Key Protected Deputy Signature Scheme against the Deputy Signing Key Exposure

Jianhong Chen
Faculty of Computer and Software Engineering, Huaiyin Institute of Technology, Huaian 223003, Jiangsu, China

Kun Yu
Faculty of Computer and Software Engineering, Huaiyin Institute of Technology, Huaian 223003, Jiangsu, China

Wenhao Wang
Faculty of Computer and Software Engineering, Huaiyin Institute of Technology, Huaian 223003, Jiangsu, China

Received: June 25, 2021. Revised: July 21, 2021. Accepted: July 23, 2021. Published: July 30, 2021

Abstract—Key exposure is very harmful to a cryptographic system. To decrease the loss from the deputy signing key vulnerability in identity-based proxy signature systems, we propose the method of key protected deputy signature (IBKPDS) using the method of parallel key insulation. The proposed IBKPDS is based on identities and is shown to be secure with the cryptographic proof. In the proof, there is no random oracle. In an IBKPDS cryp-system, a user stores his short-lived deputy signing key by himself and saves two long-lived keys in two heavily guarded boxes respectively. The derived IBKPDS cryptographic system is heavily key-separated. A thief who wants to obtain crucial information can not corrupt the IBKPDS when he get only one long-lived key. In addition, the user can change the short-lived deputy signing keys frequently at low risk.

Keywords—Identity, key-separated, long-lived key, deputy signature, low risk.

I. INTRODUCTION

Loss of secret keys is dangerous to the security of the public key cryptosystem [1, 2]. It is import to rescind users for PKI or Identity-based setting in the settings of the loss of their private keys. When lots of users renew their proprietary keys at short intervals, communication and computation overhead will make the PKI authority unbearable. Several complementary approaches are to evolve secret keys in case where secrets are in danger [3]. One of the key evolving notions is forward security. In the model of forward security, the opponent is unable to compromise secret keys associated with prior time periods. Another key evolving notions is key insulation mechanism. In the case of key insulation, there are two classes of private keys, i.e. long-lives keys and short-lives keys. A user stores his long-lives keys by himself and saves short-lives keys in a heavily guarded box. The notion of parallel key insulation complements the notion of key insulation. In the setting of parallel key insulation, the user uses different independent helper keys in key evolving operations. In a Parallel key-separated crypto-system, a user stores his long-lives keys by himself and saves two short-lived keys in two physically-secure computationally-limited devices respectively.

In cryptography digital signature [4, 5] employs the advanced mathematical technique to check the authenticity of digital messages. In the system of proxy signature [6], there are a foremost signer and a deputy signer. In the event that the foremost signer appoints a deputy signer or an agent to sign papers, the foremost signer should submit a letter of authorization or a warrant to the agent [7]. In a blind signature system [8], the verifier verifies against the original, unblinded message while the signer signs a disguised (blinded) message. Blind signatures are very useful when the signer and message author are different parties. Deputy sightless signature [9, 10] enjoys the advantage of the proxy signature and the merit of sightless signature. In an application system of deputy sightless signature, the deputy signer produces a blind sightless for the foremost signer. The deputy re-signature [11, 12] is similar to the deputy signature. In a deputy re-signature system, Jeff is not a fully trusted agent and works as an interpreter between Mary and Tom. To interpret, Jeff changes an old signature into a new signature. The old signature is Mary’s signature. The new signature is Tom’s signature. Jeff can neither sign in the name of Mary nor sign in the name of Tom because Jeff doesn’t know their personal signing keys. In a deputy signature system, Jeff work as a trusted deputy of Mary and can sign arbitrary messages in the name of Mary. Proxy signcryption [13] integrates the functions of signcryption and proxy signature.

Against the deputy key vulnerability in identity-based
deputy signature systems, we propose a construction method of a key protected deputy signature based on identity (IBKPDS) scheme using the method of parallel key insulation. The proposed IBKPPIS method is shown to be secure with the cryptographic proof.

II. MODEL OF IBKPPIS

A. Our proposed IBKPPIS model

In the system of the IBKPDS, there are some steps as below:

- Setup: The public agency computes the main privy constant and some overt constants. Then the public agency gives the main privy constant and some overt constants to the people involved.
- Ext: The Private Key Generator computes the deputy signer’s first short-lived secret key, two long-lived refreshing keys of the deputy signer and the foremost signer’s long-lived secret key.
- DUUpdateLongLived: The deputy signer computes his short-lived refreshing key for time period \( t \) using his short-lived device.
- DUUpdateUser: The deputy signer computes his short-lived secret key for time period \( t \) using his short-lived device.
- DelegGen: The foremost signer computes the deputy document.
- DelegVerify: The verifier of the proxy document checks whether the deputy document is valid.
- GenPSig: The deputy signer computes the short-lived deputy signature key using his short-lived secret key and the valid deputy document above.
- PSsignatureVerify: The deputy signer computes the deputy signature for the designated length of time using his short-term deputy signing key and his deputy document.
- PVerification: The verifier of proxy document verifier checks whether the deputy signature for the designated length of time is valid.

