An Efficient ID-based Proxy Blind Signature with Pairing-free Realization

Sahadeo Padhye\textsuperscript{1}, and Namita Tiwari\textsuperscript{2}

Abstract—Proxy blind Signature allows an original signer to delegate his/her signing capability to a proxy signer who performs message signing blindly on behalf of original signer but he cannot make any link between the blind signature and the identity of the message's owner. Using bilinear pairings, so many ID-based proxy blind signature (IDPBS) schemes have been proposed. But the computation cost of a pairing is realized very high now a days. On the other hand, map to point hash function is used very frequently in the pairing based schemes that is another very expensive primitive. Therefore we propose an efficient PBS scheme without pairings in the ID-based setting that is provable secure. With the pairing-free realization, our proposed scheme is much efficient as well as applicable than existing schemes.

Keywords—Digital signature, Identity-based cryptography, Proxy-blind signature, Elliptic curve cryptography, Provable security

I. INTRODUCTION

Identity based public key cryptography [23] simplifies the issue of public-key and certificate management by using a user’s identity (e.g., name, IP address, email address etc.) as its public key. For this, the system requires a trusted third party, typically called a ‘Private Key Generator’ (PKG), to generate user’s secret keys from its ‘master secret’ and the user’s identity. Only the PKG has a traditional public key. In an identity based signature scheme, the verifier verifies a signature by using the signer’s identity and the PKG’s public key, the verification information does not include any certificate or any individual public key for the signer.

Proxy signature was firstly introduced by Mambo et al. [19] in 1996. A proxy signature scheme allows an entity called original signer to delegate his signing capability to another entity, called proxy signer. Since it is proposed, the proxy signature schemes have been suggested for use in many applications, particularly in distributed computing.

On the other hand, blind signature also has its own importance for specific situations and purposes. Blind Signature firstly introduced by David Chaum [8], is a signature on a message, signed by another party without having any information about the message. Its name is so, because the message is blind to the signer. Blind signatures are applicable where sender’s privacy is important, like: digital cash scheme, electronic voting etc.

A proxy blind signature combines the properties of proxy signature and blind signature in which the proxy signer is allowed to generate a blind signature on behalf of the original signer. Using bilinear pairings, many new ID-based signature schemes [1, 5, 13], ID-based proxy signature (IDPS) schemes [9, 10, 14, 24, 27] and PBS schemes [17, 22, 25, 26] etc. have been proposed.

Motivation: All of the above IDPBS schemes are very practical and possess bilinear pairings based constructions. In such designs they also use the map to point hash function. Now a days, sufficient amount of literatures regards pairing as one of the very expensive cryptographic primitive. Map-to-point hash function is another very expensive operation. As per the literatures [6, 11], the relative computation cost of a pairing is approximately more than ten times of the scalar multiplication over elliptic curve group. Therefore, IDPBS schemes with pairing-free realization would be more appealing in terms of efficiency. Elliptic Curve Cryptography (ECC) is considered as an important topic in public key cryptography and independently proposed by Koblitz [15] and Miller [18]. The security of this cryptosystem is based on elliptic curve discrete log problem (ECDLP). The main advantage of ECC is that it provides the same security level with smaller key size [20]. Smaller key means less management time and smaller storage, which supplies convenience to realization by software and hardware. To achieve 1024-bits RSA level security, 512-bits supersingular elliptic curve and 160-bits non-supersingular elliptic curves are used in applications. In general pairing is defined on the supersingular elliptic curve. But the ECC uses non supersingular elliptic curves.

Our Contribution: Having motivated from [6, 11, 12], we propose an efficient PBS scheme without bilinear pairings. Because our scheme is based on the elliptic curve cryptosystem, it is efficient and have smaller key size than pairing based system. Security of the proposed scheme is based on ECDLP and proven secure in random oracle model. With the pairing-free realization, proposed scheme is much efficient than existing schemes [17, 22, 26] etc. in computation. Recently, Zuowen Tan also proposed a similar approach [21]. Our scheme is independent of the scheme [21] and computation shows that our scheme is also more efficient than the scheme [21].

