Certificateless Designated Verifier Signature Schemes

Xinyi Huang  
*University of Wollongong*

Willy Susilo  
*University of Wollongong*, wsusilo@uow.edu.au

Yi Mu  
*University of Wollongong*, ymu@uow.edu.au

Futai Zhang  
*Nanjing Normal University*

Follow this and additional works at: [https://ro.uow.edu.au/infopapers](https://ro.uow.edu.au/infopapers)

Part of the [Physical Sciences and Mathematics Commons](https://ro.uow.edu.au/infopapers)

**Recommended Citation**

Huang, Xinyi; Susilo, Willy; Mu, Yi; and Zhang, Futai: Certificateless Designated Verifier Signature Schemes 2006.  

Research Online is the open access institutional repository for the University of Wollongong. For further information contact the UOW Library: research-pubs@uow.edu.au
Certificateless Designated Verifier Signature Schemes

Abstract
Designated verifier signature schemes allow a signer to convince a designated verifier, in such a way that only the designated verifier will believe with the authenticity of such a signature. The previous constructions of designated verifier signature rely on the underlying Public Key Infrastructure, that requires both signer and verifier to verify the authenticity of the public keys, and hence, the certificates are required. In contrast to the previous constructions, in this paper, we propose the first notion and construction of the certificateless designated verifier signature scheme. In our new notion, the necessity of certificates are eliminated. We show that our scheme satisfies all the requirements of the designated verifier signature schemes in the certificateless system. We also provide complete security proofs for our scheme and prove that our scheme is unforgeable under the assumption of the Gap Bilinear Diffie-Hellman Problem in the random oracle model.

Disciplines
Physical Sciences and Mathematics

Publication Details
This article was originally published as: Huang, X, Susilo, W, Mu, Y & Zhang, F, Certificateless Designated Verifier Signature Schemes, 20th International Conference on Advanced Information Networking and Applications (AINA 2006), 18-20 April 2006, 2, 15-19. Copyright IEEE 2006.
Certificateless Designated Verifier Signature Schemes

Xinyi Huang1, Willy Susilo1, Yi Mu1 and Futai Zhang2

1 School of Information Technology and Computer Science
University of Wollongong, Australia
2 College of Mathematics and Computer Science
Nanjing Normal University, P.R. China

Email: xinyinjnu@126.com, wsusilo@uow.edu.au,
ymu@uow.edu.au, zhangfutai@njnu.edu.cn

Abstract: Designated verifier signature schemes allow a signer to convince a designated verifier, in such a way that only the designated verifier will believe with the authenticity of such a signature. The previous constructions of designated verifier signature rely on the underlying Public Key Infrastructure, that requires both signer and verifier to verify the authenticity of the public keys, and hence, the certificates are required. In contrast to the previous constructions, in this paper, we propose the first notion and construction of the certificateless designated verifier signature scheme. In our new notion, the necessity of certificates are eliminated. We show that our scheme satisfies all the requirements of the designated verifier signature schemes in the certificateless system. We also provide complete security proofs for our scheme and prove that our scheme is unforgeable under the assumption of the Gap Bilinear Diffie-Hellman Problem in the random oracle model.

KeyWord: Certificateless Cryptography, Designated Verifier, Gap Bilinear Diffie-Hellman Problem

I. INTRODUCTION

In a designated verifier signature scheme, the signature provides authentication of a message without providing a non-repudiation property of traditional signatures. A designated verifier scheme can be used to convince a single party, i.e. the designated verifier, and only this designated verifier who can be convinced about its validity or invalidity of the signatures, due to the fact that the designated verifier can always construct a signature intended for himself that is indistinguishable from an original signature. This kind of signature has numerous applications, for example, call for tenders, electronic voting, electronic auction, and distributed contract signing. Some recent works about the designated verifier signature are given in [5]–[10]. The first construction of the identity-based designated verifier signature scheme was proposed in [12]. In the identity-based setting, the public key is the identity of the participants themselves. However, in the latter setting, the trusted authority, known as the Private Key Generator (PKG), can always impersonate any user, and hence, the problem of key escrow is inherent in this setting.