B. the Isolation of the Short-lived Deputy Signing Keys

We use a match between a defier and an opponent to simulate the scenario in which the short-lived deputy signing Keys is isolated against the corruption of the opponent. The match in which the defier competes with opponent is shown as below.

- Setup. The defier computes the main privy constant and some overt constants. The main privy constant and some overt constants are kept by the defier and the opponent respectively.
- Queries. The opponent asks the defier many times for the answers of some questions.

1. Questions about Ext. The defier computes the foremost secret key, the earliest deputy signing key and helper keys of the deputy signer. The defier computes the secret key of the foremost signer. The overt constants are stored by the opponent. The opponent keeps the foremost signing key, the earliest deputy signing key and helper keys.

2. Questions about the deputy document. The defier computes the deputy document. The opponent keeps the deputy document.

3. Questions about the short-lived deputy signing key. The defier computes the short-lived deputy signing key. The opponent keeps the short-lived deputy signing key.

4. Questions about the deputy signature. The defier computes the deputy signature. The opponent keeps the deputy signature.

5. Questions about Deputy short-lived keys of the deputy signer. The Isolation of the short-lived deputy signing keys.
Counterfeit. The opponent counterfeits the deputy document and the signature. If the statements are valid, the opponent defeats the defier in the match above.

① The opponent makes a counterfeit tuple, \( (\Psi^*, \hat{z}, \hat{\phi}, p^*) \) and \( (\Psi'^*, \hat{z}') \) is the deputy document of time period \( \hat{z}' \) if the statements below are valid: \( \mathcal{D}(\hat{z}, \hat{z}'') = 1 \); the attacked identity of the foremost signer was not involved in the questions about Ext; \( (p^*, \hat{z}, \Psi^*) \) was not involved in the questions about the deputy document; \( (p^*, \hat{z}', \Psi'^*) \) was not involved in the questions about the short-lived deputy signing keys.

② The opponent makes a counterfeit tuple \( (\Psi^*, \hat{z}, \hat{\phi}, m^*) \) when the statements below are valid: \( \mathcal{D}(\hat{z}, \hat{z}'') = 1 \); \( p^* \) and \( p^* \) are the foremost signer and the deputy signer respectively written in \( \Psi^* \); \( p^* \) was not involved in the questions about Ext; \( (p^*, \hat{z}, \Psi^*) \) was not involved in the questions about the short-lived deputy signing keys; \( (\hat{z}, \Psi'^*, m^*) \) was not involved in the questions about the deputy signature.

### III. OUR PROPOSED IBKPDDS

We propose a scheme of the identity-based key protected deputy signature (IBKPDDS). Our method is to combine the scheme of Feng Cao et al.'s IBPS (proxy signature based on identity) with the scheme of Jian Weng et al.'s IBKPSS (parallel key-separated signature based on identity) which introduced the method of parallel key-isolation into IBS (signature based on identity).

We show IBS (signature based on identity) as below.

(1) Su: \( \Omega \) is a cyclic group of multiplication. The order of \( \Omega \) is a prime number. \( \Omega \times \Omega \) is a cyclic group of multiplication. The order of \( \Omega \) is a prime number. \( \Omega \times \Omega \to \Omega \) is a bilinear pairing \( \mathcal{P} \) of which the generator is \( \mathcal{G} \). We pick a random integer \( a, \eta_1 \) is \( \eta \) to the power of \( a \). We pick a random integer \( \eta_2 \) from \( \Omega \). We pick a random integer \( \mathcal{G}_1, \mathcal{G}_2 : \{0, 1\}^* \to \{0, 1\} \) is a function that is of hash and against collision. We pick a random integer \( \eta_2 \). \( \mathcal{K}_1 : \{0, 1\}^* \to \{0, 1\} \) is a function that is of hash and against collision. We pick a random integer \( \eta_3 \). \( \mathcal{K}_2 : \{0, 1\}^* \to \{0, 1\} \) is a function that is of hash and against collision. \( \mathcal{K}_3 \) is used to modify the length of the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of the message to the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of a document to the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of the identity to the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of a message to the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of a document to the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of the identity to the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of a message to the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of a document to the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of the identity to the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of a message to the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of a document to the user's defined length. \( \mathcal{K}_3 \) is used to modify the length of the identity to the user's defined length.

(2) Ext: The foremost signer's identity \( \phi \) is a string of many bits. The length of \( \phi \) is \( \mathcal{G} \). \( \phi \) is the \( \mathcal{G} \)th bit of \( \phi \). \( \mathcal{G} \) is equal to the set of the subscript \( \mathcal{G} \), in which \( \phi \) is equal to 1. The elements of \( \mathcal{G} \) are integers which are less than \( \mathcal{G} \) and bigger than 1. The public agency randomly chooses a positive integer \( \beta_2 \) and sets the foremost signer's privy signing key as \( \lambda_2^{\phi} = (\lambda_1, \lambda_2) = (\eta_2^{\phi}, \eta_2^{\phi'}) \).

