JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 38, 1051-1063 (2022)
DOI: 10.6688/JISE.202209_38(5).0011

A Generic Construction of Designated Verifier Signature from Standard Cryptographic Algorithms*

JUN-RUI WANG[1], XIN XIE[2] AND YU-CHI CHEN[1,2,+]
[1]*Department of Computer Science and Engineering*
Yuan Ze University
Taoyuan, 320 Taiwan
[2]*Department of Computer Science and Information Engineering*
National Taipei University of Technology
Taipei, 106 Taiwan
E-mail: {fs1076027; s1066052g}@mail.yzu.edu.tw; wycchen@ieee.org[+]

Designated verifier signature (DVS) is a variant of digital signature which can designate a verifier to verify signatures. The key difference between message authentication code and DVS is that there is no initial and shared key in DVS. In this paper, we propose a new generic construction of DVS from standard cryptographic algorithms. Our DVS construction is very modular and composed of a few fundamentally cryptographic primitives (*i.e.*, message authentication code, public key encryption and collision resistant hash). Based on the DVS construction, we can also obtain a generic construction of threshold designated verifier signature with only slight modification.

Keywords: designated verification, signature, standard cryptographic algorithms, threshold, security

1. INTRODUCTION

Recently, IT products and network technologies are widely used for realizing, such as smart phone and smart watch which can somehow achieve the functionalities of the traditional products like blood pressure monitor or electrocardiogram. The such device can be referred to as a sensor which can further send the data to institutes (*i.e.*, hospitals) to help users to perform physical examination. For example, the arterial blood pressure result is send to the department of cardiology, and the sleep monitoring result is sent to the department of psychiartry. In a nutshell, it forms a consortium service, where each department is responsible for the different examination.

In some practical applications such as e-voting, signatures are not suitable for public verification, as only few persons have sufficient privileges to verify these special signatures. To deal with this requirement, Chaum and van Antwerpen [1] presented an undeniable signature scheme in which the signature verification process can only be carried out by a verifier and a signer together. That is to say, a signer has the absolute control over who is able to verify his/her signatures. Nevertheless, undeniable signature schemes

Received August 27, 2020; revised December 21, 2020; accepted March 17, 2021.
Communicated by Raylin Tso.
[+] Corresponding author.
* This work was supported in part of Ministry of Science and Technology of Taiwan, under grants 106-2218-E-155-008-MY3 and 109-2628-E-155-001-MY3.

have an obvious drawback that a signer has to be involved in every signature verification process.

In an ordinary digital signature scheme [2, 3], anyone can use the public key of the signer to verify the validity of the signature. However, the public verification of an ordinary signature is not desired, since the signer may not want the recipient of the digital signature to show this signature to a third party at will. In order to solve the above problems, in 1989, Chaum and van Antwerpen introduced undeniable signature [1, 4] that gives signers full control over their signatures. That is, the verification of this signature requires the participation of the signer (by means of an interactive protocol) to avoid the validity of the signature of the undesired verifier. However, these signatures do not always achieve their goal, since the signer does not know to whom he is proving the validity of a signature [5].

For overcome the above issue, Chaum proposed designated confirmer signature scheme [6], where the designated parties can confirm the signature without signer's help. Later, Jakobsson *et al*. [7] proposed a designated verifier signature (DVS) scheme. Precisely, the DVS scheme enables the signer to convince the designated verifier that signer has signed a statement so that designated verifier cannot transfer the signature to a third party. This is achieved since designated verifier himself can efficiently simulate a signature which is indistinguishable from Signer's signature. This property of DVS is non-transferable. Due to the property, DVS is suitable for E-voting, tendering and software licensing. Additionally, Jakobsson *et al*. also introduced a stronger designated verifier scheme (SDVS). In this stronger notion, no third party can even verify the validity of the designated verifier's signature, because the private key of the designated verifier is required during the verification phase. In [8], Saeednia *et al*. firstly formalized the concept of strong DVS, and proposed an effective solution in their paper. In 2004, Laguillanumie *et al*. [9, 10] gave two variants of designated verifier signature schemes. To date, there are many variants of DVS systems are proposed, such like [11–13]

As an extension of the DVS scheme, Steninfeld *et al*. [14] proposed a universal designated verifier signature (UDVS) in Asiacrypt in 2003. Such a scheme enables any holder of a signature to designate a third party as the designated verifier to check the validity of this signature. And other extension for the DVS scheme, Feng *et al*. [15] proposed a scheme composed of 2-party ring signature to construct a DVS scheme and another scheme composed of deniable authenticated key exchange. And Rastegari *et al*. [16] proposed the multi-designated verifiers signature with threshold verifiability. It uses bilinear map to construct a multi-designated verifier signature.

