a, g_2) e(g^r, u' \prod_{i=1}^{n} u_i^{m_i})$
$= e(g, g_2^a) e(g, (u' \prod_{i=1}^{n} u_i^{m_i})^r)$
$= e(g, g_2^a (u' \prod_{i=1}^{n} u_i^{m_i})^r)$
$= e(g, \sigma_1)$

Hence, a valid signature satisfies the verification equation.

Sanitization: A sanitized signature is obtained as:

$$(\sigma_1' = \sigma_1 \frac{\prod_{i \in I_1} u_i^r}{\prod_{i \in I_2} u_i^r} u'^{\tilde{r}} \prod_{i=1}^{n} u_i^{m_i' \tilde{r}}, \sigma_2' = \sigma_2 g^{\tilde{r}})$$

where $I_1 = \{i \in I_{\mathcal{S}} : m_i = 0, m_i' = 1\}$, $I_2 = \{i \in I_{\mathcal{S}} : m_i = 1, m_i' = 0\}$.
We note that $m_i' - m_i$ is 1 when $i \in I_1$, -1 when $i \in I_2$,and 0, otherwise. Hence, we can see that:

$$\begin{aligned}
\sigma_1' &= \sigma_1 u'^{\tilde{r}} \textstyle\prod_{i \in I_1} u_i^r / \prod_{i \in I_2} u_i^r \prod_{i=1}^n u_i^{m_i' \tilde{r}} \\
&= \sigma_1 u'^{\tilde{r}} \textstyle\prod_{i=1}^n u_i^{r(m_i' - m_i)} \prod_{i=1}^n u_i^{m_i' \tilde{r}} \\
&= g_2^{\alpha} u'^{r} u'^{\tilde{r}} \textstyle\prod_{i=1}^n u_i^{r(m_i)} \prod_{i=1}^n u_i^{r(m_i' - m_i)} \prod_{i=1}^n u_i^{m_i' \tilde{r}} \\
&= g_2^{\alpha} u'^{(r+\tilde{r})} \textstyle\prod_{i=1}^n u_i^{(r+\tilde{r})(m_i')}
\end{aligned}$$

The sanitized signature is of the form $(\sigma_1' = g_2^{\alpha} (u' \prod_{i=1}^n u_i^{m_i'})^{(r+\tilde{r})}, \sigma_2' = g^{(r+\tilde{r})})$, whose distribution is identical to a regular signature on m' by the signer. Hence, a sanitized signature also satisfies the verification equation.

Unforgeability. We prove the following theorem about unforgeability.

Theorem 1. *The proposed sanitizable signature scheme in scheme-1 is (ϵ, q_s)-unforgeable under the ϵ'-CDH assumption where $\epsilon \leq (8q_s^2(n+1)^2 + 2)\epsilon' + 2/p$, where q_s is the polynomial number of queries.*
Proof. Assume there is a (ϵ, q_s)-adversary A exists. We shall formulate another probabilistic polynomial time (PPT) algorithm B that uses A to solve the CDH problem with probability at least ϵ' and in time at most t'. B is given a problem instance as follow: Given a group $\mathbb{G}$, a generator $g \in \mathbb{G}$, two elements $g^a, g^b \in \mathbb{G}$. It is asked to output another element $g^{ab} \in G$. In order to use A to solve for the problem, B needs to simulates a challenger and the signing oracle for A. B does it in the following way (Recall here that g^a and g^b are the input for the CDH problem that B should solve).

Setup Phase. Let $l = 2q_s$. B randomly selects an integer k such that $0 \leq k \leq n$. Also assume that $l(n+1) < p$, for the given values of q_s and n. It randomly selects:

1. $x' \in_R \mathbb{Z}_l; y' \in_R \mathbb{Z}_p$
2. $\hat{x}_i \in_R \mathbb{Z}_l$, Let $\hat{X} = \{\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_n\}$.
3. $\hat{y}_i \in_R \mathbb{Z}_p$, Let $\hat{Y} = \{\hat{y}_1, \hat{y}_2, \cdots, \hat{x}_n\}$.

We further define the following functions for binary string $M = (m_1, m_2, \cdots, m_n)$, where $m_i \in \{0, 1\}$ $1 \leq i \leq n$, as follows:

$$F(M) = x' + \sum_{i=1}^{n} \hat{x}_i m_i - lk$$

$$J(M) = y' + \sum_{i=1}^{n} \hat{y}_i m_i$$

B constructs a set of public parameters as follows:

$$g_2 = g^b, u' = g_2^{-lk+x'} g^{y'}, u_i = g_2^{\hat{x}_i} g^{\hat{y}_i}, i = 1, \cdots, n$$

We have the following equation:

$$u' \prod_{i=1}^{n} u_i^{m_i} = g_2^{F(M)} g^{J(M)}$$

All the above public parameters and public key $g_1 = g^a$ are passed to A.