The deputy signer's identity \( \phi \) is a string of many bits. The length of \( \phi \) is \( \mathcal{G} \). \( \phi \) is the \( \mathcal{G} \)th bit of \( \phi \). \( \mathcal{G} \) is equal to the set of the subscript \( \mathcal{G} \), in which \( \phi \) is equal to 1. The elements of \( \mathcal{G} \) are integers which are less than \( \mathcal{G} \) and bigger than 1. The public agency randomly chooses a positive integer \( \beta_2 \) and sets the deputy signer's privy signing key as \( \lambda_2^{\phi} = (\lambda_1, \lambda_2) = (\eta_2^{\phi}, \eta_2^{\phi'}) \).
(3) DelegGen: The authorization \( \psi \) is a string of many bits. The length of \( \psi \) is \( \sigma \Delta \). \( \psi_i \) is the \( i \)th bit of \( \psi \). \( \Psi_{\psi} \) is equal to the set of the subscript \( i \), in which \( \psi_i \) is equal to 1. The elements of \( \Psi_{\psi} \) are integers which are less than \( \sigma \Delta \) and bigger than 1. The signer randomly chooses a positive integer \( \beta_{\psi} \) and sets his overt deputy document as 

\[
S_{\psi} = (S_{\psi_1}, S_{\psi_2}, S_{\psi_3})
\]

\[
= (\lambda_{\psi_1}(\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i}), \lambda_{\psi_2}(\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i}), \eta_{\psi_3}(\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i}))
\]

\[
= (\eta_2^3(\sigma_1 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_1}, (\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_2}, \eta_{\psi_3}(\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_3})
\]

Then the foremost signer gives the deputy tuple \((\psi_i, S_{\psi})\) to the deputy signer.

(6) DelegVerify: The deputy signer who wants to check the validity of a deputy document \( S_{\psi} \) parses the deputy document as \((S_{\psi_1}, S_{\psi_2}, S_{\psi_3})\) at first. Then he checks the truth of the equation below.

\[
\hat{p}(S_{\psi_1}, \eta) = \hat{p}(\eta_i, \eta_j)^2 \hat{p}(a_1' \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i}, S_{\psi_2}) \hat{p}(a_3' \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i}, S_{\psi_3})
\]

(7) GenPSig: If the deputy signer accepts the deputy document \( S_{\psi} \), he randomly chooses a positive integer \( \beta_{\psi} \) and constructs his deputy signing key as

\[
tdsk = (tdsk_1, tdsk_2, tdsk_3, tdsk_4)
\]

\[
= (S_{\psi_1}, \lambda_{\psi_1}(\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i}), S_{\psi_2}, \lambda_{\psi_2}(\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i}), \eta_{\psi_3})
\]

\[
= (\eta_2^2(\sigma_1 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_1}, (\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_2}, \eta_{\psi_3}(\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_3})
\]

\[
= (\eta_2^3(\sigma_1 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_1}, (\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_2}, \eta_{\psi_3}(\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_3})
\]

\[
= (\eta_2^3(\sigma_1 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_1}, \eta_{\psi_3}(\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_2}, \eta_{\psi_3}(\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_3})
\]

in which \( \beta_{\psi} \) is equal to the sum of \( \beta_i' \) and \( \beta_i \).

(9) SIGNATURE: The user who wants to check the validity of a deputy signature \( S_{\psi} \) parses the deputy signature as \((S_{\psi_1}, S_{\psi_2}, S_{\psi_3}, S_{\psi_4})\) at first. Then he checks the truth of the equation below.

\[
\hat{p}(S_{\psi_1}, \eta) = \hat{p}(\eta_i, \eta_j)^2 \hat{p}(a_1' \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i}, S_{\psi_2}) \hat{p}(a_3' \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i}, S_{\psi_3}) \hat{p}(a_2' \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i}, S_{\psi_4})
\]

We show the proposed IBKPDS as below.

(1) Su: \( \Omega \) is a cyclic group of multiplication. The order of \( \Omega \) is a prime number. \( \Omega \) is a cyclic group of multiplication. The order of \( \Omega \) is a prime number. \( \Omega \times \Omega \rightarrow \Omega \) is a bilinear pairing \( p \) of which the generator is \( \eta \). We pick a random integer \( a \eta \) is to the power of \( a \). We pick a random integer \( \eta_2 \) from \( \Omega \). We pick a random integer \( \alpha_1 \). \( F_{\alpha_1} : \{0,1\}^* \rightarrow \{0,1\}^{m_2} \) is a function that is of hash and against collision. We pick a random integer \( \alpha_2 \). \( F_{\alpha_2} : \{0,1\}^* \rightarrow \{0,1\}^{m_2} \) is a function that is of hash and against collision. We pick a random integer \( \alpha_3 \). \( F_{\alpha_3} : \{0,1\}^* \rightarrow \{0,1\}^{m_3} \) is a function that is of hash and against collision. \( F_{\alpha_3} \) is used to modify the length of the identity to the user's defined length. \( F_{\alpha_3} \) is used to modify the length of a message to the user's defined length. \( F_{\alpha_3} \) is used to modify the length of a document to the user's defined length. We pick a random integer \( a_1 \). \( \eta_1 = \eta \). \( \eta_1 \alpha_1 \) and \( \alpha_2 \) and \( \alpha_3 \) are random integers from \( \Omega \). The vector \( \Phi \) is equal to \( (\phi) \). The length of \( \Phi \) is \( \alpha_1 \). The vector \( \Gamma \) is equal to \( (\gamma) \). The length of \( \Gamma \) is \( \alpha_2 \). The vector \( \Psi \) is equal to \( (\psi) \). The length of \( \Gamma \) is \( \alpha_3 \). \( \Omega \), \( \Omega \), \( \hat{p} \), \( \hat{p} \), \( \eta \), \( \eta_1 \), \( \alpha_1 \), \( \alpha_2 \), \( \alpha_3 \) are the overt constants. Then the public agency gives the main privy constant and some overt constants to the people involved.