In addition, in 2014, Shim [17] emphasized the importance of non-commissionable SDVS programs. Specifically, without revealing the knowledge of the private key, neither the signer nor the designated verifier can delegate its signing rights to any third party. He also demonstrated that the DVS scheme relies on common values such as Diffie-Hellman type values or bilinear Diffie-Hellman type value for generating and verifying an SDVS are all delegatable.

1.1 Contribution

The methods of Steninfeld *et al*.'s, Feng *et al*.'s and Susilo *et al*.'s [14, 15, 18] used difficult assumptions to construct DVS systems. We hope to construct a DVS system

based on some standard cryptographic algorithms and it can be extended easily by replacing algorithms. In this paper, we present a generic construction of DVS and standard proof, we also construct threshold DVS as an extension system. Comparing with non-generic (specific) constructions, a generic construction can easily replace the underlying primitives for different purposes; for example, we can rely on learning with error-based primitives to have a construction against quantum attacks.

Technique Highlight. In the Designated Verifier Signature (or DVS) scheme, we proposed a generic and succinct designated verifier signature model. It composed of message authentication code (MAC), public key encryption (PKE), digital signature (DS) and collision resistant hash (CRH). Our proposed scheme are different from MAC, and MAC needs to share secret key to verify tag, and DVS can use public key to verify the signature. It can also apply extension for other requirements. Give example of our extension, if we run the scheme in privacy group chat room, we need to build a multi-designated verifier for participants. For this case, we replaced the PKE scheme with threshold decryption (TD).

The rest of the article is arranged as follows. In Section 2, we introduce some tools that will be used throughout this paper. In Section 3, we show the system model and security definition. In Section 4, we propose our DV-signature construction, and Section 5 provided proof details. Then discussions are presented in Section 6. Finally, Section 7 concludes this paper.

2. PRELIMINARIES

In this section, we denote the security parameter and probabilistic polynomial time by λ and PPT. The following assumptions are produced by using λ. Moreover, we have to define the negligible function. It is a function that for every polynomial time function $p(\lambda)$ and all sufficiently large λ, $negl(\lambda) \leq \frac{1}{p(\lambda)}$ holds. Then, we review the background of Message Authentication Codes, public key encryption, digital signature and collision resistant hash describing their definition and their security where the length of keys is some polynomial of λ.

2.1 Public Key Encryption

A public-key encryption (PKE) scheme consists of three probabilistic polynomial-time (PPT) algorithms, $\Pi = (\mathsf{PKE.Gen}, \mathsf{Enc}, \mathsf{Dec})$.

- $\mathsf{PKE.Gen}(1^\lambda)$: The generator algorithm takes a security parameter λ as input. It outputs a public key pk and a secret key sk. We assume that each key has at least n length.
- $\mathsf{Enc}(pk, m)$: The encryption algorithm takes the public key pk and a message $m \subset M$ as input. It outputs the ciphertext C.
- $\mathsf{Dec}(sk, C)$: The decryption algorithm takes the secret key sk and the ciphertext C as input. It outputs the message m if its valid or $\perp$ if invalid.

Definition 1: *A public key encryption scheme* Π *is eavesdropping secure (also CPA secure) if all PPT adversary* $\mathscr{A}$ *has negligible probability to distinguish* $\mathsf{Enc}(pk, m_0)$ *and* $\mathsf{Enc}(pk, m_1)$ *formally as*

$$\mathsf{Enc}(pk, m_0) \approx \mathsf{Enc}(pk, m_1).$$

2.2 Message Authentication Code

A message authentication code (MAC) [19] consists of three probabilistic polynomial-time algorithms, $\Pi = (\mathsf{MAC.Gen}, \mathsf{MAC}, \mathsf{MAC.Vrfy})$.

- $\mathsf{MAC.Gen}(1^\lambda)$: The key generator algorithm takes a security parameter λ as input. It outputs a key k.
- $\mathsf{MAC}(k, m)$: The Mac algorithm takes the MAC key k and a message m as input, where $m \in \{0,1\}^*$. It outputs a tag t.
- $\mathsf{MAC.Vrfy}(k, m, t)$: The tag verify algorithm takes the MAC key k, the message m and the tag t as input. It outputs a bit b. If the verification is valid and outputs $b = 1$, otherwise outputs $b = 0$.

Definition 2: *A message authentication code* Π *is unforgeable, if for all polynomial time adversaries there is a negligible function negl that*

$$Pr[Mac\text{-}forge_{\mathscr{A},\pi}(\lambda) = 1] \leq negl(\lambda).$$

The message authentication experiment $Mac\text{-}forge_{\mathscr{A},\Pi}(\lambda)$ outputs 1 if $\mathscr{A}$ wins.