Simulation Phase. B simulates the signing oracle as follow. Upon receiving the j^{th} query for a document M_j, although B does not know the secret key, it can still construct the signature by assuming $F(M_j) \neq 0 \bmod p$. It randomly chooses $r_j \in_R \mathbb{Z}_p$ and computes the signature as

$$\sigma_{1,j} = g_1^{-J(M_j)/F(M_j)} (g_2^{F(M_j)} g^{J(M_j)})^{r_j}, \sigma_{2,j} = g_1^{-1/F(M_j)} g^{r_j}$$

By letting $\hat{r}_j = r_j - a/F(M_j)$, it can be verified that $(\sigma_{1,j}, \sigma_{2,j})$ is a valid signature on M_j as shown below:

$$\begin{aligned}
\sigma_{1,j} &= g_1^{-\frac{J(M_j)}{F(M_j)}} (g_2^{F(M_j)} g^{J(M_j)})^{r_j} \\
&= g^{-a\frac{J(M_j)}{F(M_j)}} (g_2^{F(M_j)} g^{J(M_j)})^{\frac{a}{F(M_j)}} (g_2^{F(M_j)} \ g^{J(M_j)})^{-\frac{a}{F(M_j)}} \ (g_2^{F(M_j)} g^{J(M_j)})^{r_j} \\
&= g_2^a (g_2^{F(M_j)} g^{J(M_j)})^{\hat{r}_j} \\
\sigma_{2,j} &= g_1^{-1/F(M_j)} g^{r_j} \\
&= g^{r_j - a/F(M_j)} \\
&= g^{\hat{r}_j}
\end{aligned}$$

If $F(M_j) = 0 \bmod p$, since the above computation cannot be performed (division by 0), the simulator aborts. To make it simple, the simulator will abort if $F(M_j) = 0 \bmod l$. The equivalence can be observed as follow. From the assumption that $l(n+1) < p$, it implies $0 \leq lk < p$ and $0 \leq x' + \sum_{i=1}^{n} \hat{x}_i m_i < p$ (as $x' < l, \hat{x}_i < l$). We have $-p < F(M_j) < p$ which implies if $F(M_j) = 0 \bmod p$ then $F(M_j) = 0 \bmod l$. Hence, $F(M_j) \neq 0 \bmod l$ implies $F(M_j) \neq 0 \bmod p$. Thus the former condition will be sufficient to ensure that a signature can be computed without aborting.

Challenge Phase. If B does not abort, A will return a document $M^* = m_1^* \cdots m_n^*$ with a forged signature $\sigma^* = (\sigma_1^*, \sigma_2^*)$. The algorithm B aborts if $x' + \sum_{i|m_i^*=1} \hat{x}_i - lk \neq 0 \bmod l$. From the verification equation, we can write:

$$\begin{aligned}
\sigma_1^* &= g_2^a (g_2^{F(M^*)} g^{J(M^*)})^{r^*} \\
&= g_2^a (g_2^{(x' + \sum_{i|m_i^*=1} \hat{x}_i - lk) r^*} g^{(y' + \sum_{i|m_i^*=1} \hat{y}_i) r^*} \\
&= g_2^a g^{(y' + \sum_{i|m_i^*=1} \hat{y}_i) r^*}
\end{aligned}$$

Hence the algorithm successfully computes the solution to the CDH problem:

$$Z = \sigma_1^* {\sigma_2^*}^{-y' - \sum_{i|m_i^*=1} \hat{y}_i} = g_2^a = g^{ab}$$

□

Probability Analysis. The probability that the simulation does not abort is characterized by the events A_j, A^* where:

1. A_j is the event that $F(M_j) \neq 0 \bmod l$ where $j = 1, \cdots, q_s$.
2. A^* is the event that $x' + \sum_{i|m_i^*=1} \hat{x}_i - lk = 0 \bmod p$.

The probability that B does not abort:

$$\Pr[\text{not abort}] \geq \Pr[\bigwedge_{j=1}^{q_s} A_j \wedge A^*]$$

As the adversary can at most make B abort by randomly choosing M^*, we have $\Pr[A^*] = \frac{1}{l(n+1)}$. Also noting that A_j is independent of A^* we have:

$$\begin{aligned} \Pr[\text{not abort}] &\geq \Pr[\textstyle\bigwedge_{j=1}^{q_s} A_j \wedge A^*] \\ &\geq \Pr[A^*]\Pr[\textstyle\bigwedge_{j=1}^{q_s} A_j | A^*] \\ &\geq \tfrac{1}{(l(n+1))^2}(1 - \textstyle\sum_{j=1}^{q_s} \Pr[\neg A_j | A^*]) \\ &\geq \tfrac{1}{8(n+1)^2 q_s^2} \end{aligned}$$

Indistinguishability. As shown in the correctness section, a valid signature σ_s produced by a signer on a message m' has a distribution identical to a valid sanitization of a message m_1 to result in message m' and signature σ'_a produced by the sanitizer. Similarly, the distribution is also identical to a valid sanitization of another message m_2 to result in message m' and signature σ'_b, produced by the sanitizer. Hence the signatures σ'_a and σ'_b are indistinguishable as their distributions are identical.

Immutability. We prove the following theorem to show immutability.

Theorem 2. The proposed sanitizable signature scheme in scheme-1 is ϵ-immutable under the ϵ'-CDH assumption, where there exists constant $l : \epsilon < l\epsilon'$.

Proof: We prove that the sanitizer cannot modify any bit other than bits at positions $I_{\mathcal{S}} \subseteq \{1, \cdots, n\}$ for which the values $\{u_i^r : i \in I_{\mathcal{S}}\}$ are known to the sanitizer. We will prove the following lemma on immutability in order to prove the above theorem.

Lemma 1. For any randomized polynomial time algorithm algorithm B with an advantage ϵ_b in the immutability game $\mathsf{Exp}_{\text{imm}}$ on a message of length n with access to sanitize m bits at positions $I_{\mathcal{S}}$, there exists a randomized polynomial time algorithm A with an advantage $\epsilon_a \geq \epsilon_b$ in the unforgeability game $\mathsf{Exp}_{\text{unf}}$ on a message of length $n - m$.

