Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2013) 21: 945 – 956

doi:10.3906/elk-1102-1051

A blind digital signature scheme using elliptic curve digital signature algorithm

İsmail BÜTÜN[1], **Mehmet DEMİRER**[2,*]

[1]Department of Electrical Engineering, Faculty of Engineering, University of South Florida, Tampa, Florida, USA

[2]Department of Electrical Engineering, Faculty of Engineering, Hacettepe University, Ankara, Turkey

Received: 18.02.2011 • **Accepted:** 27.11.2011 • **Published Online:** 03.06.2013 • **Printed:** 24.06.2013

Abstract: In this study, we propose a blind digital signature (BDS) scheme based on the elliptic curve digital signature algorithm that increases the performance significantly. The security of our scheme is based on the difficulty of the elliptic curve discrete algorithm problem. Therefore, it offers much smaller key lengths for the desired security levels, along with much faster cryptographic processes, leading to fewer hardware and software requirements. According to our simulation results, the relative performance improvement of our proposed BDS scheme is up to 96% when compared with previously proposed schemes.

Key words: Blind digital signature, elliptic curve digital signature algorithm, elliptic curve discrete logarithm problem, digital privacy

1. Introduction and related work

Nowadays people can accomplish their daily tasks, such as banking transactions, without leaving their homes by using the Internet. People also carry out their shopping needs through the Internet, which has increased the growing rate of e-commerce. Now the challenge is to improve the security and anonymity of people in this uncontrollable and dangerous Internet environment. As a solution, we use the concept of the blind digital signature (BDS) presented in [1]. Since 1983, several applications of BDS have been developed through the e-commerce and e-voting fields.

The verification of data using any methods of information technology is called "electronic signature". "Digital signature" is a special kind of electronic signature, which uses public key cryptography (PKC) algorithms to provide data integrity and authenticity. The function of a digital signature is to prevent forged signatures that cannot be distinguished from the original ones and to prevent the modification of the original documents. The aim of a digital signature is to form an electronic basis for the replacement of handwritten signatures. Today, digital signatures are used for identifying the owners of electronic data. The existence of reliable digital signatures that are being used to secure online transactions, such as web-based financial transactions, motivates individuals and corporations to use digital signatures on a widespread basis. Sufficiently reliable digital signature schemes expedite the process of maintaining the obligation of the digital signatures in justice.

Privacy is one of the basic rights for individuals and institutions that need to preserve their confidentiality. BDS is presented as a special application of digital signatures with a distinct property of blindness (unlinkability),

*Correspondence: mehmet@ee.hacettepe.edu.tr

and it can be used in many applications of cryptography where user privacy is important, such as digital voting systems, digital payment systems, online transactions, or electronic government services [1–6].

Digital signature is used for identifying the owners of electronic data, and it thereby assures the traceability of electronic transactions. On the other hand, BDS disguises the content of a message from its signer and thereby assures the privacy of the users. Chaum [1] pioneered the concept of BDS, and he later applied this method to establish customer privacy in electronic payment systems [7]. He attained this by altering the traditional digital signature algorithm (DSA) in an intelligent way.

The BDS scheme involves 2 parties, namely a "signer" and a signature "requester". The scheme allows the requester to have the message signed by the signer without revealing any information about the message. With a secure BDS scheme, the signer is unable to trace the signed message to the previous signing process, where the requester cannot be traced while using the signed message. For example, a bank can sign an electronic coin without seeing its serial number and later cannot distinguish this particular electronic coin from others. Since the customer's transactions cannot be traced, the privacy of the customer is ensured.

BDS is used to provide user anonymity and the unlinkability of electronic transactions, which prevents the signer from linking a blinded message he signed to the unblinded version that he may be asked to verify. The signed blinded message is unblinded prior to verification in such a way that the signature remains valid for the unblinded version of the message. This is a very important feature to ensure user privacy. Hence, BDS schemes can guarantee the anonymity of the customers in secure electronic payment systems [7,8] and the privacy of the voters in secure electronic voting systems [9–11].