1. Key k is generated by running $\mathsf{MAC.Gen}(1^\lambda)$.
2. $\mathscr{A}(1^n, \mathsf{MAC}(k,\cdot)) \rightarrow (m,t)$, where $\mathscr{A}$ is the adversary and it's given $\mathsf{MAC}(k,\cdot)$ and it gives access to the oracle $\mathsf{MAC}(k,\cdot)$. Q denotes the set of all queries that adversary asked its oracle.
3. $\mathsf{MAC.Vrfy}(k, m, t) = 1$ and $m \notin Q$, $\mathscr{A}$ succeeds if and only if the output of the experiment is defined to be 1.

2.3 Digital Signature

A digital signature (or DS) [20] scheme consists of three PPT algorithms, $\Pi = (\mathsf{DS.Gen}, \mathsf{Sign}, \mathsf{DS.Vrfy})$.

- $\mathsf{DS.Gen}(1^\lambda)$: The generator algorithm takes a security parameter n as input. It outputs a public key pk and a secret key sk. We assume that each key has at least n length.

- $\mathsf{Sign}(sk, m)$: The sign algorithm takes the secret key sk and a message $m \subset M$ as input. It outputs the signature σ.
- $\mathsf{DS.Vrfy}(pk, m, \sigma)$: The verify algorithm takes the public key pk, the message m and signature σ as input. It outputs a bit b, $b = 1$ if valid and $b = 0$ if invalid.

Definition 3: *A digital signature scheme* Π *is unforgeable if for all probabilistic polynomial-time adversaries* $\mathscr{A}$*, there is a negligible function negl such that*

$$Pr[Sig\text{-}forge_{\mathscr{A},\pi}(\lambda) = 1] \leq negl(\lambda).$$

This definition holds if $\mathscr{A}$ is not able to verify message m with its corresponding signature σ in the Signature experiment $Sig\text{-}forge_{\mathscr{A},\Pi}(\lambda)$:

- Pair of keys (pk, sk) are obtained by running $\mathsf{DS.Gen}(1^{\lambda})$.
- $\mathscr{A}(pk, \mathsf{Sign}(sk, \cdot)) \rightarrow (m, \sigma)$, where $\mathscr{A}$ is the adversary and it gives access to the oracle $\mathsf{Sign}(sk, \cdot)$. Q denotes the set of all queries that $\mathscr{A}$ asked its oracle.
- $\mathsf{DS.Vrfy}(pk, m, \sigma) = 1$ and $m \notin Q$. $\mathscr{A}$ succeeds if and only if the output of the experiment is defined to be 1.

2.4 Collision Resistant Hash

A collision resistant hash (or CRH) [21] function consists of a PPT algorithm, $\Pi = (\mathsf{Hash}(\cdot))$

- $\mathsf{Hash}(\cdot)$: The hash algorithm takes a string $x \in \{0,1\}^*$ and outputs a string $\mathsf{Hash}(x) \in \{0,1\}^{\mathscr{L}(n)}$.

Definition 4: *A hash function* Π *is collision resistant if for all PPT adversary* $\mathscr{A}$*, there is a negligible function negl satisfy*

$$Pr[Hash\text{-}coll_{\mathscr{A},\pi}(\lambda) = 1] \leq negl(\lambda).$$

This definition holds if $\mathscr{A}$ outputs a x' $(x' \neq x)$ that $Hash(x') = Hash(x)$.

- $\mathscr{A}(Hash(x))$. $\mathscr{A}$ succeed if and only if finds a hash value $H(x') = H(x)$.

3. SYSTEM FRAMEWORK

There are two parties in a Generic Construction of DVS (or DVS) including a signer and an intended verifier. A signer can generate a DVS signature on a message for an intended verifier such that the signature can only be verified by the intended verifier's private key. A DVS scheme is composed of five algorithms described as follows:

Notation	Description
λ	Security parameter
i	ID
m	Message
pk	Public key
sk	Secret key
σ	Signature generated by DS
m'	Message different with m
pk_i	Public key of i's
sk_i	Secret key of i's
t	The tag generated by MAC
C	The ciphertext generated by encryption
H	Hash value
Hash	Hash function
σ_{DV}	DV signature

- DVS-KeyGen(1^λ, i): It takes as input a security parameter λ and ID i, then generates user's public key pk and secret key sk.
- DVS-Sign(m, pk, sk): It first generates a DVS secret key $k \in \{0,1\}^*$, then takes as input a message m, DVS secret key k, the designated verifier public key pk and signer secret key sk and generates the DV signature σ
- DVS-Verification(pk, sk, m, σ): It takes as input a public key of signer pk, a secret key of verifier sk, a message m and signature σ. It outputs 1 if σ is a valid signature for m. Otherwise, returns 0.
- DVS-Simulation(m', σ, sk, pk): It takes as input a message m', a DVS Sign σ, the secret key of designated verifier sk and the public key of signer pk. It outputs another valid signature σ^* for m'.

Correctness. A DVS scheme is said to be correct where DV-Verify$(pk, sk, m, \sigma) = 1$ for all $\sigma =$ DV-Sign(m, pk, sk).