Certificateless Cryptography was first proposed by Al-Riyami and Paterson [1] in Asiacrypt 2003. In contrast to the traditional cryptography, this notion does not require the use of any certificate to ensure the authenticity of public keys. Instead, certificateless cryptography relies on the existence of a trusted third party KGC who has the master-key. In this sense, it is similar to identity-based cryptography [11]. Nevertheless, certificateless cryptography does not suffer from the key escrow property that seems to be inherent in identity-based cryptography. In the certificateless system, KGC only knows the partial private key of the user and the user must use the secret value, which is chosen by the user himself, to obtain the full private key. For more about the certificateless system, one can refer the paper [1]. Some recent works about the certificateless system are given in [2]–[4], [13], [14].

Our Contribution In this paper, we propose the first notion and construction of the certificateless designated verifier (or CLDVS for short) signature scheme. We also provide a formal definition of the certificateless designated verifier signature. Our scheme is very efficient. Nevertheless, as we shall show in this paper, our scheme achieves all the required properties of the certificateless designated verifier signature. We provide security proofs for our scheme based on the random oracle model.

Roadmap In the next section, we will review some preliminaries required throughout the paper. In Section III, we describe our certificateless designated verifier signature. The security analysis is also given in the Section IV. At last, Section V concludes the paper.

II. PRELIMINARIES

A. Bilinear Pairing

Let $G_1$ denote an additive group of prime order $q$ and $G_2$ be a multiplicative group of the same order. Let $P$ denote a generator in $G_1$. Let $e: G_1 \times G_1 \rightarrow G_2$ be a bilinear mapping defined in [1].

Definition 1: Bilinear Diffie-Hellman (BDH) Problem.

Given a randomly chosen $P \in G_1$, as well as $aP, bP, cP$ (for unknown randomly chosen $a, b, c \in Z_q^*$), compute $e(P, P)^{abc}$. 

\[ e(P, P)^{abc} \]
Definition 2: Decisional Bilinear Diffie-Hellman (DBDH) Problem.
Given a randomly chosen \( P \in G_1 \), as well as \( aP, bP, cP \) (for unknown randomly chosen \( a, b, c \in Z_q^\ast \)) and \( h \in G_2 \), decide whether \( h = e(P, P)^{abc} \).

Definition 3: Gap Bilinear Diffie-Hellman (GBDH) Problem.
Given a randomly chosen \( P \in G_1 \), as well as \( aP, bP \) and \( cP \) (for unknown randomly chosen \( a, b, c \in Z_q^\ast \)), compute \( e(P, P)^{abc} \) with the help of the GBDH oracle.

B. Certificateless Signature Schemes
As defined in [1], a certificateless signature scheme is defined by seven algorithms: Setup, Partial-Private-Key-Extract, Set-Private-Value, Set-Private-Key, Set-Public-Key, Sign, and Verify. For a formal definition of these algorithms, we refer the reader to [1].

C. Certificateless Designated Verifier Signature Schemes
We assume there are two parties in the system, the sender \( A \) and the designated verifier \( B \). A certificateless designated verifier signature scheme is defined by eight algorithms: Setup, Partial-Private-Key-Extract, Set-Secret-Value, Set-Private-Key, Set-Public-Key, Sign, Verify and Transcript-Simulation. The first five algorithms are the same as the certificateless signature scheme defined in Section II-B, the other algorithms are defined as follows:

- **Sign**: The signing algorithm accepts a message \( m \), a parameter list \( \text{param} \), \((S_A, x_A, D_A, ID_A)\) of the sender \( A \) and the designated verifier \( B \)'s \((P_B, ID_B)\) to produce a signature \( \sigma \).
- **Verify**: The verifying algorithm accepts a message \( m \), a signature \( \sigma \), a parameter list \( \text{param} \), \((S_B, x_B, D_B, ID_B)\) of the designated verifier \( B \) and the sender's \((P_A, ID_A)\) to output \( \tau \) true if the signature is correct, or \( \bot \) otherwise.
- **Transcript-Simulation**: An algorithm that is run by the designated verifier \( B \) to produce identically distributed transcripts that are indistinguishable from the original signer \( A \).