Proof: Assume that there exists a randomized polynomial time algorithm B which plays the immutability game $\mathsf{Exp}_{\text{imm}}$ with advantage ϵ_b with access to sanitize m bits at positions $I_{\mathcal{S}}$. Consider a randomized polynomial time algorithm A which plays the unforgeability game $\mathsf{Exp}_{\text{unf}}$ on messages of length $n-m$. Then we show that the algorithm A can simulate the challenger interacting with algorithm B, and thereby obtain an advantage $\epsilon_a \geq \epsilon_b$ in $\mathsf{Exp}_{\text{unf}}$. In the setup phase, A interacts with B and the challenger in $\mathsf{Exp}_{\text{unf}}$, denoted by C, as follows:

1. B provides A the set, $I_{\mathcal{S}}$, of bit positions where sanitization is allowed. In general, we have $I_{\mathcal{S}} \subseteq \{1, \cdots, n\}$. However for the ease of exposition we assume $I_{\mathcal{S}} = \{n - m + 1, \cdots, n\}$, where $m = |I_{\mathcal{S}}|$. Note that the argument can be easily extended for the general form of $I_{\mathcal{S}}$.
2. C provides A the public parameters $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, p, g_2, u'$, $u_1, \cdots, u_{n-m}$.
3. A chooses $t_i \in_R \mathbb{Z}_p^*$, $i = n-m+1, \cdots, n$. A sets $u'_i = g^{t'_i}$, for $i = n-m, \cdots, n$
4. A provides B the public parameters $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, p, g_2, u'$, $u_1, \cdots, u_{n-m}$, $u'_{n-m+1} \cdots, u'_n$.

In the simulation phase, for every message M_j, $j = 1, \cdots, q_s$, requested by B, A interacts with B and C as follows:

1. B requests signature for a message $M_j = m_{j,1} \cdots m_{j,n}$ from A.
2. A requests a signature for message $M_j = m_{j,1} \cdots m_{j,n-m}$ from C.
3. A obtains $(\sigma_{j,1}, \sigma_{j,2})$ from C, and sets $\sigma'_{j,1} = \sigma_{j,1} \prod_{i=n-m+1}^{n} \sigma_{j,2}^{t'_i m_{j,i}}$ and $\sigma'_{j,2} = \sigma_{j,2}$.
4. A sends the signature $(\sigma'_{j,1}, \sigma'_{j,2})$ to B and the secret information $\{\sigma_{j,2}^{t'_i m_{j,i}} | i = n - m + 1, \cdots, n\}$ to B.

In the challenge phase, if B is successful in obtaining a valid message signature pair $(M^{*\prime}, \sigma^{*\prime})$, then A obtains a valid signature tuple as follows:

1. B sends A a valid message-signature tuple $(M^{*\prime} = m_1^{*\prime} \cdots m_n^{*\prime}, \sigma^{*\prime} = {\sigma^{*}}'_1, \sigma_2^{*\prime})$. Clearly $\forall j \in \{1, \cdots, q_s\}$ $\exists i \notin \{n - m + 1, \cdots, n\} : m_{j,i} \neq m_i^{*\prime}$.
2. A sets $M^* = m_1^* \cdots m_{n-m}^*$, where $m_i^* = m_i^{*\prime}$ for all $i = 1, \cdots, n - m$. A sets
$$\sigma_1^* = \frac{\sigma_1^{*\prime}}{\prod_{i=n-m+1}^{n} \sigma_2^{t'_i {m^*_{j,i}}'}}, \sigma_2^* = \sigma_2^{*\prime}$$
3. A sends C a valid message-signature pair $(M^*, \sigma^* = (\sigma_1^*, \sigma_2^*))$. Clearly, it follows that $\forall j \in \{1, \cdots, q_s\}$ $\exists i \in \{1, \cdots, n - m\} : m_{j,i} \neq m_i^*$.

It is easy to see that if B's signature tuple verifies, then A's signature tuple verifies as well. Hence the advantage of A winning the game $\mathsf{Exp}_{\text{unf}}$, $\epsilon_a \geq \epsilon_b$, where ϵ_b is the advantage of B in winning the immutability game $\mathsf{Exp}_{\text{imm}}$.

From theorem-1, the advantage of any probabilistic polynomial time algorithm in winning the unforgeability game $\mathsf{Exp}_{\mathrm{unf}}$ is negligible under the CDH assumption. Applying lemma-1, clearly the advantage of any probabilistic polynomial time algorithm in winning the immutability game $\mathsf{Exp}_{\mathrm{imm}}$ is also negligible under the CDH assumption. This proves theorem-2.

4.4 Extensions

Control on Sanitized message. The signer can control what the sanitizer assigns to set of bits in $I_{\mathcal{S}}$ which the latter has control over. For example, this is applicable to scenarios where the sanitized portion can take only certain values. Consider the case where the sanitizer is only allowed to change the message from m to m'. Then, this is done by the signer revealing the value $U = \prod_{i \in I_{\mathcal{S}} | m'_i \neq m_i} u_i^{(m'_i - m_i)}$, instead of the individual $u_i^r | i \in I_{\mathcal{S}}$ values to the sanitizer. Then the sanitizer obtains a signature σ' from a signature $\sigma = (\sigma_1, \sigma_2)$ on m using $\sigma' = (U\sigma_1, \sigma_2)$.