Organization: The rest of this paper is organized as follows: In Section II, we introduce the complexity assumption. In Section III, we define a formal model of identity-based proxy blind-
signature scheme and it’s security. In Section IV, we propose a new identity-based proxy blind-signature scheme. In Section V, we compare the efficiency of our scheme with the schemes given in [17, 21, 22, 26]. Section VI discusses some important applications. Finally, Section VII concludes the paper.

II. PRELIMINARIES

A. Background of Elliptic Curve Group.

An elliptic curve $E$ over a prime finite field $F_p$ (denoted by $E(F_p)$) is the set of points $(x, y)$ with $x, y \in F_p$ which satisfy the equation $y^2 = (x^3 + ax + b)$ mod $p$, $a, b \in F_p$, together with an extra point $\infty$ (called the point at infinity). If the discriminant $\Delta = (4a^3 + 27b^2)$ mod $p \neq 0$, equivalently, the polynomial $x^3 + ax + b$ has distinct factors then $E(F_p)$ is nonsingular i.e. it does not have any cusp or node singularity. Therefore, we can define a binary operation (the point addition $\oplus$) on the points of $E(F_p)$ as follows: Let $P, Q \in E(F_p)$, $l$ be the line joining $P$ and $Q$ (tangent line to $E(F_p)$ if $P = Q$), $R$, the third point of intersection of $l$ with $E(F_p)$. Let $l'$ be the vertical line through $R$ which intersects the elliptic curve $E(F_p)$ at another point say $-R$. Then $P \oplus Q$ is the reflected point $-R$. There is a problem that vertical line through $P$ and $-P$ does not intersect elliptic curve $E(F_p)$ at a third point and we need a third point to define $P + (-P)$. Since there is no point in the plane that works, we create an extra point $\infty$ at infinity. Here $\infty$ is a point on every vertical line. Thus elliptic curve with this binary operation $\oplus$ forms an additive abelian group $(E(F_p), \oplus) = \{(x, y) : x, y \in F_p ; E(x; y) = 0\} \cup \{\infty\}$. Let $G$ be a cyclic additive subgroup of $(E(F_p), \oplus)$ with generator $P$ of prime order $n$.

B. Addition law of Elliptic Curve $E(a, b)$ over $F_p$

The addition algorithm for $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ on the elliptic curve $E(F_p)$ is:

1. If $P = (x_1, y_1)$ then additive inverse of $P$ is given by $-P = (x_1, -y_1)$.
2. If $Q = (x_2, y_2)$ and $Q \neq -P$, then $P \oplus Q = (x_3, y_3)$ where $x_3 = \lambda^2 - x_1 - x_2$, $y_3 = \lambda(x_1 - x_3) - y_1$, and

\[
\lambda = \frac{3x_1^2 + a}{2y_1} \text{ if } P \neq Q.
\]

C. Complexity assumption

Elliptic curve discrete logarithm problem (ECDLP): Given $x \in \mathbb{Z}_n^*$ and $P$ the generator of $G \subset E(F_p)$ and $Q \in G$, to compute $x$ s.t $Q = xP$ is called ECDLP and assumed to be intractable.

III. FORMAL MODEL OF IDENTITY-BASED PROXY-BLIND SIGNATURE SCHEME

A. Definition-1: An identity-based proxy blind signature scheme can be described as a collection of the following seven algorithms with $A$ and $B$ as original and proxy signer respectively.