D. Adversarial Model of Certificateless Designated Verifier Signature Schemes
As defined in [1], there are two types of adversary with different capabilities:

**Type I Adversary**: This type of adversary \( A_I \) does not have access to the master-key, but \( A_I \) has the ability to replace the public key of any entity with a value of his choice, because there is no certificate involved in certificateless signature schemes. Given the public keys of the signer and the receiver with system parameter, a type I adaptively chosen-message attacker \( A_I \) can ask the sign oracle and verify oracle in the polynomial time adaptively. At last \( A_I \) outputs a message-signature pair and the new public key of the signer. \( A_I \) is successful if the message has not been submitted to the sign oracle and the message-signature pair is valid under the public key given by \( A_I \).

**Type II Adversary**: This type of adversary \( A_{II} \) has access to the master-key but cannot perform public keys replacement. Given the public keys of the signer (and the receiver), system parameter and the system’s master-key, a type II adaptively chosen-message attacker \( A_{II} \) can ask the sign oracle and verify the oracle in the polynomial time adaptively. At last \( A_{II} \) outputs a message-signature pair. \( A_{II} \) is successful if the message has not been submitted to the sign oracle and the message-signature pair is valid.

Definition 4: A certificateless designated verifier signature scheme is existential unforgeable against adaptively chosen-message attacks iff it is secure against both types of adversaries.

III. OUR SCHEME
In this section we will propose our certificateless designated verifier signature scheme (CLDVS). We regard it as the main result of this paper. There are two parties in our scheme, the sender \( A \) and the designated verifier \( B \), all the algorithms are described as follows.

- **Setup**: This algorithm runs as follows.
  1) Run \( \mathcal{G} \) on input \( \ell \) to generate \((G_1, G_2, e)\) where \( G_1 \) and \( G_2 \) are groups of some prime order \( q \) and \( e : G_1 \times G_1 \rightarrow G_2 \) is a bilinear pairing.
  2) Select a random generator \( P \in G_1 \).
  3) Select a master-key \( s \) randomly from \( Z_q^\ast \) and set \( \bar{P}_0 = sP \).
  4) Select cryptographic hash functions \( H_1 : \{0,1\}^\ast \rightarrow G_1 \) and \( H_2 : \{0,1\}^\ast \times G_2 \rightarrow Z_q^\ast \).

The system parameters \( \text{param} = (G_1, G_2, e, q, P, P_0, H_1, H_2) \). The master-key \( s \) is in \( Z_q^\ast \). The message space \( \mathcal{M} \) is \( \{0,1\}^\ast \).

- **Partial-Private-Key-Extract**: This algorithm accepts an identity \( ID_i \in \{0,1\}^\ast, i \in \{A, B\} \) and constructs the partial private key for the user as follows.
  1) Compute \( Q_i = H_i(ID_i) \).
  2) Output the partial private key \( D_i = sQ_i \).

- **Set-Secret-Value**: This algorithm takes as inputs \( \text{param} \) and the user’s identity \( ID_i \), and selects a random \( x_i \in Z_q^\ast \) and outputs \( x_i, i \in \{A, B\} \) as the user’s secret value. That is the sender \( A \) randomly chooses \( x_A \in Z_q^\ast \) and the designated verifier \( B \) randomly chooses \( x_B \in Z_q^\ast \).

- **Set-Private-Key**: This algorithm accepts param, a user’s partial private key \( D_i \) and the user’s secret value \( x_i \in Z_q^\ast \) to transform the partial private key \( D_i \) to a full private key \( S_i \) by computing \( S_i = x_i D_i = x_i s Q_i \) and output \( S_i, i \in \{A, B\} \).

- **Set-Public-Key**: This algorithm accepts param and a user’s secret value \( x_A \in Z_q^\ast \) to produce the user’s public key \( P_A = (X_A, Y_A) \), where \( X_A = x_A P \) and \( Y_A = x_A P_B = x_A s P_0, i \in \{A, B\} \).

Now, the sender \( A \) obtains his secret key \( S_A = x_A s Q_A \) and public key \( P_A = (X_A, Y_A) = (x_A P, x_A P_B) \). The designated verifier \( B \) obtains his secret key
$S_B = x_B s_B$ and public key $P_B = (X_B, Y_B) = (x_B P, x_B P_0)$.