Multiple sanitizations. The protocol can be readily extended to multiple sanitizers by providing appropriate secret information to each sanitizer. If sanitizer S_j has permission to sanitize bits in $I_{\mathcal{S}}^j$, the signer must provide the values $u_i^r | i \in I_{\mathcal{S}}^j$, in a secure fashion to this sanitizer.

4.5 Salient Features

In this scheme, the signer has a high degree of control over the bits that the sanitizer is permitted to change, as well as the possible ways in which the sanitizer may change these bits. However, as a trade-off the signer incurs the additional overhead of securely transmitting secret information on a per-message basis to the sanitizer. This is inevitable in protocols where the positions where the sanitizer is permitted to sanitize the message are not known a priori, as this information needs to be relayed to the sanitizer on a per-message basis. However, this may be avoided in cases where the set of bits that the sanitizer is expected to modify are fixed during the key generation phase. This is common in cases where the message to be sanitized has a standard format (for e.g., databases, forms, etc.). In scheme-2, we discuss one such protocol which achieves strong transparency under the CDH assumption in the standard model. We discuss scheme-2 in the following section.

5 Scheme 2

5.1 Outline

This scheme provides a strong transparent sanitizable signature protocol where the sanitizer is provided private information in the key generation phase. Using this, he may modify certain fixed set of positions of the signature. The signature protocol is based on the Water's signature scheme [10]. The indices which are permitted to be sanitized are fixed at the time of key generation. The sanitizer may replace those portions of the message by an appropriate message of his choice. In this section, we describe this scheme, provide security proofs based on the CDH assumption.

5.2 Scheme Description

KeyGen. Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be groups of prime order p. Given a pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. Let $g \in \mathbb{G}_1$ and $g_2, u', u \in \mathbb{G}_2$. Let $\alpha_1, \cdots, \alpha_n \in \mathbb{Z}_p^*$. Compute $u_1 = u^{\alpha_1}, \cdots, u_n = u^{\alpha_n}$.

Public parameters: $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, p, g_2, u', u, u_1, \cdots, u_n$ Public key of the signer is $g_1 = g^{\alpha}$, where $\alpha \in \mathbb{Z}_p^*$.

Private parameters: Private key of the signer is $\alpha \in_R \mathbb{Z}_p^*$.
Private key of a sanitizer j with access to modify bit positions $I_{\mathcal{S}_j} \subseteq \{1, \cdots, n\}$ is $\alpha_i \ \forall i \in I_{\mathcal{S}_j}$.

Sign. Let m be the n-bit message $m_1 m_2 \cdots m_n \in \{0,1\}^n$. Signer randomly picks $r \in \mathbb{Z}_p^*$ and outputs the following values $(\sigma_1, \sigma_2, \sigma_3)$:

$$(\sigma_1 = g_2^{\alpha}(u' \prod_{i=1}^{n} u_i^{m_i})^r, \sigma_2 = g^r, \sigma_3 = u^r)$$

Sanitize. The sanitizer $\mathcal{S}_j$ obtains the values $(\sigma_1, \sigma_2, \sigma_3)$, from the signer. It runs the verification protocol to check if the signature is valid. It then computes the values $u_i^r \leftarrow \sigma_3^{\alpha_i} \ \forall i \in I_{\mathcal{S}_j}$. Let m' be the message whose signature is sought, which differs from m at positions $I \subseteq I_{\mathcal{S}_j}$. Define $I_1 = \{i \in I : m_i = 0, m'_i = 1\}$, $I_2 = \{i \in I : m_i = 1, m'_i = 0\}$. The sanitizer chooses $\tilde{r} \in_R \mathbb{Z}_p^*$. Then the required sanitized signature is:

$$(\sigma'_1 = \sigma_1 \frac{\prod_{i \in I_1} u_i^r}{\prod_{i \in I_2} u_i^r} u'^{\tilde{r}} \prod_{i=1}^{n} u_i^{m'_i \tilde{r}}, \sigma'_2 = \sigma_2 g^{\tilde{r}}, \sigma'_3 = \sigma_3 u^{\tilde{r}})$$

Verify. Receives the tuple: $(\sigma_1, \sigma_2, \sigma_3)$ on a message m,
Verifier checks if the following relations hold from public parameters:

$$e(g, \sigma_1) \stackrel{?}{=} e(g_1, g_2) e(\sigma_2, u' \prod_{i=1}^{n} u_i^{m_i})$$

$$e(g, \sigma_3) \stackrel{?}{=} e(\sigma_2, u)$$

Note that the verification protocol is same for a sanitized and non-sanitized message.

5.3 Security

Correctness. To show correctness, we need to show that any valid normal signature, as well as sanitized signature verifies successfully.

Verification: The signature σ, on a given message m is given by the three-tuple $(\sigma_1 = g_2^\alpha(u' \prod_{i=1}^n u_i^{m_i})^r \ , \sigma_2 = g^r, \sigma_3 = u^r)$. If valid, then clearly:

$$\begin{aligned}
&e(g_1, g_2)e(\sigma_2, u' \textstyle\prod_{i=1}^n u_i^{m_i})\\
&= e(g^a, g_2)e(g^r, u' \textstyle\prod_{i=1}^n u_i^{m_i})\\
&= e(g, g_2^a)e(g, (u' \textstyle\prod_{i=1}^n u_i^{m_i})^r)\\
&= e(g, g_2^a(u' \textstyle\prod_{i=1}^n u_i^{m_i})^r)\\
&= e(g, \sigma_1)
\end{aligned}$$

Also we note that:

$$\begin{aligned}
e(g, \sigma_3) &= e(g, u^r)\\
&= e(g^r, u)\\
&= e(\sigma_2, u)
\end{aligned}$$

Hence, a valid signature satisfies the verification equations.