- **Setup**: PKG takes a security parameter $1^k$ as input and returns a master secret key $x$ and system parameters $\Omega$.
- **Extract**: Given an identity $ID_t \in \{0, 1\}^*$, the master secret key $x$ and parameters $\Omega$, PKG generates the private key $DK_t$ of each user $U$ with identity $ID_t$.
- **DelGen**: The delegation algorithm, takes A’s secret key $DK_A$ and warrant $m_n$ as inputs and outputs the delegation $W_{A \rightarrow B}$.
- **DelVerif**: This algorithm, takes IDA, WA/AB as input and verifies whether $W_{A \rightarrow B}$ is a valid delegation came from A.
- **PKGen**: This algorithm, takes $W_{A \rightarrow B}$, secret key of the executor as inputs, and outputs a proxy signing key $DK_P$ for proxy signature.
- **PBSign**: By this algorithm, proxy signer generates the signature blindly for the message's owner C, which is often divided into three sub-algorithms (Blind, BSign, UnBlind).
  - $i$: Blind: Having interaction with proxy signer, this algorithm takes a message $m$ and a random string $r$ as input and outputs blinded message.
  - $ii$: BSign: This algorithm takes blinded information and proxy private key $DK_p$ as input and outputs a proxy blind signature $s'$ then sends it to C.
  - $iii$: UnBlind: After receiving signature $s'$, this algorithm takes previously used string $r$ and outputs unblinded signature $s$.
- **PBVerif**: Given a signature $(ID, m, s)$ and system parameters $\Omega$, this algorithm outputs $1$ if $(ID, m, s)$ is a valid proxy blind signature, otherwise outputs $0$.

B. Definition-2 : (Blindness [12]). Let $E$ be a probabilistic polynomial-time adversary which plays the role of the signer, $U_0$ and $U_1$ be two honest users. $U_0$ and $U_1$ engage in the proxy blind signature issuing scheme with $E$ on messages $m_b$ and $m_{b'}$, and output signatures $\sigma_0$ and $\sigma_{1(b,b)}$, respectively, where $b \in \{0, 1\}$ is a random bit chosen uniformly. $(m_0, m_1, \sigma_0, \sigma_{1(b,b)})$ are sent to $E$ and then $E$ outputs $b' \in \{0, 1\}$. If for all such $E$, $U_0$ and $U_1$, $|Pr[b = b'] - 1/2| < \epsilon$, for any constant $\epsilon$, and for sufficiently large $n$, then signature is said to be satisfy blindness.

IV. PROPOSED SCHEME

We propose an ID-based proxy blind signature scheme without use of pairing. Our scheme consists of following algorithms:

- **Setup**: PKG takes a security parameter $k$, returns system parameters $\Omega$ as follows.
  - $i$. Chooses a $k$-bit prime $p$ and determines the tuple $F_p, E(F_p)$, $G, P$ as defined in Section 2.
  - $ii$. Chooses the master private key $x \in \mathbb{Z}_p^*$ and computes the master public key $P_{pub} = xp$.
  - $iii$. Chooses two cryptographic secure hash functions $H_1 : \{0, 1\}^* \times G \rightarrow \mathbb{Z}_p^n$ and $H_2 : \{0, 1\}^* \times G \rightarrow \mathbb{Z}_p^n$.
  - $iv$. Publishes $\Omega = \{F_p, E(F_p), G, P, P_{pub}, H_1, H_2\}$ as system parameters and keeps the master key $x$ secretly.
Extract: Taking system parameters, master key, and users identity as input, PKG returns the ID-based private key D_\text{U} of each user U having identity ID_\text{U} as follows.

\text{i.} \text{Chooses at random } r_\text{U} \in \mathbb{Z}_q^{*}, \text{ computes } R_\text{U} = r_\text{U} \cdot P \text{ and } h_u = H_i(ID_\text{U}, R_\text{U}).

\text{ii. Computes } D_\text{U} = r_\text{U} + h_u \cdot x.

U’s private key is D_\text{U}, then the tuple (D_\text{U}, R_\text{U}) is transmitted to each user U via a secure channel. U can validate her private key by checking whether the equation D_\text{U} \cdot P = R_\text{U} + h_u \cdot P_{\text{pub}} holds.

The private key is valid if the equation holds and vice versa.

\text{DelGen:} \text{ This algorithm takes A's secret key D_\text{A} and a warrant m_w as input, and outputs the delegation W_{\text{A} \rightarrow \text{B}} as follows:}

i: \text{Generates a random } a \in \mathbb{Z}_q^{*}, \text{ computes } K = aP.

\text{ii: Computes } e_1 = H_i(m_w, \text{ID}_\text{B}, K) \text{ and } e = e_1 \cdot D_\text{A} + a \mod n.

A sends delegation W_{\text{A} \rightarrow \text{B}} = \{ R_\text{A}, K, m_w, e_1 \} to proxy signer B.