- **Sign:** To sign a message $m \in M$ for $B$, the signer $A$ computes the signature $\sigma = H_2(m)\{e(S_A, x_A^3Q_B + X_B)\}$.

- **Verify:** To verify a signature $\sigma$ on a message $m \in M$ from an identity $ID_A$ and public key $(X_A, Y_A)$, $B$ performs the following steps.
  1. Verify whether $e(X_A, P_0) = e(Y_A, P)$ holds with equality. If not, then output $\perp$ and abort.
  2. Verify whether $\sigma = H_2(m)\{e(Q_A, D_B + x_B Y_A)\}$ holds with equality. If it does, output true. Otherwise, output $\perp$.

- **Transcript-Simulation:** $B$ can produce the signature $\tilde{\sigma}$ intended for himself, by computing $\tilde{\sigma} = H_2(m)\{e(Q_A, D_B + e(x_B Q_A, Y_A))\}$.

**IV. SECURITY ANALYSIS**

**Theorem 1:** Our CLDVS scheme is a designated verifier signature scheme.

**Proof:** We note that the verification algorithm requires $D_B, x_B$, where $D_B$ is the partial private key of the designated verifier $B$ and $x_B$ is the secret value of $B$. Hence, $B$ can always “simulate” a valid signature by producing a valid signature himself. This is achieved by constructing a signature $\tilde{\sigma} = H_2(m)\{e(Q_A, D_B + e(x_B Q_A, Y_A))\}$. Note that the signature produced by $B$ is indistinguishable from the one that was produced by the sender $A$. Hence, no third party can be convinced with the validity or invalidity of this signature other than the designated verifier himself. If the designated verifier has not generated such a signature, then he will believe that the signature was indeed generated by the signer $A$.

**Theorem 2:** Let $A_I$ be an $\ell$-adaptively chosen-message attacker against our CLDVS scheme with success probability greater than $\text{Suc}_{\text{CLDVS},SA}^{\text{EF-CMA}}(\ell)$, after asking $q_B$ queries to the hash function $H_2$, $q_S$ queries to the sign algorithm and $q_V$ queries to the verify algorithm, then there exists an algorithm $B$ which can use $A_I$ to solve a random instance of the GBDH problem with the probability $\text{Suc}_{\text{CLDVS},SA}^{G_{\text{BDH}}}(\ell) \geq (1 - \frac{q_V}{2\ell})\text{Suc}_{\text{CLDVS},SA}^{\text{EF-CMA}}, \ell$ is the security parameter of our CLDVS scheme.

**Proof:** Given a random instance $(P, P_0 = aP, P_2 = bP_0, P_3 = eP)$ of the Gap Bilinear Diffie-Hellman (GBDH) problem, we will show how $B$ can use $A_I$ to obtain the value of $e(P, P)^{abc}$ with the help of the Decisional Bilinear Diffie-Hellman (DBDH) Oracle. In the proof, we regard the hash function $H_2$ as the random oracle. We assume $A_I$ is well-behaved in the sense that $A_I$ doesn’t repeat any two identical queries.

**Setup:** In this game, $B$ will set the system parameters. There are two parts in the proof, the sender $A$ and the designated verifier $B$. $B$ starts by set $Q_A = P_1, Q_B = P_2$ and $P_0 = P_3$ where $(P_1, P_2, P_3)$ is the instance of the Gap Bilinear Diffie-Hellman problem given to $B$. Then the algorithm $B$ also randomly chooses $x_A, x_B \in \mathbb{Z}_q^*$ and sets $P_A = (X_A, Y_A) = (x_A P, x_A P_0), P_B = (X_B, Y_B) = (x_B P, x_B P_0)$. $B$ will return all the parameters to $A_I$.