Sanitization: A sanitized signature is obtained as:

$$(\sigma_1' = \sigma_1 \frac{\prod_{i \in I_1} u_i^r}{\prod_{i \in I_2} u_i^r} u'^{\tilde{r}} \prod_{i=1}^n u_i^{m_i' \tilde{r}}, \sigma_2' = \sigma_2 g^{\tilde{r}}, \sigma_3' = \sigma_3 u^{\tilde{r}})$$

where $I_1 = \{i \in I_{\mathcal{S}} : m_i = 0, m_i' = 1\}$, $I_2 = \{i \in I_{\mathcal{S}} : m_i = 1, m_i' = 0\}$.
We note that $m_i' - m_i$ is 1 when $i \in I_1$, -1 when $i \in I_2$,and 0, otherwise. Hence, we can see that:

$$\begin{aligned}
\sigma_1' &= \sigma_1 u'^{\tilde{r}} \textstyle\prod_{i \in I_1} u_i^r / \prod_{i \in I_2} u_i^r \prod_{i=1}^n u_i^{m_i' \tilde{r}}\\
&= \sigma_1 u'^{\tilde{r}} \textstyle\prod_{i=1}^n u_i^{r(m_i' - m_i)} \prod_{i=1}^n u_i^{m_i' \tilde{r}}\\
&= g_2^\alpha u'^r u'^{\tilde{r}} \textstyle\prod_{i=1}^n u_i^{r(m_i)} \prod_{i=1}^n u_i^{r(m_i' - m_i)} \prod_{i=1}^n u_i^{m_i' \tilde{r}}\\
&= g_2^\alpha u'^{(r+\tilde{r})} \textstyle\prod_{i=1}^n u_i^{(r+\tilde{r})(m_i')}
\end{aligned}$$

The sanitized signature is of the form $(\sigma_1' = g_2^\alpha(u' \prod_{i=1}^n u_i^{m_i'})^{(r+\tilde{r})}, \sigma_2 = g^{(r+\tilde{r})}, \sigma_3 = u^{(r+\tilde{r})})$, whose distribution is identical to a regular signature on m' by the signer. Hence, a sanitized signature also satisfies the verification equations.

Unforgeability. We prove the following theorem about unforgeability.

Theorem 3. *The proposed sanitizable signature scheme in scheme-2 is (ϵ, q_s)-unforgeable under the ϵ'-CDH assumption where $\epsilon \leq (8q_s^2(n+1)^2 + 2)\epsilon' + 2/p$.*

Proof: Assume there is a (ϵ, t, q_s)-adversary A exists. We shall formulate another probabilistic polynomial time (PPT) algorithm B that uses A to solve the CDH problem with probability at least ϵ' and in time at most t'. B is given a problem instance as follow: Given a group $\mathbb{G}$, a generator $g \in \mathbb{G}$, two elements $g^a, g^b \in \mathbb{G}$. It is asked to output another element $g^{ab} \in G$. In order to use A to solve for the problem, B needs to simulates a challenger and the signing oracle for A. B does it in the following way.

Setup Phase. Let l = $2q_s$. B randomly selects an integer k such that $0 \leq k \leq n$. Also assume that $l(n+1) < p$, for the given values of q_s and n. It randomly selects:

1. $x', v \in_R \mathbb{Z}_l; y' \in_R \mathbb{Z}_p$
2. $\hat{x}_i \in_R \mathbb{Z}_l$, Let $\hat{X} = \{\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_n\}$.
3. $\hat{y}_i \in_R \mathbb{Z}_p$, Let $\hat{Y} = \{\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_n\}$.

We further define the following functions for binary string $M = (m_1, m_2, \cdots, m_n)$, where $m_i \in \{0, 1\}$, $1 \leq i \leq n$, as follows:

$$F(M) = x' + \sum_{i=1}^{n} \hat{x}_i m_i - lk$$

$$J(M) = y' + \sum_{i=1}^{n} \hat{y}_i m_i$$

B constructs a set of public parameters as follows:

$$g_2 = g^b, u' = g_2^{-lk+x'} g^{y'}, u = g^v, u_i = g_2^{\hat{x}_i} g^{\hat{y}_i}, i = 1, \cdots, n$$

We have the following equation:

$$u' \prod_{i=1}^{n} u_i^{m_i} = g_2^{F(M)} g^{J(M)}$$

All the above public parameters and public key $g_1 = g^a$ are passed to A.