(2) Ext: The foremost signer’s identity \( \phi \) is a string of many bits. The length of \( \phi \) is \( \alpha_1 \). \( \phi \) is the \( n \)th bit of \( \phi \). \( \phi_0 \) is equal to the set of the subscript \( i \), in which \( \phi_i \) is equal to 1. The elements of \( \Phi_0 \) are integers which are less than \( \alpha_1 \) and bigger than 1. The public agency randomly chooses a positive integer \( \beta_{\phi} \) and sets the foremost signer’s privy signing key as

\[
\lambda_{\phi} = \lambda_{\phi_1}, \lambda_{\phi_2} = (\eta_2^2(\sigma_1 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_1}, \eta_{\psi_3}(\sigma_3 \prod_{i = \psi_1}^{\psi_2} \phi_i^{\psi_i})^{\psi_3})
\]

The deputy signer’s identity \( \phi_0 \) is a string of many bits. The length of \( \phi_0 \) is \( \alpha_1 \). \( \phi_0 \) is the \( n \)th bit of \( \phi_0 \). \( \phi_0 \) is equal to the set of the subscript \( i \), in which \( \phi_0 \) is equal to 1. The elements of \( \Phi_0 \) are integers which are less than \( \alpha_1 \) and bigger than 1. The public agency randomly chooses a positive integer \( \beta_{\phi_0} \) and sets the deputy signer’s short-lived privy signing key for time slot 0 as

\[
\lambda_{0,\phi} = (\lambda_{0,\phi_1}, \lambda_{0,\phi_2}, \lambda_{0,\phi_3}, \lambda_{0,\phi_4})
\]
DelegVerify: The deputy signer who wants to sign the document randomly chooses a positive integer \( \beta \) and sets his overt deputy signature for time slot \( \tau \) as

\[
t_{\text{Update}, \phi} = (t_{\text{Update}, \phi_1}, t_{\text{Update}, \phi_2}) = ((\phi_1^{\beta})^{1-h_{1-2}} \phi^{1-h_{1-3}} \theta, (\phi_2^{\beta})^{1-h_{1-2}} \phi^{1-h_{1-3}} \theta).
\]

DelegVerify: The user who wants to check the validity of a deputy signature \( S_{\psi, \phi} \) for the time slot \( \tau \) parses the deputy signature as \( (S_{\psi, \phi_1}, S_{\psi, \phi_2}, S_{\psi, \phi_3}) \) at first. Then he checks the truth of the equation below.

\[
\hat{p}(S_{\psi, \phi}, \eta) = \hat{p}(\eta, \eta')\hat{p}(\phi_1^{\beta})^{1-h_{1-2}} \phi^{1-h_{1-3}} \theta, \hat{p}(\phi_2^{\beta})^{1-h_{1-2}} \phi^{1-h_{1-3}} \theta, S_{\psi, \phi_3}).
\]

GenPSig: If the deputy signer accepts the deputy document \( S_{\psi, \phi} \) he randomly chooses three positive integers \( \beta_{\psi, \phi}, \beta_{\psi}, \beta_{\psi}', \beta_{\psi}'' \). and constructs his short-lived deputy signing key for time slot \( \tau \) as

\[
t_{\text{DelegVerify}, \phi} = (t_{\text{DelegVerify}, \phi_1}, t_{\text{DelegVerify}, \phi_2}, t_{\text{DelegVerify}, \phi_3}, t_{\text{DelegVerify}, \phi_4}).
\]
\[ \hat{p}(\eta, \eta, \eta, \eta) \hat{p}(\sigma_1' \Pi \phi, S_{r, 1}) \hat{p}(\sigma_1' \Pi \phi, S_{r, 1}) \hat{p}(\sigma_1' \Pi \phi, S_{r, 1}) \]
\[ \hat{p}(\sigma_1' \Pi \phi, S_{r, 1}) \hat{p}(\sigma_1' \Pi \phi, S_{r, 1}) \hat{p}(\sigma_1' \Pi \phi, S_{r, 1}) \hat{p}(\sigma_1' \Pi \phi, S_{r, 1}) \]
\[ \hat{p}(\sigma_2' \Pi \gamma, S_{r, 1}) \]

We show the truth of the computation of the proposed deputy signature above for the time slot \( \tau \) with two long-lived devices as below.

\[ \hat{p}(S_{r, 1}, \eta) = \hat{p}(\eta^2) \hat{p}(\sigma_1' \Pi \phi, \eta^2) \hat{p}(\sigma_1' \Pi \phi, \eta^2) \hat{p}(\sigma_1' \Pi \phi, \eta^2) \]
\[ \hat{p}(\sigma_2' \Pi \gamma, \eta^2) \hat{p}(\sigma_2' \Pi \gamma, \eta^2) \hat{p}(\sigma_2' \Pi \gamma, \eta^2) \hat{p}(\sigma_2' \Pi \gamma, \eta^2) \]

For a scenario where the user has only one long-lived device and has only one long-lived key, we modify the above method of IBKPDS and show the deputy signing key separated signature based on identity with only one long-lived device (IBKPDS1) as below.