Elliptic curve cryptography (ECC) is one of the latest methods of PKC, which is becoming more attractive for researchers. This is because it not only increases the security, but also decreases the resource requirements at the same time. ECC offers the same security level with a shorter key length [12]. It is clear that no one can regret such a property in security systems nowadays, where increasing the security level is a requirement. This is an appealing development, especially for security applications used in resource-limited devices (where "resources" are the processor speed, memory size, and power supplies [i.e. batteries] of said devices), such as smart cards, cell phones, and other similar palm devices. Therefore, the applications that use ECC on such devices will require fewer processor loops, less memory size, smaller key lengths, and less power consumption when compared with the applications using other PKC algorithms such as the Rivest–Shamir–Adleman (RSA) algorithm [13] or the DSA [14].

With growing potential in e-commerce, ECC systems will be considered to be an important alternative solution to ensure security. Since they increase the security level per bit compared to the other traditional PKC algorithms, the required hardware capacity also decreases. This is a property of prime importance in systems where the computing resources are limited. Thus, together with ECC systems, we can generate not only high speed hardware implementations, but also robust systems against any known attacks towards PKC systems.

Elliptic curve DSA (ECDSA) [15] helps us in the application of this cryptography approach (ECC) to digital signatures. It was developed to match the properties of today's standardized DSA [14]. In this paper, in order to generate effective and satisfactory BDSs, we present an efficient method of ECDSA. This work is based on our previous research published in [16].

In this study, we propose a BDS scheme that offers an increased performance in terms of processing time compared to its counterparts. The relative performance improvement of our scheme is up to 96% compared to that in [1] and up to 66% compared to that in [17]. In our proposed scheme, we achieve this by employing ECC (i.e. ECDSA), which is superior to the other cryptography algorithms, such as RSA or DSA.

The rest of the paper is organized as follows: Section 2 provides a brief introduction to the concept of BDS. Section 3 presents our proposed BDS scheme. Section 4 includes the analysis and simulation results to provide a comparison of our proposed scheme to the schemes presented in [1] and [17]. Finally, Section 5 concludes the paper and outlines the future work.

2. Brief introduction to BDS

BDS is a method of signing in such a way that the signer does not see the content of the document. Moreover, if the signer sees the document/sign pair, he cannot determine for whom or when the document has been signed (although it is possible to verify the validity of the signature). This is equivalent to signing a document blindly. We can verify the signature if we see the document and signature, but it is clear that we would need to remember when or for whom we have signed the document. Initially, this concept may be seen as rather strange – why do we need to sign a document without reading it? It has been shown that this concept can efficiently be applied to systems where user privacy is of primary significance. Online voting and electronic payment are very good examples of these systems: when we vote online, we would like to keep secret for whom we voted, just the same as in the case of voting with ballots. When shopping with cash money, a customer does not show an ID to a vendor (in most of the cases, assuming age restrictions are fulfilled). Moreover, the vendor does not recognize the customer, but can tell whether the customer's money is forged or not. In the same manner, through an online transaction, we do not want anybody to learn when or what we have paid for, but a vendor would be able to verify the legitimacy of our payment.

The first BDS scheme that appears in the cryptography literature is based on the factoring algorithm problem proposed by Chaum [1]. According to Chaum's BDS scheme, there are 5 phases of initialization, blinding, signing, unblinding, and verifying, and a BDS scheme must satisfy the following properties:

- **Correctness:** The correctness of the signature of a message signed through the proposed BDS scheme can be checked by anyone using the signer's public key.
- **Blindness:** The content of the message should be blind to the signer.
- **Unforgeability:** The signature is the proof of the signer, and no one else can derive any forged signature and pass it through the verification.
- **Unlinkability:** The signer of the BDS is unable to link the message/signature pair, even when the signature has been revealed to the public.

In [1], a BDS transaction has 2 parties, namely the requester and signer. When a requester requires a BDS of the signer in response to a message, the system blinds the message by multiplying with a blinding factor. Next, the requester sends the blinded message to the signer. The signer signs the blind message using his own private key, and then sends the resultant BDS to the requester. Afterwards, the requester unblinds (extracts) the signer's digital signature from the message by deducting the blinding factor. At the end of the transaction, the requester obtains the signer's signature in the original message without revealing the original message to the signer. The signer's public key can be used for authentication purposes (to authenticate the signature when needed).