Later, we present security definitions: DVS-existential unforgeability and non-transferability.

Definition 5: *A DVS scheme is said to meet DVS-existential unforgeability if there is no adversary $\mathscr{A}$ having non-negligible advantage to win the following game played with a challenger $\mathscr{C}$*

- Setup: $\mathscr{A}$ gets all of users' public key pk.
- DVS-Sign queries: $\mathscr{A}$ can adaptively request a DVS signature on his chosen message m for a signer public key pk_i and a designated verifier public key pk_j.
- DVS-Verification queries: $\mathscr{A}$ can adaptively request the verification result of his chosen message m, and DVS σ. If signature is valid, $\mathscr{C}$ output **True**. Otherwise, an error symbol $\perp$ is return instead.
- Forgery: $\mathscr{A}$ generates $(pk_i^*, pk_j^*, m^*, \sigma^*)$ with respect to a signer and a designated verifier . $\mathscr{A}$ wins this game if σ^* is a valid signature on the message m^*, and σ^* is not obtained from any DVS-Sign query.

Definition 6: *A DVS scheme is said to meet non-transferability if there is no distinguisher $\mathscr{D}$ having non-negligible advantage to win the following game played with a challenger $\mathscr{C}$.*

- Setup: $\mathscr{C}$ runs DVS-KeyGen algorithm to obtain the key pairs of signer and verifier respectively, and gives (pk_i, pk_j) to $\mathscr{D}$.
- Sign and verfiy query: $\mathscr{D}$ queries adaptively for polynomial times in the unforgeability game.
- Challenge: $\mathscr{D}$ submits a new message m' to the challenger $\mathscr{C}$. Then $\mathscr{C}$ uniform a bit $b \in \{0,1\}$, if $b = 0$, $\mathscr{C}$ returns the signature $\sigma_0 = \mathsf{DVS\text{-}Sign}(m', pk_j, sk_i)$. Otherwise, $\mathscr{C}$ returns the signature $\sigma_1 = \mathsf{DVS\text{-}Simulation}(m', \sigma_0, sk_j, pk_i)$.
- Output: $\mathscr{D}$ outputs a bit b', if $b' = b$, $\mathscr{D}$ wins the game.

4. CONSTRUCTIONS

In this section, we present the generic construction of the DVS scheme.

4.1 Generic Construction of Designated Verifier Signature

The details of each algorithm are as follows:

- $\mathsf{DVS\text{-}KeyGen}(1^\lambda, i)$: It takes as input a security parameter λ and ID i to run $\mathsf{PKE.Gen}$ algorithm. It outputs public key $PKE.pk_i$, secret key $PKE.sk_i$, then runs $\mathsf{DS.Gen}$ algorithm. It outputs public key $DS.pk_i$ and secret key $DS.sk_i$ as follows; (1) $(\mathsf{PKE}.pk_i, \mathsf{PKE}.sk_i) \leftarrow \mathsf{PKE.Gen}(1^\lambda, i)$; (2) $(\mathsf{DS}.pk_i, \mathsf{DS}.sk_i) \leftarrow \mathsf{DS.Gen}(1^\lambda, i)$.
- $\mathsf{DVS\text{-}Sign}(m, PKE.pk_j, DS.sk_i)$: It generates the DVS secret key $k \in \{0,1\}^*$ and runs MAC on a chosen message $m \in_R \{0,1\}^*$ with DVS secret key k to compute (1) $k \leftarrow \mathsf{MAC.Gen}(1^\lambda)$, (2) $t \leftarrow \mathsf{MAC}(m,k)$.

  Second, it takes as input DVS secret key k with public key of designated verifier $PKE.pk_j$ to generate ciphertext C as follow, $C \leftarrow \mathsf{Enc}(PKE.pk_j, k)$.

  Third, it takes as input DVS secret key k and ciphertext C, then runs the collision resistant hash function to generate hash value H as follow, $H \leftarrow \mathsf{Hash}(C||k)$.

  Fourth, it generates a digital signature on the hash value H with signer secret key $DS.sk_i$ as follow, $\sigma \leftarrow \mathsf{Sign}(DS.sk_i, H)$.

  Finally, it outputs DVS signature $\sigma_{DV} = (t, C, H, \sigma)$.
- $\mathsf{DVS\text{-}Verification}$ $(DS.pk_i, PKE.sk_j, m, \sigma_{DV})$: After receiving σ_{DV}, the designated verifier needs to verify the signature σ: $\{0,1\} \leftarrow \mathsf{DS.Vrfy}(DS.pk_i, H, \sigma)$.

  If it outputs 0, the system terminates. Otherwise, designated verifier decrypts the ciphertext C: $k \leftarrow \mathsf{Dec}(PKE.sk_j, C)$, and checks the hash value: $H^* \leftarrow \mathsf{Hash}(C||k)$.