(1) Su: \( \Omega_l \) is a cyclic group of multiplication. The order of \( \Omega_l \) is a prime number. \( \Omega \) is a cyclic group of multiplication. The order of \( \Omega \) is a prime number. \( \Omega \times \Omega \rightarrow \Omega \) is a bilinear pairing \( \hat{p} \) of which the generator is \( \eta \). We pick a random integer \( a \), \( \eta^a \) is \( \eta \) to the power of \( a \). We pick a random integer \( \eta^2 \) from \( \Omega \). We pick a random integer \( a \), \( F_{a1} : \{0, 1\}^* \rightarrow \{0, 1\}^a \) is a function that is of hash and against collision. We pick a random integer \( a \). \( F_{a2} : \{0, 1\}^* \rightarrow \{0, 1\}^a \) is a function that is of hash and against collision. \( F_{a3} \) is used to modify the length of the identity to the user's defined length. \( F_{a2} \) is used to modify the length of a message to the user's defined length. \( F_{a3} \) is used to modify the length of a deputy document to the user's defined length. We pick a random integer \( a \). \( \eta^a \) is equal \( \eta^a \), \( \eta^a \), \( \eta^a \), and \( \eta^a \) are random integers from \( \Omega \). The vector \( \Phi \) is equal to \( \phi \). The length of \( \Phi \) is \( \sigma_1 \). The vector \( \Gamma \) is equal to \( \gamma \). The length of \( \Phi \) is \( \sigma_2 \). The vector \( \Psi \) is equal to \( \psi \). The length of \( \Gamma \) is \( \sigma_3 \). \( \Omega_1 \), \( \Omega_2 \), \( \hat{p} \), \( \eta \), \( \eta^a \), \( \sigma_1' \), \( \Phi \), \( \sigma_2' \), \( \Gamma \), \( \sigma_3' \) and \( \Psi \) are the overt constants. Then the public agency gives the main privy constant and some overt constants to the people involved.

(2) Ext: The foremost signer’s identity \( \Phi \) is a string of many bits. The length of \( \Phi \) is \( \sigma_1 \). The length of \( \Phi \) is \( \sigma_1 \) to the set of the subscript \( i \), in which \( \Phi \) is equal to 1. The elements of \( \Phi \) are integers which are less than \( \sigma_1 \) and bigger than 1. The public agency randomly chooses a positive integer \( \beta_0 \) and sets the foremost signer’s privy signing key as \( \lambda_{\Phi} = (\lambda_{\Phi_1}, \lambda_{\Phi_2}, \lambda_{\Phi_3}) \), \( \eta_{\Phi}^2 \), \( \eta_{\Phi} \).

The deputy signer’s identity \( \Phi \) is a string of many bits. The length of \( \Phi \) is \( \sigma_2 \). The length of \( \Phi \) is \( \sigma_2 \) to the set of the subscript \( i \), in which \( \Phi \) is equal to 1. The elements of \( \Phi \) are integers which are less than \( \sigma_1 \) and bigger than 1. (0, \( \Phi \)) is a string of many bits. The length of (0, \( \Phi \)) is \( \sigma_1 \). The length of (0, \( \Phi \)) is \( \sigma_1 \) to the set of (0, \( \Phi \)). \( \Phi_{\Phi_0} \) is the output of \( \Phi_{\Phi_0} \rightarrow \{0, 1\}^3 \) and equal to the set of the subscript \( i \), in which \( \Phi \) is equal to 1. The elements of \( \Phi_{\Phi_0} \) are integers which are less than \( \sigma_1 \) and bigger than 1. The public agency randomly chooses \( \eta_{\Phi} \) from \( \{0, 1\}^3 \). The function \( \Phi_{\Phi_0} \) is random and pseud. \( \lambda_{\Phi_0} \) is equal to \( \Phi_{\Phi_0} \). The public agency randomly chooses a positive integers, \( \beta_0 \), and sets the deputy signer’s short-lived privy signing key for time slot 0 as

\[ \lambda_{\Phi_0} = (\lambda_{\Phi_0, \Phi_0}, \lambda_{\Phi_0, \Phi_2}, \lambda_{\Phi_0, \Phi_3}) \]
\[ \eta_{\Phi_0}^2 = (\eta_{\Phi_0}^2, \eta_{\Phi_0}^2, \eta_{\Phi_0}^2), \]