\text{DelVerif: To verify the delegation W_{\text{A} \rightarrow \text{B}}, proxy signer B first computes}

\[ e_1 = H_i(m_w, \text{ID}_\text{B}, K), \quad h_\lambda = H_i(D_\text{A}, R_\text{A}) \text{ then checks whether } \sigma P = e_1(R_\text{A} + h_\lambda P_{\text{pub}}) + K. \]

Accepts if it is equal, otherwise rejects.

\text{PKGen:} \text{ If B accepts the delegation W_{\text{A} \rightarrow \text{B}}, he computes the}

proxy signing key D_\text{B} as:

\[ D_\text{B} = \sigma D_\text{B} \mod m, \text{ where } e_2 = H_i(m_w, \text{ID}_\text{A}, K). \]

\text{PBSign:} \text{ Let m be the message to be signed. Proxy signer B chooses } k' \in \mathbb{Z}_q^{*}, \text{ computes } R' = k' \cdot P \text{ and sends } (R', R_\text{B}, R_\text{A}, K) \text{ to owner C as commitment.} \text{ This algorithm has three following sub-algorithms.}

\text{i. Blind:} \text{ C computes } h_\lambda = H_i(D_\text{B}, R_\text{B}), h_\gamma = H_i(D_\text{A}, R_\text{A}) \text{ and chooses } \alpha, \beta, \gamma \in \mathbb{Z}_q^{*}, \text{ computes } R = \alpha R' + \beta P + \gamma \cdot e_1(R_\text{B} + h_\lambda P_{\text{pub}}) + e_1(R_\text{A} + h_\gamma P_{\text{pub}}) + K.

\[ h = H_2(m, R) \text{ and } h' = \alpha \cdot (h + \gamma) \mod n, \text{ then sends } h \text{ to proxy signer B.} \]

\text{ii. BSign:} \text{ Proxy signer B computes } s' = h \cdot D_\text{B} + k \mod n \text{ and sends } s' \text{ to owner C.} \text{ Thus } (R_\text{A}, R_\text{B}, K, R, m_w, m, s') \text{ is a valid proxy blind signature on message } m.

\text{PBVerif:} \text{ To verify the signature } (R_\text{A}, R_\text{B}, K, R, m_w, m, s) \text{ for message } m, \text{ the verifier first computes}

\[ e_1 = H_i(m_w, \text{ID}_\text{B}, K), h_\lambda = H_i(D_\text{A}, R_\text{A}), e_2 = H_i(m_w, \text{ID}_\text{A}, K), h_\beta = H_2(m, R) \text{ and } h = H_2(m, R) \text{ then checks whether the equation } sP = R + h[e_1(R_\text{B} + h_\lambda P_{\text{pub}}) + e_1(R_\text{A} + h_\gamma P_{\text{pub}}) + K] \text{ holds.} \]

If holds then accepts otherwise rejects it.

\textbf{Correctness:} \text{Since } s' = h \cdot D_\text{B} + k \mod n, \text{ and } s = \alpha s' + \beta \mod n, \text{ we have}

\[ sP = (\alpha s' + \beta)P = \alpha [h \cdot D_\text{B} + k]P + \beta P = \alpha [h \cdot (h + \gamma)D_\text{B} + k]P + \beta P = \alpha kP + (h + \gamma)h_\lambda P_{\text{pub}} + \beta P = \alpha R' + \beta P + (h + \gamma)(e_1(P) + h_\alpha P_{\text{pub}} + c_1(R_\text{A} + h_\lambda P_{\text{pub}}) + K). \]

\textbf{V. Comparative Analysis}

In this section, we will compare the efficiency of our new scheme with schemes \cite{17, 22, 26} and scheme \cite{21}. Since, we do not use even a single pairing operation in our proposed signature construction. In addition, we also avoid the map to point hash function. Both of them are very time taking and expensive cryptographic operations. Therefore we are going to show that our proposed scheme is much efficient while maintaining provable security feature. We just use the running time results of the different cryptographic operations on some cryptographic environment from the paper \cite{11} as an example to show the efficiency comparison.