  If hash value $H^* \neq H$, the system is terminated. Otherwise, do next.
  Finally it verifies whether the received tag t is valid, as follow: $\{0,1\} \leftarrow \mathsf{MAC.Vrfy}(k, m, t)$.

- DV-Signature-Simulation(m', σ_{DV}, $PKE.sk_j$, $DS.pk_i$): The designated verifier chooses a message m' with same DV secret key to generate a valid signature for this m' as follows: (1) $\{0,1\} \leftarrow \mathsf{Vrfy}(DS.pk_i, H, \sigma)$, (2) $k \leftarrow \mathsf{Dec}(PKE.sk_j, C)$ and (3) $t' \leftarrow \mathsf{MAC}(k, m')$.

  Because we use same DV secret key k, we have same C, H and σ: (1) $C \leftarrow \mathsf{Enc}(PKE.pk_j, k)$, (2) $H \leftarrow \mathsf{Hash}(C||k)$ and (3) $\sigma \leftarrow \mathsf{Sign}(DS.sk_i, H)$.

  Then we get the valid simulation signature $\sigma'_{DV} = (t', C, H, \sigma)$.:

  Obviously, DV-Signature-Simulation scheme SIM expressed in the following formula: $\sigma'_{DV} \leftarrow \mathsf{SIM}(m', \sigma_{DV}, DS.sk_i, PKE.pk_j)$.

  This algorithm directly offers *non-transferability*. The non-transferability is the indispensable part of any DVS scheme. In the proposed scheme, the DVS-Simulation algorithm provides simulating signature for designated verifier to simulate a computationally indistinguishable signature. Therefore, the non-transferability requirement is satisfied in our scheme.

  Correctness. A DVS-signature scheme for its signature $\sigma_{DV} = (m, t, C, H, \sigma)$ is said to be correct if for all λ, DV secret key $k \subset \mathscr{K}$, $m \subset \mathscr{M}$, $\mathsf{Vrfy}(k, m, t) = 1$, where $\mathsf{t} \leftarrow \mathsf{MAC}(k, m)$. $\mathsf{DS.Vrfy}(pk_i, H, \sigma) = 1$, where $\sigma \leftarrow \mathsf{Sign}(sk_i, H)$ and $H \leftarrow \mathsf{Hash}(C||k)$. And $\mathsf{Dec}(sk, C) \rightarrow k$, where $C \leftarrow \mathsf{Enc}(pk, k)$.

5. SECURITY PROOF

In this section we provide a security proof of our scheme. Formally, we have the following theorem.

Theorem 1. *The proposed DVS scheme satisfies DV-existential unforgeability, assuming that digital signature is unforgeable, public key encryption is secure, hash function is collision resistant and MAC is unforgeable.*

Proof. Assuming that there is a PPT adversary $\mathscr{A}$ to break DVS signature, we construct a PPT algorithm $\mathscr{B}$ to break one of the assumptions (the underlying digital signature security, public key encryption security, hash security and MAC security).

Recap that the DVS includes $C = \mathsf{Enc}(PKE.pk_j, k)$, $H = \mathsf{Hash}(C||k)$, $t = \mathsf{MAC}(k, m)$, $\sigma = \mathsf{Sign}(DS.sk_i, H)$, where $\mathscr{A}$ successes to make a forgery. The forgery is denoted by $(m^*, C^*, H^*, t^*, \sigma^*)$. To complete our proof, we divide reductions into couple of cases.

Lemma 1. *In Case 1, if our DVS scheme is not DV-existentially unforgeable against a PPT adversary $\mathscr{A}$, then digital signature is not unforgeable against a PPT adversary $\mathscr{B}_{DS}$.*

Proof. We construct another PPT algorithm $\mathscr{B}_{DS}$ to break the digital signature.

- **Setup:** $\mathscr{C}$ runs the $\mathsf{DVS\text{-}KeyGen}(1^\lambda, i)$ algorithm to generate a signature public key and secret key pair (pk_i, sk_i). It gives (pk_i, sk_i) to $\mathscr{A}$.

- **Sign queries:** When $\mathscr{A}$ submits sign query(pk_j, sk_i, m), $\mathscr{B}_{DS}$ asks Sign oracle to get the signature σ. Then $\mathscr{B}_{DS}$ returns σ_{DV} includes signature σ, tag t, ciphertext C and hash value H to $\mathscr{A}$.
- **Verify query:** When $\mathscr{A}$ submits verify query$(ID_i, ID_j, m, \sigma_{DV})$, if ID_j is the designated verifier's ID, $\mathscr{B}_{DS}$ directly returns terminated symbol $\perp$. Otherwise, $\mathscr{B}_{DS}$ searches the sign query records, if σ_{DV} in records, return 1. Otherwise, $\mathscr{B}_{DS}$ returns terminated symbol $\perp$.
- **Forgery:** $\mathscr{A}$ makes a forgery $(m^*, t^*, C^*, H^*\sigma^*)$, $\mathscr{A}$ wins if the following conditions are held, (1) $\mathsf{DS.Vrfy}(pk_i, H^*, \sigma^*) = 1$, (2) $\mathsf{MAC.Vrfy}(k', m^*, t^*) = 1$ and (3) $m^* \neq m$.
  If $\mathscr{A}$ wins, then digital signature is not unforgeable:

$$Pr[\mathscr{A} = 1] = Pr[Sig\text{-}forge_{\mathscr{B},\pi}(\lambda) = 1].$$

By contradiction, $\mathscr{B}$ has negligible probability to break digital signature unforgeable, then $\mathscr{A}$ has negligible probability to forge the DVS-signature. □

We divide Case 2 into two cases:

– Case 2.1: $\mathsf{Hash}(C||k^*) = \mathsf{Hash}(C||k')$ where $k' \neq k^*$ and k' is one of k in the sign query.
– Case 2.2: The underlying k^* is identical to one of k in the sign query.

Lemma 2. *In Case 2.1, if our DVS scheme is not DV-existentially unforgeable against a PPT adversary $\mathscr{A}$, then hash function is not collision resistant.*

Proof. We construct another PPT algorithm $\mathscr{B}$ to break collision resistant hash.

- **Setup:** $\mathscr{B}$ runs the $\mathsf{DVS\text{-}KeyGen}(1^\lambda, i)$ algorithm to generate a signature public key and secret key pair (pk_i, sk_i). It gives (pk_i, sk_i) to the adversary $\mathscr{A}$.
- **Sign queries:** When $\mathscr{A}$ submits sign query(pk_j, sk_i, m), $\mathscr{B}_h$ runs $\mathsf{Hash}(k)$, then get the hash value H. Then $\mathscr{B}$ returns σ_{DV} includes the hash value H, tag t, ciphertext C and signature σ to $\mathscr{A}$.
- **Verify query:** When $\mathscr{A}$ submits verify query$(ID_i, ID_j, m, \sigma_{DV})$, if ID_j is the designated verifier's ID, $\mathscr{B}_h$ directly returns terminated symbol $\perp$. Otherwise, $\mathscr{B}$ searches the sign query records, if σ_{DV} in records, return 1. Otherwise, $\mathscr{B}$ returns terminated symbol $\perp$.
- **Forgery:** $\mathscr{A}$ makes a forgery $(m^*, t^*, C^*, H^*, \sigma^*)$. $\mathscr{A}$ wins if the following conditions are held, (1) $\mathsf{Hash}(C||k^*) = \mathsf{Hash}(C||k')$ and (2) $m^* \neq m$.
  If $\mathscr{A}$ wins, then collision hash is broken.

  By contradiction, $\mathscr{B}$ has negligible probability to break collision resistant hash, then $\mathscr{A}$ has negligible probability to break DVS.

$$Pr[\mathscr{A} = 1] = Pr[Hash\text{-}coll_{\mathscr{B},\pi}(\lambda) = 1] \leq negl(\lambda).$$

□

Lemma 3. *In Case 2.2, if our DVS is not DV-existentially unforgeable against a PPT adversary $\mathscr{A}$, and MAC is not unforgeable.*

Proof. We construct another PPT algorithm $\mathscr{B}$ to break MAC.

- **Setup:** $\mathscr{C}$ runs the $\mathsf{DVS\text{-}KeyGen}(1^\lambda, i)$ algorithm to generate a signature public key and secret key pair (pk_i, sk_i). It gives (pk_i, sk_i) to the adversary $\mathscr{A}$.
- **Sign queries:** When $\mathscr{A}$ submits sign query(pk_j, sk_i, m), $\mathscr{B}$ picks a $B \in \{0,1\}$, where $Pr[B=1] = \frac{1}{q}$. If $B=0$, $\mathscr{B}$ returns $\sigma_{DV}(m,t,C,H,\sigma)$ as usual. Otherwise, $\mathscr{B}$ asks MAC oracle for m, then get the oracle result t'. Because $\mathscr{B}$ does not have DVS secret key k, by PKE Definition 2.1:

  $$\mathsf{Enc}(pk, m_0) \approx \mathsf{Enc}(pk, m_1),$$

  $\mathscr{B}$ can produce ciphertext C' and corresponding σ', H' with a random k.[1] Finally, $\mathscr{B}$ stores $\sigma_{DV}(t', C', H', \sigma')$ and returns σ_{DV} to $\mathscr{A}$.
- **Verify query:** When $\mathscr{A}$ submits verify query$(ID_i, ID_j, m, \sigma_{DV})$, if ID_j is the designated verifier's ID, $\mathscr{B}$ directly returns terminated symbol $\perp$. Otherwise, $\mathscr{B}$ searches the sign query records, if σ_{DV} in records, return 1. Otherwise, $\mathscr{B}$ returns terminated symbol $\perp$.
- **Forgery:** $\mathscr{A}$ makes a forgery (m^*, t^*, C^*, H^*, σ^*), $\mathscr{A}$ wins if the following conditions are held, (1) $\mathsf{MAC.Vrfy}(k', m^*, t^*) = 1$ and (2) $m^* \neq m$.