(3) DUpdateLongLived: \( (\tau, \Phi) \) is a string of many bits. The length of \( (\tau, \Phi) \) is \( \sigma_1 \). The length of \( (\tau, \Phi) \) is \( \sigma_1 \) to the set of \( \Phi_{(\tau, \Phi)} \). \( \Phi_{(\tau, \Phi)} \) is the output of \( \Phi_{(\tau, \Phi)} \rightarrow \{0, 1\}^3 \) and equal to the set of the subscript \( i \), in which \( \Phi \) is equal to 1. The elements of \( \Phi_{(\tau, \Phi)} \) are integers which are less than \( \sigma_1 \) and bigger than 1. \( k_{\tau, \Phi} \) is equal to \( \Phi_{(\tau, \Phi)} \). The deputy signer sets his privy short-lived refreshing key for time slot \( \tau \) as

\[ \tau_{ui_{\Phi_0}} = (\tau_{ui_{\Phi_0}}, \tau_{ui_{\Phi_0}}) \]
\[ = (\eta_{\Phi_0}^2, \eta_{\Phi_0}^2, \eta_{\Phi_0}^2) \]
\[ \eta_{\Phi_0}^2 = (\eta_{\Phi_0}^2, \eta_{\Phi_0}^2, \eta_{\Phi_0}^2) \]

(4) DUpdateUser: Using his privy short-lived refreshing key, \( \tau_{ui_{\Phi_0}} \), for time slot \( \tau \), the deputy signer sets his short-lived privy signing key for time slot \( \tau \) as

\[ \lambda_{\Phi_{\Phi_0}} = (\lambda_{\Phi_{\Phi_0}}, \lambda_{\Phi_{\Phi_2}}, \lambda_{\Phi_{\Phi_3}}) \]
\[ = (\lambda_{\Phi_{\Phi_0}}, \lambda_{\Phi_{\Phi_0}}, \lambda_{\Phi_{\Phi_0}}) \]
\[ = (\eta_{\Phi_0}^2, \eta_{\Phi_0}^2, \eta_{\Phi_0}^2) \]

(5) DelegGen: The authorization \( \lambda \) is a string of many bits. The length of \( \lambda \) is \( \sigma_3 \). \( \psi \) is the nth bit of \( \lambda \). \( \psi_{\Phi} \) is equal to the
The elements of \( \mathcal{Y}_\psi \) are integers which are less than \( \sigma_3 \) and bigger than 1. The signer randomly chooses a positive integer \( \beta_\psi \) and sets his overt deputy document as

\[
S_\psi = (S_{\psi_1}, S_{\psi_2}, S_{\psi_3}) = (\lambda_{\psi_1}, (\sigma_3' \Pi \psi_1)_{\psi_0}, \eta_{\psi_1}, \eta_{\psi_0}, \beta_{\psi_0})
\]

\[
= (\eta_{\psi_2}' (\sigma_1' \Pi \phi)_{\psi_0} (\sigma_3' \Pi \psi_1)_{\psi_0}, \eta_{\psi_0}, \eta_{\psi_0}, \beta_{\psi_0})
\]

Then the foremost signer gives the deputy tuple \((\psi, S_\psi)\) to the deputy signer.

(6) DelegVerify: The user who wants to check the validity of a deputy document \( S_\psi \) parses the deputy document as \((S_{\psi_1}, S_{\psi_2}, S_{\psi_3})\) at first. Then he checks the truth of the equation below.

\[
\hat{p}(S_{\psi_1}, \eta) = \hat{p}(\eta_{\psi_2}' \hat{p}((\sigma_1' \Pi \phi)_{\psi_0}, S_{\psi_2}) \hat{p}((\sigma_3' \Pi \psi_1)_{\psi_0}, S_{\psi_3})
\]

(7) GenPsig: If the deputy signer accepts the deputy document \( S_\psi \), he randomly chooses two positive integers \( \beta_{\psi_0}', \beta_{\psi_0}' \), and constructs his short-lived deputy signing key for time slot \( \tau \) as

\[
tdsk = (S_{\psi_2}, (\sigma_3' \Pi \psi_1)_{\psi_0}, \lambda_{\psi_{2}}, (\sigma_1' \Pi \phi)_{\psi_0}, \eta_{\psi_2}' \hat{p}((\sigma_1' \Pi \phi)_{\psi_0}, S_{\psi_2}) \hat{p}((\sigma_3' \Pi \psi_1)_{\psi_0}, S_{\psi_3})
\]

\[
= (\eta_{\psi_2}' (\sigma_1' \Pi \phi)_{\psi_0} (\sigma_3' \Pi \psi_1)_{\psi_0}, \eta_{\psi_0}, \eta_{\psi_0}, \beta_{\psi_0}')
\]

\[
= (\eta_{\psi_2}' (\sigma_1' \Pi \phi)_{\psi_0} (\sigma_3' \Pi \psi_1)_{\psi_0}, \eta_{\psi_0}, \eta_{\psi_0}, \beta_{\psi_0}')
\]