\textit{Cryptographic Operation Time (in milliseconds)\cite{11}}

<table>
<thead>
<tr>
<th>Operation</th>
<th>Modular exp</th>
<th>OP</th>
<th>MP</th>
<th>ME</th>
<th>HM</th>
<th>General hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>5.31</td>
<td>20.0</td>
<td>6.38</td>
<td>2.21</td>
<td>3.0</td>
<td>&lt;0.001</td>
</tr>
</tbody>
</table>

Where M_E, M_P, M_H, O_p stand for one ECC based scalar multiplication, pairing based scalar multiplication, Map-to-point hash function and pairing operation respectively.

\textbf{Computational cost and running time analysis of our scheme with the schemes \cite{17, 22, 26} and scheme \cite{21} are given in the following table 1 and 2 respectively.}

\textbf{Table I}

\textbf{Computational Cost Comparison}

<table>
<thead>
<tr>
<th>Scheme</th>
<th>PBSign</th>
<th>PBVerif</th>
<th>DelGen</th>
<th>DelVerif</th>
<th>PKGen</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scheme \cite{17}</td>
<td>6M_E + 1H_M + 10O_p</td>
<td>1M_E + 1H_M + 2O_p</td>
<td>1M_E + 1H_M + 20O_p</td>
<td>1M_E + 5H_M + 30O_p</td>
<td></td>
</tr>
<tr>
<td>Scheme \cite{22}</td>
<td>7M_E + 2H_M</td>
<td>1M_E + 2H_M + 20O_p</td>
<td>15M_E + 5H_M + 4O_p</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scheme \cite{26}</td>
<td>6M_E + 2H_M</td>
<td>2M_E + 2H_M + 20O_p</td>
<td>13M_E + 6H_M + 4O_p</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scheme \cite{21}</td>
<td>8M_E</td>
<td>6M_E</td>
<td>21M_E</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Our Scheme</td>
<td>7M_E</td>
<td>6M_E</td>
<td>18M_E</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\textbf{Table II}

\textbf{Running Time Comparison(in ms)}

<table>
<thead>
<tr>
<th>Scheme</th>
<th>PBSign</th>
<th>PBVerif</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scheme \cite{17}</td>
<td>49.50</td>
<td>185.58</td>
<td></td>
</tr>
<tr>
<td>Scheme \cite{22}</td>
<td>49.50</td>
<td>191.06</td>
<td></td>
</tr>
<tr>
<td>Scheme \cite{26}</td>
<td>44.36</td>
<td>181.34</td>
<td></td>
</tr>
<tr>
<td>Scheme \cite{21}</td>
<td>13.26</td>
<td>46.41</td>
<td></td>
</tr>
<tr>
<td>Our Scheme</td>
<td>15.47</td>
<td>39.78</td>
<td></td>
</tr>
</tbody>
</table>

\text{http://dx.doi.org/10.15242/IIE.E0816016}
Above computations show that the running time of PBSSign algorithm of our scheme is 25.21% of the scheme [17], 30.48% of scheme [22] and 34.87% of scheme [26] and as well as 87.50% of scheme [21]. Furthermore total running time of our scheme is 21.43% of scheme [17], 20.82% of the scheme [22], 21.93% of the scheme [26] and as well as 85.7% of scheme [21].

VI. APPLICATION

Proxy blind signature is just like an untraceable signature scheme by which a user can withdraw a valid e-coin from a proxy branch of the bank and spend the coin anonymously for any kind of purchasing at a shop. The user requests the bank to sign blindly a coin using any of the blind signature schemes. Whenever he/she goes to a valid branch to withdraw a coin, needs the proxy branch to make proxy blind signature on behalf of the signee bank (original). It is very important in electronic cash payment system [3,4, 7], anonymous proxy electronic voting etc.

VII. CONCLUSION

In this paper, we have proposed an efficient identity-based proxy blind signature scheme without bilinear pairings. We have also proved the security of the proposed scheme against adaptive chosen message and identity attack under random oracle model. Compared with previous schemes, the new scheme reduces the running time heavily. Therefore, our scheme is more practical than the previous related schemes for practical application.

REFERENCES


http://dx.doi.org/10.15242/IIE.E0816016