Simulation Phase. B simulates the signing oracle as follow. Upon receiving the j^{th} query for a document M_j, although B does not know the secret key, it can still construct the signature by assuming $F(M_j) \neq 0 \bmod p$. It randomly chooses $r_j \in_R \mathbb{Z}_p$ and computes the signature as

$$\sigma_{1,j} = g_1^{-J(M_j)/F(M_j)} (g_2^{F(M_j)} g^{J(M_j)})^{r_j}, \sigma_{2,j} = g_1^{-1/F(M_j)} g^{r_j}$$

$$\sigma_{3,j} = g_1^{-v/F(M_j)} g^{v r_j}$$

By letting $\hat{r}_j = r_j - a/F(M_j)$, it can be verified that $(\sigma_{1,j}, \sigma_{2,j}, \sigma_{3,j})$ is a valid signature on M_j as shown below:

$$\begin{aligned}\sigma_{1,j} &= g_1^{-\frac{J(M_j)}{F(M_j)}} (g_2^{F(M_j)} g^{J(M_j)})^{r_j} \\ &= g^{-a\frac{J(M_j)}{F(M_j)}} (g_2^{F(M_j)} g^{J(M_j)})^{\frac{a}{F(M_j)}} (g_2^{F(M_j)} \ g^{J(M_j)})^{-\frac{a}{F(M_j)}} \ (g_2^{F(M_j)} g^{J(M_j)})^{r_j} \\ &= g_2^a (g_2^{F(M_j)} g^{J(M_j)})^{\hat{r}_j} \\ \sigma_{2,j} &= g_1^{-1/F(M_j)} g^{r_j} \\ &= g^{r_j - a/F(M_j)} \\ &= g^{\hat{r}_j} \\ \sigma_{3,j} &= g_1^{-v/F(M_j)} g^{vr_j} \\ &= g^{v(r_j - a/F(M_j))} \\ &= u^{\hat{r}_j}\end{aligned}$$

If $F(M_j) = 0 \bmod p$, since the above computation cannot be performed (division by 0), the simulator aborts. To make it simple, the simulator will abort if $F(M_j) = 0 \bmod l$. The equivalence can be observed as follow. From the assumption that $l(n+1) < p$, it implies $0 \leq lk < p$ and $0 \leq x' + \sum_{i=1}^{n} \hat{x}_i m_i < p$ (as $x' < l, \hat{x}_i < l$). We have $-p < F(M_j) < p$ which implies if $F(M_j) = 0 \bmod p$ then $F(M_j) = 0 \bmod l$. Hence, $F(M_j) \neq 0 \bmod l$ implies $F(M_j) \neq 0 \bmod p$. Thus the former condition will be sufficient to ensure that a signature can be computed without aborting.

Challenge Phase. If B does not abort, A will return a document $M^* = m_1^* \cdots m_n^*$ with a forged signature $\sigma^* = (\sigma_1^*, \sigma_2^*, \sigma_3^*)$. The algorithm B aborts if $x' + \sum_{i|m_i^*=1} \hat{x}_i - lk \neq 0 \bmod l$. From the verification equation, we can write:

$$\begin{aligned}\sigma_1^* &= g_2^a (g_2^{F(M^*)} g^{J(M^*)})^{r^*} \\ &= g_2^a (g_2^{(x' + \sum_{i|m_i^*=1} \hat{x}_i - lk) r^*} g^{(y' + \sum_{i|m_i^*=1} \hat{y}_i) r^*} \\ &= g_2^a g^{(y' + \sum_{i|m_i^*=1} \hat{y}_i) r^*}\end{aligned}$$

Hence the algorithm successfully computes the solution to the CDH problem:

$$Z = \sigma_1^* {\sigma_2^*}^{-y' - \sum_{i|m_i^*=1} \hat{y}_i} = g_2^a = g^{ab}$$

□

Probability Analysis. The probability that the simulation does not abort is characterized by the events A_j, A^* where:

1. A_j is the event that $F(M_j) \neq 0 \bmod l$ where $j = 1, \cdots, q_s$.
2. A^* is the event that $x' + \sum_{i|m_i^*=1} \hat{x}_i - lk = 0 \bmod p$.

The probability that B does not abort:

$$\Pr[\text{not abort}] \geq \Pr[\bigwedge_{j=1}^{q_s} A_j \wedge A^*]$$

As the adversary can at most make B abort by randomly choosing M^*, we have $\Pr[A^*] = \frac{1}{l(n+1)}$. Also noting that A_j is independent of A^* we have:

$$\begin{aligned}\Pr[\text{not abort}] &\geq \Pr[\textstyle\bigwedge_{j=1}^{q_s} A_j \wedge A^*] \\ &\geq \Pr[A^*]\Pr[\textstyle\bigwedge_{j=1}^{q_s} A_j | A^*] \\ &\geq \tfrac{1}{(l(n+1))^2}(1 - \textstyle\sum_{j=1}^{q_s} \Pr[\neg A_j | A^*]) \\ &\geq \tfrac{1}{8(n+1)^2 q_s^2}\end{aligned}$$

Indistinguishability. As shown in the correctness section, a valid signature σ_s produced by a signer on a message m' has a distribution identical to a valid sanitization of a message m_1 to result in message m' and signature σ'_a produced by the sanitizer. Similarly, the distribution is also identical to a valid sanitization of another message m_2 to result in message m' and signature σ'_b, produced by the sanitizer. Hence the signatures σ'_a and σ'_b are indistinguishable as their distributions are identical.

Immutability. We prove the following theorem to show immutability.

Theorem 4. The proposed sanitizable signature scheme in scheme-2 is ϵ-immutable under the ϵ'-CDH assumption, where there exists constant $l : \epsilon < l\epsilon'$.

Proof: We prove that the sanitizer cannot modify any bit other than bits at positions $I_{\mathcal{S}} \subseteq \{1, \cdots, n\}$ for which the values $\{u_i^r : i \in I_{\mathcal{S}}\}$ are known to the sanitizer. We will prove the following lemma on immutability in order to prove the above theorem.