- **Hash Queries:** In this game, $B$ will simulate the hash function $H_2$. At any time algorithm $A_I$ can query the random oracle. To respond to these queries algorithm, $B$ maintains a list $H$-list which consists of the tuples $(m_i, r_i, \sigma_i, X_i, Y_i)$ as described below. The list is initially empty. When $A_I$ queries the oracle $H$ with the request $(m_i, r_i)$, algorithm $B$ checks the $H$-list:
  1. If there is no item $(m_i, r_i, \sigma_i, X_i, Y_i)$ in $H$-list, $B$ will choose a random $\sigma_i \in \mathbb{Z}_q$ such that there is no item $(\cdot, \cdot, \cdot, \cdot, \cdot)$ in the $H$-list. Then $B$ adds $(m_i, r_i, \sigma_i, \perp, \perp)$ into the $H$-list and returns $\sigma_i$ to $A_I$ as the answer. Here the notation $\perp$ means $B$ doesn’t know the corresponding value.
  2. Else, there is an item $(m_j, \cdot, \cdot, \cdot, \cdot, \cdot)$ in the $H$-list such that $m_i = m_j$.
    a) This item has the form $(m_j, r_j, \cdot, \cdot, \cdot)$ such that $r_i \neq r_j$.

Note that either way $\sigma_i$ is uniform in $\mathbb{Z}_q$ and is independent of $A_I$’s current view as required, so $B$ simulates the hash function perfectly.

- **Sign Queries:** In this game, $B$ will simulate the sign algorithm. At any time algorithm $A_I$ can query the sign algorithm and $B$ will answer $A_I$’s queries. Since $A_I$ is the type I adversary, $A_I$ can choose the public key $(X_i, Y_i)$ for the sender $A$. After receiving $A_I$’s choice of the message $m_i$ and the public key $(X_i, Y_i)$, $B$ checks whether $e(X_{i}, Y_{i}) = e(P, P)^{abc}$. If the equation does not hold, $B$ terminates this query and asks $A_I$ to choose a valid public key. Otherwise, $B$ checks the $H$-list:
  1. If there is no item $(m_j, \cdot, \cdot, \cdot, \cdot, \cdot)$ in $H$-list, $B$ will choose a random $\sigma_i \in \mathbb{Z}_q$ such that there is no item $(\cdot, \cdot, \cdot, \cdot, \cdot)$ in the $H$-list. Then $B$ adds $(m_i, r_i, \sigma_i, \perp, \perp)$ into the $H$-list and returns $\sigma_i$ to $A_I$ as the answer.
  2. Else there is an item $(m_j, \cdot, \cdot, \cdot, \cdot, \cdot)$ in the $H$-list such that $m_j = m_i$.
    a) This item has the form $(m_j, \perp, \cdot, \cdot, \cdot, \cdot)$
such that $X_A^i \neq X_A^j$.

If this case happens, $B$ will choose a random

$\sigma_i \in Z_q$ such that there is no item $(\cdots, \sigma_i, \cdots)$ in the $H$-list.

Then $B$ adds $(m_i, \perp, \sigma_i, X_A^i, Y_A^i)$ into the $H$-list and returns $\sigma_i$ to $A_I$ as the
answer.

b) Otherwise, this item must have the form

$(m_j, r_j, \sigma_j, \perp, \perp)$ which can only be added into the $H$-list during the Hash
 Queries.

If this case happens, $B$ will submit $(P_i, P_2, P_3, r_j/e(Q_A, x_B Y_A^i))$ to the DBDH
 oracle and the DBDH oracle will tell $B$ whether

$r_j/e(Q_A, x_B Y_A^i) = e(P_i, P)^{abc}$.

i) If $r_j/e(Q_A, x_B Y_A^i) = e(P_i, P)^{abc}$, then $B$ rewrites

this form as $(m_j, r_j, \sigma_j, X_A^i, Y_A^i)$. Then $B$ returns

$\sigma_j$ as the answer to $A_I$.

ii) Else $r_j/e(Q_A, x_B Y_A^i) \neq e(P_i, P)^{abc}$, then $B$ will choose a random

$\sigma_i \in Z_q$, such that there is no item $(\cdots, \sigma_i, \cdots)$ in the $H$-list. Then $B$ adds

$(m_i, \perp, \sigma_i, X_A^i, Y_A^i)$ into the $H$-list and returns $\sigma_i$ to $A_I$ as the
answer.