(8) PSsignatureGen: The message \( \gamma \) is a string of many bits. The length of \( \gamma \) is \( \sigma_2 \). \( \gamma_i \) is the \( i \)th bit of \( \gamma \). \( \Gamma_\tau \) is equal to the set of the subscript \( i \), which \( \gamma_i \) is equal to 1. The elements of \( \Gamma_\tau \) are integers which are less than \( \sigma_3 \) and bigger than 1. The deputy signer randomly chooses two positive integers \( \beta_{\gamma}, \beta_{\gamma}' \), and sets his overt deputy signature for time slot \( \tau \) as

\[
S_{\tau, \gamma} = (S_{\tau_1, \gamma}, S_{\tau_2, \gamma}, S_{\tau_3, \gamma}, S_{\tau_4, \gamma}, S_{\tau_5, \gamma}, S_{\tau_6, \gamma})
\]

\[
= (\sigma_1' \Pi \phi, (\sigma_3' \Pi \psi_1)_{\psi_0}, (\sigma_2' \Pi \gamma)_{\psi_0},
\]

\[
= (\sigma_1' \Pi \phi, (\sigma_3' \Pi \psi_1)_{\psi_0}, (\sigma_2' \Pi \gamma)_{\psi_0},
\]

\[
\eta_{\psi_0}, \eta_{\psi_0}, \beta_{\psi_0}, \beta_{\psi_0}
\]

\[
= (\sigma_1' \Pi \phi, (\sigma_3' \Pi \psi_1)_{\psi_0}, (\sigma_2' \Pi \gamma)_{\psi_0},
\]

\[
\eta_{\psi_0}, \eta_{\psi_0}, \beta_{\psi_0}, \beta_{\psi_0}
\]

\[
= (\sigma_1' \Pi \phi, (\sigma_3' \Pi \psi_1)_{\psi_0}, (\sigma_2' \Pi \gamma)_{\psi_0},
\]

\[
\eta_{\psi_0}, \eta_{\psi_0}, \beta_{\psi_0}, \beta_{\psi_0}
\]

\[
= (\sigma_1' \Pi \phi, (\sigma_3' \Pi \psi_1)_{\psi_0}, (\sigma_2' \Pi \gamma)_{\psi_0},
\]

\[
\eta_{\psi_0}, \eta_{\psi_0}, \beta_{\psi_0}, \beta_{\psi_0}
\]

\[
= (\sigma_1' \Pi \phi, (\sigma_3' \Pi \psi_1)_{\psi_0}, (\sigma_2' \Pi \gamma)_{\psi_0},
\]

\[
\eta_{\psi_0}, \eta_{\psi_0}, \beta_{\psi_0}, \beta_{\psi_0}
\]

IV. Security

The proposed IBKPDS is secure from the proof of the following two theorems.

**Theorem 1.** The deputy signing key is isolated so that The proposed IBKPDS is against key exposure.

*Proof:* We use a match between a defier and an opponent to simulate the scenario in which the short-lived deputy signing keys are isolated against the corruption of the opponent. The match in which the defier competes with opponent is shown as below. The opponent can ask the defier some questions and know \( \eta \) to the power of \( a \) and \( \eta \) to the power of \( b \). Although the opponent does not know \( a \) and \( b \), he wants to know \( \eta \) to the power of \( ab \) using the answers of the defier. The defier tosses a coin and decides which game he will play with the opponent.
If the defier gets the front of the coin, he will play game I with
the opponent. If the defier gets the back of the coin, he will play
Game II with the opponent.

Game I:
In Game I, the defier assumes that the opponent
the opponent can get only one long-lived key and can not two
long-lived keys.

The opponent and the defier interact with each other as
follows.

1. Initiation. $ku$, $kv$ and $km$ are randomly chosen from $Z$ by
the defier. $k_a$ is a positive integer. $lu$ is set to be
$3(q_2 + 2q_1 + 2q_0 + 2p_m)$ and $lv$ is set to be
$3(q_2 + 2q_1 + 2q_0 + 2p_m)$.

2. The question of the short-lived deputy signing key. When
the input is $\langle u \rangle$, the defier will say “error” to exit if $R(u)$ is 0.
When $R(u)$ is not 0, the defier sets the short-lived deputy privy
signing key for time slot $\tau$ as

3. The question of the deputy document. When the input is $\langle u \rangle$, the defier will say “error” to exit if $R(u)$ is 0. When $R(u)$ is not
0, the defier sets the proxy signature for time slot $\tau$ as

4. The question of the deputy signature. When the input is $\langle u \rangle$, the defier will say “error” to exit if $R(u)$ is 0. When $R(u)$ is not
0, the defier sets the proxy signature for time slot $\tau$ as

(3) Forgery. The simulation will succeed in computing $s^{ab}$
when the defier does not say “error” and answers the above
questions.

Game II:
The simulation is similar to that of Game I. In Game II, however, the opponent
can obtain only one of the two long-lived keys.

The running time and advantage can be computed using the
method of IBS (identity based signature).

Theorem 2. The deputy signing key is forcefully isolated so that
The proposed IBKPDS is forcefully against key exposure.
Proof: The proof is the same with that of Theorem 1 except that in
the opponent can obtain two long-lived keys.