If $\mathscr{A}$ wins, then MAC is broken:

$$Pr[Mac\text{-}forge_{\mathscr{B},\pi}(\lambda) = 1] \leq \frac{1}{q} Pr[\mathscr{A} = 1].$$

By contradiction, $\mathscr{B}$ has negligible probability to break MAC, then $\mathscr{A}$ has negligible probability to break DVS-signautre. □

This proof is directly done by proving Lemmas 1, 2 and 3. □

6. DISCUSSIONS

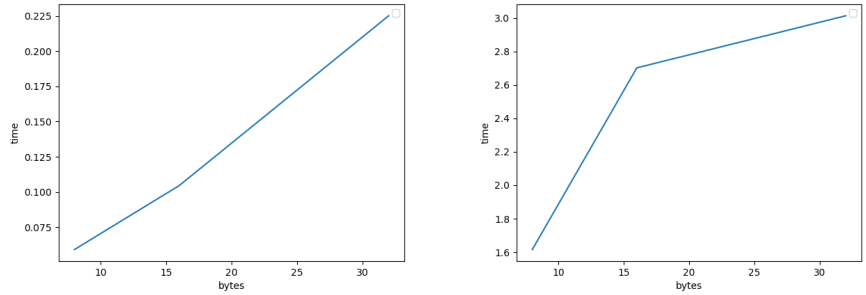
In this section, we will give some analysis and discussions about characteristics and advantages of the DVS proposed in this paper.

Comparisons. We compare our DVS system with Steninfeld *et al*.'s [14], Feng *et al*.'s [15] and Susilo *et al*.'s [18] in terms of different attributes. [14] presented two constructions of DVS systems. First system is inefficient UDVS schemes which composed of DS, PKE, trapdoor commitment (TC), second system is efficient UDVS schemes which uses assumptions of Bilinear Diffie-Hellman (BDH) and hash. [14] also presented two construction of DVS systems. And [18] proposed an ID-based DVS system which composed of BDH and TC. First system is composed of PKE and ring signature (RS), second system uses the deniable one-pass authenticated key exchange (DOP-AKE) protocol to construct DVS. The specific comparisons of these systems are shown in Table 1. It can be seen from the table that the biggest difference is that we can transform to TDV signature by changing PKE to TD.

[1] Note that if $\mathscr{A}$ can distinguish it, we can easily create an adversary to break public key encryption.

Table 1. Comparisons of DVS systems.

Scheme	Assumptions	Security	Extension to TDV
[14]-1	DS, PKE, TC	No formal proof	–
[14]-2	BDH,Hash	ROM	–
[15]-1	PKE,RS	No formal proof	–
[15]-2	DOP-AKE	No formal proof	–
[18]	BDH, TC	Standard	–
Ours	PKE, DS, MAC, CRH	Standard	✓



(a) PC with Windows. (b) Raspberry Pi.

Fig. 1. Performance.

Implementations. We use python programming to implement experiments, with Intel(R) Core(TM) i7-9700 3.00GHz and 8GB of memory, running in Windows 10 and Raspberry Pi (with 64-bit quad-core Cortex-A72 processor and 4GB LPDDR4 RAM running in ubuntu 16.04). To implement our construction, it imports the hashlib, hmac, crypto, secrets, cryptography packages. MAC key is randomly generated by secrets, and keys of PKE and DS are generated by RSA algorithm, and then we take 32, 16, 8 bytes of message to evaluate the performance in Fig. 1.

7. CONCLUSIONS

In this paper, we proposed a generic construction of designated verifier signature from standard cryptographic algorithm system which can extension by replacing algorithm, and an extended construction, threshold designated verifier signature system. It replaced PKE by TD and provided multi-verifier to verify signature without leaking the signer's information such like a privacy group chat. Finally, we provided the security proof details respectively.

REFERENCES

1. D. Chaum and H. V. Antwerpen, "Undeniable signatures," in *Proceedings of Conference on the Theory and Application of Cryptology*, 1989, pp. 212-216.
2. C. P. Schnorr, "Enhancing the security of perfect blind dl-signatures," *Information Sciences*, Vol. 176, 2006, pp. 1305-1320.
3. Y. Zhou, B. Fang, Z. Cao, X. Yun, and X. Cheng, "How to construct secure proxy cryptosystem," *Information Sciences*, Vol. 177, 2007, pp. 4095-4108.