Lemma 2. For any randomized polynomial time algorithm algorithm B with an advantage ϵ_b in the immutability game $\mathsf{Exp}_{\text{imm}}$ on a message of length n with access to sanitize m bits at positions $I_{\mathcal{S}}$, there exists a randomized polynomial time algorithm A with an advantage $\epsilon_a \geq \epsilon_b$ in the unforgeability game $\mathsf{Exp}_{\text{unf}}$ on a message of length $n - m$.

Proof: Assume that there exists a randomized polynomial time algorithm B which plays the immutability game $\mathsf{Exp}_{\text{imm}}$ with advantage ϵ_b with access to sanitize m bits at positions $I_{\mathcal{S}}$. Consider a randomized polynomial time algorithm A which plays the unforgeability game $\mathsf{Exp}_{\text{unf}}$ on messages of length $n-m$. Then we show that the algorithm A can simulate the challenger interacting with algorithm B, and thereby obtain an advantage $\epsilon_a \geq \epsilon_b$ in $\mathsf{Exp}_{\text{unf}}$. In the setup phase, A interacts with B and the challenger in $\mathsf{Exp}_{\text{unf}}$, denoted by C, as follows:

1. B provides A the set of bit positions where sanitization is allowed, $I_{\mathcal{S}}$. In general, we have $I_{\mathcal{S}} \subseteq \{1, \cdots, n\}$. However for the ease of exposition we assume $I_{\mathcal{S}} = \{n - m + 1, \cdots, n\}$, where $m = |I_{\mathcal{S}}|$. Note that the argument can be easily extended for the general form of $I_{\mathcal{S}}$.
2. C provides A the public parameters $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, p, g_2, u, u', u_1, \cdots, u_{n-m}$.

3. A chooses $t_i \in_R \mathbb{Z}_p^*$, $i = n-m+1, \cdots, n$. A sets $u'_i = u^{t'_i}$, for $i = n-m+1, \cdots, n$
4. A provides B the public parameters $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g, p, g_2, u, u', u_1, \cdots, u_{n-m}, u'_{n-m+1} \cdots, u'_n$.
5. A provides B secret information $SI = \{t_i | i = n-m+1, \cdots, n\}$.

In the simulation phase, for every message M_j, $j = 1, \cdots, q_s$, requested by B, A interacts with B and C as follows:

1. B requests signature for a message $M_j = m_{j,1} \cdots m_{j,n}$ from A.
2. A requests a signature for message $M_j = m_{j,1} \cdots m_{j,n-m}$ from C.
3. A obtains $(\sigma_{j,1}, \sigma_{j,2}, \sigma_{j,3})$ from C, and sets $\sigma'_{j,1} = \sigma_{j,1} \prod_{i=n-m+1}^{n} \sigma_{j,3}^{t'_i m_{j,i}}$, $\sigma'_{j,2} = \sigma_{j,2}$ and $\sigma'_{j,3} = \sigma_{j,3}$.
4. A sends the signature $(\sigma'_{j,1}, \sigma'_{j,2}, \sigma'_{j,3})$ to B.

In the challenge phase, if B is successful in obtaining a valid message signature pair $(M^{*\prime}, \sigma^{*\prime})$, then A obtains a valid signature tuple as follows:

1. B sends A a valid message-signature tuple $(M^{*\prime} = m_1^{*\prime} \cdots m_n^{*\prime}, \sigma^{*\prime} = {\sigma^*}'_1, \sigma_2^{*\prime}, \sigma_3^{*\prime})$. Clearly $\forall j \in \{1, \cdots, q_s\}\ \exists i \notin \{n-m+1, \cdots, n\} : m_{j,i} \neq m_i^{*\prime}$.
2. A sets $M^* = m_1^* \cdots m_{n-m}^*$, where $m_i^* = m_i^{*\prime}$ for all $i = 1, \cdots, n-m$. A sets

$$\sigma_1^* = \frac{\sigma_1^{*\prime}}{\prod_{i=n-m+1}^{n} \sigma_3^{t'_i m_{j,i}^{*\prime}}}, \sigma_2^* = \sigma_2^{*\prime}, \sigma_3^* = \sigma_3^{*\prime}$$

3. A sends C a valid message-signature pair $(M^*, \sigma^* = (\sigma_1^*, \sigma_2^*, \sigma_3^*))$. Clearly, it follows that $\forall j \in \{1, \cdots, q_s\}\ \exists i \in \{1, \cdots, n-m\} : m_{j,i} \neq m_i^*$.

It is easy to see that if B's signature tuple verifies, then A's signature tuple verifies as well. Hence the advantage of A winning the game $\mathsf{Exp}_{\text{unf}}$, $\epsilon_a \geq \epsilon_b$, where ϵ_b is the advantage of B in winning the immutability game $\mathsf{Exp}_{\text{imm}}$.

From theorem-3, the advantage of any probabilistic polynomial time algorithm in winning the unforgeability game $\mathsf{Exp}_{\text{unf}}$ is negligible under the CDH assumption. Applying lemma-2, clearly the advantage of any probabilistic polynomial time algorithm in winning the immutability game $\mathsf{Exp}_{\text{imm}}$ is also negligible under the CDH assumption. This proves theorem-4. □

6 Comparison

We compare our scheme against previous schemes on sanitizable signatures.