V. ASSESSMENT

We give the computational complexity of the algorithms of
the proposed IBKPDS system as follows. There is not any
pairing computation in the algorithms of DUUpdateLongLived,
DUUpdateUser, DelegGen, GenPSig and PSSignatureGen.
DelegVerify and PSSignatureVerify need 4 and 8 pairing
computations respectively.
We assess the proposed IBKPDS in Table 1. $\tau_1$ is equal to be computation time of pairings. $\tau_2$ is equal to the exponential computation time. $|\mathcal{g}|$ is equal to the number of bits used for a time slot. $|\mathcal{O}|$ is equal to the number of bits used for an element in $\mathcal{O}$.

Our IBKPDS signatures consist of 7 group elements and $|\mathcal{g}|$. To defend against key exposure, we refresh the deputy signing keys at regular intervals in the IBKPDS system. So the signature size is relatively large and the verification needs more pairing computations. In the aspect of signature size and computational cost, our IBKPDS scheme is not competitive with traditional deputy signature schemes because we trade the decreased efficiency for the increased security.

| Scheme                      | PERFORMANCE                      |
|-----------------------------|----------------------------------|
|                             | Proxy signature length | Proxy signature issuing | Verification of proxy signature |
| Proposed IBKPDS Scheme      | 1$|\mathcal{g}|+7|\mathcal{O}|$   | $6\tau_2$               | $8\tau_3$          |

Table 1. the performance of our IBKPDS scheme

VI. CONCLUSION

It is required by classical deputy signature systems that deputy signing keys be securely retained. In reality, however, there appears to be a certain inevitability for the keys to be exposed due to theft, virus, network vulnerability etc. The deputy signature system will not be secure at all if deputy signing keys are exposed.

To defend against the above threat of key exposure, we give the key protected deputy signature (IBKPDS) using the method of parallel key insulation. The given IBKPDS is shown to be secure with the cryptographic proof. We evolve the deputy signing keys with two physically-secure computationally-limited devices. An attacker who obtained the short-lived deputy signing keys of some time slots can not compromise the safety of the IBKPDS system of any other time slots.

ACKNOWLEDGMENT

At last, we are particularly grateful to the experts who review this paper for their priceless remarks.

REFERENCES

[1] N. Ghardane,“On public key cryptosystem based on the problem of solving a non linear system of polynomial equations”, WSEAS Transactions on Computer Research, vol.8, pp. 106-110, 2020.
[2] A. Chillali,“Finite Ring Of Characteristic 2 And Cryptography”, Int. J. of Applied Mathematics, Computational Science and Systems Engineering, vol.2, pp. 1-4, 2020.
[3] S. Dutta, and K. Saini,“Securing Data: A Study on Different Transform Domain Techniques”, WSEAS Transactions on Computer Research, vol.16, pp. 110-120, 2021.
[4] W. Fang, W. Chen, W. Zhang, J. Pei, W. Gao, and G. Wang,“Digital Signature Scheme for Information Non-repudiation in Blockchain: a State of the Art Review”, EURASIP Journal on Wireless Communications and Networking, no.56, pp. 1-15, 2020.
[5] G. Verma, B. Singh, N. Kumar, M. Obaidat, B. He, and H. Singh,“An efficient and provable certificate-based proxy signature scheme for IloT environment”, Information Sciences, vol. 518, pp. 142-156, 2020.
[6] F. Wu, W. Yao, X. Zhang, W. Wang, and Z. Zheng,“Identity-based proxy signature over NTRU lattice”, International journal of communication systems, vol. 32, no.3, pp. e5867.1-e5867.11, 2018.
[7] R. Gao, and J. Zeng,“Forward secure certificateless proxy multi-signature scheme”, International Journal of Electronic Security and Digital Forensics, vol. 13, no.1, pp. 1-27, 2021
[8] S. Singh, and S. Padhye,“Identity based blind signature scheme over NTRU lattices”, Information Processing Letters, vol. 155, pp.105898.1-105898.3, 2020.
[9] S. James, G. Thumbur, and P.V. Reddy,“An Efficient Pairing-Free Identity Based Proxy Blind Signature Scheme with Message Recovery”, International Journal of Information Security and Privacy, vol. 15, no.1, pp. 117-137, 2021.
[10] K. Tiliwalidi, J. Zhang, and S. Xie,”A Multi-bank E-Payment Protocol Based on Quantum Proxy Blind Signature”, International Journal of Theoretical Physics, vol. 58, no.10, pp. 3510-3520, 2020.
[11] J. Zhang,” Improvement of ID-based Proxy Re-signature Scheme with Pairing-free”, Wireless Networks, vol. 25, no.7, pp. 4319-4329, 2019.
[12] F. Luo, S. Al-Kuwaria, W. Susilo, and D. Duong,”Attribute-based Proxy Re-signature from Standard Lattices and its Applications”, Computer Standards & Interfaces, vol. 75, pp. 1-9, 2020.
[13] H. Zhu, Y. Wang, C. Wang, and X. Cheng,“An efficient identity-based proxy signcryption using lattice”, Future Generation Computer Systems, vol. 117, pp. 321-327, 2021. .

Creative Commons Attribution License 4.0 (Attribution 4.0 International , CC BY 4.0)

This article is published under the terms of the Creative Commons Attribution License 4.0 https://creativecommons.org/licenses/by/4.0/deed.en_US