* Verify Queries: In this game, $B$ will simulate the verify
algorithm. At any time algorithm $A_I$ can query the verify
algorithm and $B$ will answer $A_I$’s queries. After receiving

$A_I$’s request $(m_i, \sigma_i)$ and the sender $A$’s public
key $(X_A^i, Y_A^i)$ chosen by $A_I$, $B$ checks the $H$-list:

1) If there is no item $(\cdots, \sigma_i, \cdots)$ in the $H$-list, $B$

rejects $(m_i, \sigma_i)$ as an invalid signature.

2) Else, there is an item $(\cdots, \sigma_i, \cdots)$ in the $H$-list:

a) If this item has the form of $(m_i, \perp, \sigma_i, X_A^i, Y_A^i)$ or $(m_i, \perp, \sigma_i, X_A^i, Y_A^i)$, $B$ will accept it as a
valid signature.

b) Else if this item has the form of

$(m_i, r_i, \sigma_i, \perp, \perp)$, $B$ will submit

$(P_i, P_2, P_3, r_i/e(Q_A, x_B Y_A^i))$ to the DBDH
 oracle and the DBDH oracle will tell $B$ whether

$r_i/e(Q_A, x_B Y_A^i) = e(P_i, P)^{abc}$.

i) If $r_i/e(Q_A, x_B Y_A^i) = e(P_i, P)^{abc}$, then $B$ will accept it as a valid signature.

ii) Else $r_i/e(Q_A, x_B Y_A^i) \neq e(P_i, P)^{abc}$, then $B$ re-
jects it as an invalid signature.

c) Otherwise, $B$ rejects it as an invalid signature.

This simulation works well except that $(m_i, \sigma_i)$ is a valid
signature, while $\sigma_i$ is not queried from the random oracle $H$. Since, $H$ is uniformly distributed, this case happens
with probability less than $\frac{q_v}{q_v - q_s}$.

If $B$ doesn’t fail during all the queries, $A_I$ can output a valid
message-signature pair $(m^*, \sigma^*)$ under the sender $A$’s public
key $(X_A^*, Y_A^*)$ with probability greater than $\text{Suc}_{\text{CMA}}^{\text{DBDH, A}}$. Since $(m^*, \sigma^*)$ is a valid message-signature pair, which means
there is an item $(\cdots, \sigma^*, \cdots)$ in the $H$-list. By the definition of
the adversary model, $m^*$ cannot be queried to the signer oracle,
so $\sigma^*$ is returned as the hash value of $A_I$’s query $(m^*, r^*)$.
That is to say there is an item $(m^*, r^*, \sigma^*, \perp, \perp)$ in the $H$-list
and $r^*/e(Q_A, x_B Y_A^i) = e(P, P)^{abc}$. Since $Q_A, x_B, Y_A^i, r^*$ are
all known to $B$, $B$ can successfully solves this instance of the
GBDH problem.

However, the probability $B$ doesn’t fail is greater than

$1 - \frac{q_v}{q_v - q_s}$. Therefore, $B$ can solve this instance of
the GBDH problem with the probability: $\text{Suc}_{\text{CMA}}^{\text{DBDH, B}} \geq (1 - \frac{q_v}{q_v - q_s}) \text{Suc}_{\text{CMA}}^{\text{DBDH, A}}$.

Theorem 3: Let $A_{II}$ be a type II adaptively chosen-
message attacker against our CLDVS with success proba-
bility greater than $\text{Suc}_{\text{CMA}}^{\text{DBDH, A}}$, then asking $q_H$ queries
to the hash function $H_B$, $\mathcal{g}_B$ queries to the sign algorithm and $q_v$ queries to the verify algorithm, then there exists an
algorithm $B$ can use $A_{II}$ to solve a random instance of
the GBDH problem with the probability $\text{Suc}_{\text{CMA}}^{\text{DBDH, B}} \geq (1 - \frac{q_v}{q_v - q_s}) \text{Suc}_{\text{CMA}}^{\text{DBDH, A}}$, $\ell$ is the security number of
our CLDVS scheme.

Proof: Given a random instance $(P_i, P_{s1} = aP, P_{s2} = bP, P_3 = cP)$ of
the Gap Bilinear Diffie-Hellman(GBDH) problem, we
will show how $B$ can use $A_{II}$ to obtain the value of
$e(P, P)^{abc}$ with the help of the Decisional Bilinear Diffie-
Hellman(DBDH) Oracle. In the proof, we regard the hash
function as the random oracle. We assume $A_{II}$ is well-behaved
in the sense that $A_{II}$ doesn’t repeat any two identical queries.