Scheme	Transparency	Security	Model
[1]	No Transparency	RSA	ROM
[13]	No Transparency	underlying signature	standard
[14]	No Transparency	underlying signature	standard
[5]	No Transparency	underlying signature and commitment	standard
[6]	No Transparency	co-GDH	ROM
[8]	No Transparency	co-GDH	ROM
[7]	No Transparency	strong RSA	standard
[15]	No Transparency	underlying signature commitment and pseudo random generator	standard
[9]	No Transparency	CDH + XDH	standard
[2]	Weak Transparency	underlying signature and chameleon hash	standard
[4]	Weak Transparency	CDH	ROM
[3]	Strong Transparency	-	-
Our Scheme	Strong Transparency	CDH	standard

7 Conclusion and Open Problems

In this paper, we proposed the first provably secure sanitizable signature protocol having strong transparency property under standard model. These signatures are of constant length, and shorter than most other protocols. In earlier schemes, such as [3] which claim strong transparency, either there is no formal proof provided or the proof is under the random oracle model. An interesting open problem is to devise a protocol which can achieve strong transparency without dividing the message into bits or blocks. The problem of using more traditional techniques such as RSA, rather than pairings to provide more efficient sanitizable signatures with strong transparency is open. Accountability is a property of sanitizable signatures by which the signer can prove that a particular signature is his, and not by the sanitizer. In our schemes, accountability is not provided as this compromises unconditional indistinguishabilty. However, an interesting open problem would be to formulate a sanitizable signature scheme with strong transparency that offers polynomial time indistinguishability as well as accountability.

References

1. Bull, L., Stañski, P., Squire, D.: Content extraction signatures using xml digital signatures and custom transforms on-demand. In: WWW. (2003) 170–177
2. Ateniese, G., Chou, D.H., de Medeiros, B., Tsudik, G.: Sanitizable signatures. In: ESORICS. (2005) 159–177
3. Klonowski, M., Lauks, A.: Extended sanitizable signatures. In: ICISC. (2006) 343–355

4. Miyazaki, K., Hanaoka, G., Imai, H.: Digitally signed document sanitizing scheme based on bilinear maps. In: ASIACCS. (2006) 343–354
5. Miyazaki, K., Iwamura, M., Matsumoto, T., Sasaki, R., Yoshiura, H., Tezuka, S., Imai, H.: Digitally signed document sanitizing scheme with disclosure condition control. IEICE Transactions **88-A**(1) (2005) 239–246
6. Suzuki, M., Isshiki, T., Tanaka, K.: Sanitizable signature with secret information. In: Research reports on mathematical and computing sciences, Springer-Verlag (2005) 114–127
7. Chang, E.C., Lim, C.L., Xu, J.: Short redactable signatures using random trees. In: CT-RSA. (2009) 133–147
8. Izu, T., Kunihiro, N., Ohta, K., Takenaka, M., Yoshioka, T.: A sanitizable signature scheme with aggregation. In: ISPEC. (2007) 51–64
9. Yuen, T.H., Susilo, W., Liu, J.K., Mu, Y.: Sanitizable signatures revisited. In: CANS '08: Proceedings of the 7th International Conference on Cryptology and Network Security, Berlin, Heidelberg, Springer-Verlag (2008) 80–97
10. Waters, B.: Efficient identity-based encryption without random oracles. In: EUROCRYPT. (2005) 114–127
11. Klonowski, M., Lauks, A.: Extended sanitizable signatures. In: ICISC. (2006) 343–355
12. Brzuska, C., Fischlin, M., Freudenreich, T., Lehmann, A., Page, M., Schelbert, J., Schröder, D., Volk, F.: Security of sanitizable signatures revisited. In: Irvine: Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography, Berlin, Heidelberg, Springer-Verlag (2009) 317–336
13. Johnson, R., Molnar, D., Song, D.X., Wagner, D.: Homomorphic signature schemes. In: CT-RSA '02: Proceedings of the The Cryptographer's Track at the RSA Conference on Topics in Cryptology, London, UK, Springer-Verlag (2002) 244–262
14. Miyazaki Kunihiko, Susaki Seiichi, I.M.: Digital document sanitizing problem. In: CIEIC Technical Report (Institute of Electronics, Information and Communication Engineers). (2003) 61–67
15. Haber, S., Hatano, Y., Honda, Y., Horne, W., Miyazaki, K., Sander, T., Tezoku, S., Yao, D.: Efficient signature schemes supporting redaction, pseudonymization, and data deidentification. In: ASIACCS '08: Proceedings of the 2008 ACM symposium on Information, computer and communications security, New York, NY, USA, ACM (2008) 353–362
16. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: EUROCRYPT. (2008) 415–432
17. Izu, T., Kanaya, N., Takenaka, M., Yoshioka, T.: Piats: A partially sanitizable signature scheme. In: ICICS. (2005) 72–83
18. Izu, T., Kunihiro, N., Ohta, K., Sano, M., Takenaka, M.: Sanitizable and deletable signature. (2009) 130–144
19. Boneh, D., Boyen, X.: Short signatures without random oracles. In: EUROCRYPT. (2004) 56–73
20. Lysyanskaya, A., Micali, S., Reyzin, L., Shacham, H.: Sequential aggregate signatures from trapdoor permutations. In: EUROCRYPT. (2004) 74–90
21. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: CCS '93: Proceedings of the 1st ACM conference on Computer and communications security, New York, NY, USA, ACM (1993) 62–73
22. Gentry, C.: Practical identity-based encryption without random oracles. In: EUROCRYPT. (2006) 445–464