4. D. Chaum, "Zero-knowledge undeniable signatures," in *Proceedings of Workshop on Theory and Application of of Cryptographic Techniques*, 1990, pp. 458-464.
5. Y. Desmedt and M. Yung, "Weaknesses of undeniable signature schemes," in *Proceedings of Workshop on Theory and Application of of Cryptographic Techniques*, 1991, pp. 205-220.
6. D. Chaum, "Designated confirmer signatures," in *Proceedings of Workshop on the Theory and Application of of Cryptographic Techniques*, 1994, pp. 86-91.
7. M. Jakobsson, K. Sako, and R. Impagliazzo, "Designated verifier proofs and their applications," in *Proceedings of International Conference on the Theory and Applications of Cryptographic Techniques*, 1996, pp. 143-154.
8. S. Saeednia, S. Kremer, and O. Markowitch, "An efficient strong designated verifier signature scheme," in *Proceedings of International Conference on Information Security and Cryptology*, 2003, pp. 40-54.
9. F. Laguillaumie and D. Vergnaud, "Multi-designated verifiers signatures," in *Proceedings of International Conference on Information and Communications Security*, 2004, pp. 495-507.
10. F. Laguillaumie and D. Vergnaud, "Designated verifier signatures: anonymity and efficient construction from any bilinear map," in *Proceedings of International Conference on Security in Communication Networks*, 2004, pp. 105-119.
11. B. Gong, M. H. Au, and H. Xue, "Constructing strong designated verifier signatures from key encapsulation mechanisms," in *Proceedings of the 18th IEEE International Conference on Trust, Security And Privacy in Computing and Communications/13th IEEE International Conference on Big Data Science and Engineering*, 2019, pp. 586-593.
12. P. Thorncharoensri, W. Susilo, and J. Baek, "Aggregatable certificateless designated verifier signature," *IEEE Access*, Vol. 8, 2020, pp. 95 019-95 031.
13. J. Cai, H. Jiang, P. Zhang, Z. Zheng, G. Lyu, and Q. Xu, "An efficient strong designated verifier signature based on $\mathscr{R}-$ sis assumption," *IEEE Access*, Vol. 7, 2019, pp. 3938-3947.
14. R. Steinfeld, L. Bull, H. Wang, and J. Pieprzyk, "Universal designated-verifier signatures," in *Proceedings of International Conference on the Theory and Application of Cryptology and Information Security*, 2003, pp. 523-542.
15. D. Feng, J. Xu, and W.-D. Chen, "Generic constructions for strong designated verifier signature," *Journal of Information Processing Systems*, Vol. 7, 2011, pp. 159-172.
16. P. Rastegari, M. Dakhilalian, M. Berenjkoub, and W. Susilo, "Multi-designated verifiers signature schemes with threshold verifiability: generic pattern and a concrete scheme in the standard model," *IET Information Security*, Vol. 13, 2019, pp. 459-468.
17. K.-A. Shim, "On delegatability of designated verifier signature schemes," *Information Sciences*, Vol. 281, 2014, pp. 365-372.
18. W. Susilo, F. Zhang, and Y. Mu, "Identity-based strong designated verifier signature schemes," in *Proceedings of Australasian Conference on Information Security and Privacy*, 2004, pp. 313-324.
19. B. Kaliski and M. Robshaw, "Message authentication with MD5," *CryptoBytes*, Vol. 1, 1995, No. 1.
20. Y. Lindell and J. Katz, *Introduction to Modern Cryptography*, Chapman and Hall / CRC, 2014.
21. S. Bakhtiari, R. Safavi-Naini, J. Pieprzyk *et al.*, "Cryptographic hash functions: A survey," *Citeseer*, Technical Report, 1995.

Jun-Rui Wang received the bachelor's degree from Department of Computer Science and Engineering, Yuan Ze University, Taiwan, in 2018. He is currently an MS student at Department of Computer Science and Engineering, Yuan Ze University, Taiwan. His research interests include cryptography and information security.

Xin Xie received the BS degree from the School of Ocean, Fuzhou University in 2017. He was currently pursuing the Joint Dual-MS degree in the College of Mathematics and Computer Science, Fuzhou University, China, and Department of Computer Science and Engineering, Yuan Ze University, Taiwan. His research interests are mainly in cryptography and cloud security with preserving privacy.

Yu-Chi Chen received the BS, MS, and Ph.D. degrees from Department of Computer Science and Engineering, National Chung-Hsing University, Taiwan, in 2008, 2009, and 2014 respectively. In 2013, he was a Visiting Scholar at Department of Electrical Engineering, University of Washington. He was a Postdoctoral Fellow at the Institute of Information Science, Academia Sinica, Taiwan from 2014 to 2017. He is currently an Associate Professor at Department of Computer Science and Engineering, Yaun Ze University, Taiwan. His research interests include cryptography and information security.