* Setup: In this game, $B$ will set the system parameters.
There are two parts in the proof, the sender $A$ and the
designated verifier $B$. $B$ starts by set $X_A = P_1, X_B = P_2$ and $Q_A = P_3$ where $(P_1, P_2, P_3)$ is the instance of
the Gap Bilinear Diffie-Hellman problem given to $B$. Then the algorithm $B$ also randomly chooses $s \in Z_q^*$, $Q_B \in \mathcal{G}_1$ and sets $P_0 = sP$. $P_A = (X_A, Y_A) = (P_1, sP)$, $P_B = (X_B, Y_B) = (P_2, sP)$. $B$ will return all the
parameters to $A_{II}$. Since $A_{II}$ is the type II adversary, $B$ will also send the master key $s$ to $A_{II}$.

* Hash Queries: In this game, $B$ will simulate the hash
function. At any time algorithm $A_{II}$ can query the random
oracle $H$. To respond to these queries algorithm $B$ maintains a list $H$-list which consists of
the tuples $(m_i, r_i, \sigma_i, c_i)$ as described
below. The list is initially empty. When $A_{II}$ queries
the oracle $H$ with the request $(m_i, r_i)$, algorithm
$B$ submits $(P_i, P_2, P_3, (r_i)^{-1}/e(Q_A, Q_B))$ to
the DBDH oracle and DBDH oracle will tell $B$ whether

$(r_i)^{-1}/e(Q_A, Q_B) = e(P, P)^{abc}$.

1) $(r_i)^{-1}/e(Q_A, Q_B) = e(P, P)^{abc}$, which means

$r_i = e(Q_A, sQ_B)e(P, P)^{abc} = e(Q_A, D_B + x_B Y_A)$.

a) If there is no item $(\cdots, \cdots)$ in $H$-list, $B$ will

set $c_i = 1$ and choose a random $\sigma_i \in Z_q$ such

that there is no item $(\cdots, \sigma_i, \cdots)$ in the $H$-list.

Then $B$ adds $(m_i, r_i, \sigma_i, c_i)$ into the $H$-list

and returns $\sigma_i$ to $A_{II}$ as the answer.

b) Else, there is an item $(m_j, \cdots)$ in the $H$-list

such that $m_i = m_j$. If this item has the form

$(m_j, r_j, \cdots)$ such that $m_i = m_j, r_i \neq r_j$, $B$ will

set $c_i = 1$ and choose a random $\sigma_i \in Z_q$ such

that there is no item $(\cdots, \sigma_i, \cdots)$ in the $H$-list.

Then $B$ adds $(m_i, r_i, \sigma_i, c_i)$ into the $H$-list

and returns $\sigma_i$ to $A_{II}$ as the answer.
c) Otherwise, as described below, this item must have the form \((m_i, r_j, \sigma, 1)\) (item of this form only can be added into the \(H\)-list during the Sign Queries). Then \(B\) returns \(\sigma, i\) to \(A_{l+1}\) as the answer.

2) Otherwise \((r_j)^{-1} e(Q, Q_B) \neq e(P, P)^{abc}\), \(B\) sets \(c_j = 0\) and chooses \(\sigma, i \in Z_q^*\) such that there is no item \((\cdot, \cdot, \sigma, r)\) in the \(H\)-list. Then \(B\) adds \((m_i, r_j, \sigma, 0)\) into the \(H\)-list and returns \(\sigma, i\) to \(A_{l+1}\) as the answer.

**Sign Queries:** In this game, \(B\) will simulate the sign algorithm. At any time algorithm \(A_{l+1}\) can query the sign algorithm and \(B\) will answer \(A_{l+1}\)'s queries. After receiving \(A_{l+1}\)'s choice of the message \(m_i\), \(B\) checks the \(H\)-list:

1) If \(m_i\) has never been submitted to the hash oracle, \(B\) will set \(c_j = 1\) and choose \(\sigma, i \in Z_q^*\) such that there is no item \((\cdot, \cdot, \sigma, r)\) in the \(H\)-list. Then \(B\) adds \((m_i, \perp, \sigma, 1)\) into the \(H\)-list and returns \(\sigma, i\) to \(A_{l+1}\) as the answer.

2) Else, \(m_i\) has been submitted to the hash oracle. There must be an item \((m_j, r_j, \sigma, c_j)\) in the \(H\)-list such that \(m_i = m_j\):
   a) If \(c_j = 1\), which means \(r_j = e(Q, s_B) e(P, P)^{abc} = e(Q_A, D_B + x_B Y_A)\), \(B\) returns \(\sigma, i\) to \(A_{l+1}\) as the answer.
   b) Otherwise, \(B\) will set \(c_j = 1\) and choose a random \(\sigma, i \in Z_q^*\) such that there is no item \((\cdot, \cdot, \sigma, r)\) in the \(H\)-list. Then \(B\) adds \((m_i, \perp, \sigma, 1)\) into the \(H\)-list and returns \(\sigma, i\) to \(A_{l+1}\) as the answer.

**Verify Queries:** In this game, \(B\) will simulate the verify algorithm. At any time algorithm \(A_{l+1}\) can query the verifying algorithm and \(B\) will answer \(A_{l+1}\)'s queries. After receiving \(A_{l+1}\)'s request \((m_i, \sigma, i)\), \(B\) checks the \(H\)-list:

1) If there is no item \((\cdot, \cdot, \sigma, r)\) in the \(H\)-list, \(B\) rejects \((m_i, \sigma, i)\) as an invalid signature.

2) Else, there is an item \((\cdot, \cdot, \sigma, r)\) in the \(H\)-list:
   a) If this item has the form \((m_i, \perp, \sigma, 1)\) or \((m_i, \perp, \sigma, 1)\), \(B\) will accept it as a valid signature.
   b) Otherwise, \(B\) rejects it as an invalid signature.

This simulation works well except that \((m_i, \sigma)\) is a valid message-signature pair, while \(\sigma\) is not queried from the random oracle \(H\). Since \(H\) is uniformly distributed, this case happens with probability less than \(\frac{q-1}{q}\).

If \(B\) doesn’t fail during all the queries, \(A_{l+1}\) can output a valid message-signature pair \((m^*, \sigma^*)\) with probability greater than \(\text{Succ}_{\text{CLDV S, A}_{l+1}}^{\text{EF, CMA}}\). Since \((m^*, \sigma^*)\) is a valid message-signature pair, which means there is an item \((\cdot, \cdot, \sigma^*, r^*)\) in the \(H\)-list. By the definition of the adversary model, \(m^*\) can not be queried to the sign oracle, so \(\sigma^*\) is returned as the hash value of \(A_{l+1}\)'s query \((m^*, r^*)\). That is to say there is an item \((m^*, r^*, \sigma^*, 1)\) in the \(H\)-list and \((r^*)^i - e(Q, Q_B) = e(P, P)^{abc}\). So if \(B\) doesn’t fail, \(B\) can successfully solve this instance of the GBDH problem with same probability \(\text{Succ}_{\text{CLDV S, A}_{l+1}}^{\text{EF, CMA}}\).

However, the probability \(B\) doesn’t fail is greater than \(1 - \frac{q}{q^2 - q^*}\). Therefore, \(B\) can solve this instance of the GBDH problem with the probability: \(\text{Succ}_{\text{GBDH, B}}^{\text{EF, CMA}} \geq (1 - \frac{q}{q^2 - q^*}) \text{Succ}_{\text{CLDV S, A}_{l+1}}^{\text{EF, CMA}}\).

**V. CONCLUSION**

In this paper, we proposed the notion of certificateless designated verifier signature scheme and the first construction of the certificateless designated verifier signature scheme. We showed that our scheme satisfies all the requirements of the designated verifier signature schemes. We also provided security proofs for our scheme in the random oracle model and proved that our scheme is unforgeable to both types of adversaries in certificateless model under the assumption of the Gap Bilinear Diffie-Hellman Problem.

**Acknowledgement.**

The authors would like to express their gratitude thanks to the anonymous referees of the Second International Workshop on Security in Networks and Distributed Systems (SNDS-06) for the suggestions to improve this paper.

**REFERENCES**