3. Proposed BDS scheme

In [17], the proposed BDS scheme was derived from a variation of the DSA. Our BDS scheme, meanwhile, is derived from a variation of the ECDSA, and again it has 5 phases:

- Initialization,
- Blinding,
- Signing,
- Unblinding,
- Verifying.

In our proposed scheme, we used the elliptic curves over the F_p prime field, which has been suggested by the National Institute of Science and Technology (NIST) and is called Federal Information Processing Standard 186-2 [18,19].

According to the Standards for Efficient Cryptography Group [20], elliptic curve domain parameters over F_p are defined as a sextuple:

$$T = (p,\ F_p abGnh), \tag{1}$$

where $p$is an integer specifying theF_p finite field and $ab \in F_p$ are integers specifying the elliptic curve $E(F_p)$ defined by Eq. (2):

$$E\,(F_p):\ y^2 \equiv x^3 + ax + b\ (mod\, p), \tag{2}$$

where $G = (x_G y_G)$is a base point on$E(F_p)$, n is a prime number defining the order of G, and h is an integer defining the cofactor: $h = \# \frac{E(F_p)}{n}$.

3.1. Initialization and key pair generation for ECDSA

The signer defines the elliptic curve domain parameters T, defined as in Eq. (1). Next, for each request, an integer k is randomly selected by the user and the elliptic curve point $\acute{R}$ is calculated accordingly (refer to Table 1 for all abbreviations used in this section):

$$\acute{R} = kG = (x\acute{}_1, y\acute{}_1), \tag{3}$$

$$\acute{r} = x\acute{}_1\ (mod\, n). \tag{4}$$

In addition, the signer checks whether Eq. (5) holds.

$$\acute{r} \neq 0 \tag{5}$$

If the result is true, the signer sends the elliptic curve point $\acute{R}$ to the requester. If the result is false, then the signer selects another k randomly and repeats Eqs. (3) and (4) until he finds an $\acute{r}$ fulfilling Eq. (5).

Table 1. Interpretation of abbreviations used throughout Section 3.

Abbreviation	Interpretation
T	Elliptic curve domain parameters
p	Order of the finite field F_p, integer
F_p	Finite field
a, b	Coefficients defining the elliptic curve
G	Generator point
n	Order of G, a prime number
h	Cofactor, integer
ECC	Elliptic curve cryptography
$ECDSA$	Elliptic curve digital signature algorithm
$H(:)$	Hash value
d	Private key of the signer
Q	Public key of the signer, a point on the elliptic curve
m	Message
$\acute{m}$	Blinded message
s	Signature
$\acute{s}$	Blind signature
r	x coordinate of R
$\acute{r}$	x coordinate of $\acute{R}$
$R, \acute{R}$	Points on the elliptic curve
A, B, k	Random integer numbers
(xy)	Coordinates for the Cartesian system

To generate the private and public key of the signer, the following steps are followed:

1. Integer d is chosen randomly in the range (1, n-1).
2. The elliptic curve point of $Q$is calculated as in Eq. (6):

$$Q = dG = (x_Q, y_Q)\,. \tag{6}$$

With these calculations, the public key of the signer is assigned as point Q and the private key of the signer is assigned as integer d.

3.2. Blinding phase

In order to blind the message m, the owner of message m needs the elliptic curve domain parameters T of the signer; refer to Eq. (1). Blinding is achieved through the following steps, which are shown in Figure 1:

1. Signer sends the elliptic point $\acute{R}$ (refer to Eq. (3)) to the requester, which will be used as the blinding coefficient.
2. Requester calculates $\acute{r}$ from the elliptic point $\acute{R}$, as shown in Eq. (4).
3. Requester randomly chooses integers A and B, which are in the range of (1, n-1).
4. Requester calculates the elliptic point R:

$$R = A\acute{R} + BG = (x_1 y_1). \tag{7}$$

5. Requester calculates r from the elliptic point R, which was given in Eq. (7):

$$r = x_1 \,(mod\, n). \tag{8}$$

6. Requester generates the blinded message $\acute{m}$ and sends it back to the signer for the signing operation:

$$\acute{m} = A\, H\,(m)\; \acute{r}\, r^{-1}\,(mod\, n), \tag{9}$$

where H is the "Hash" function, and in our scheme, we use the SHA-1 [21] algorithm as the hash function.

3.3. Signing phase

After the signer receives the blinded message $\acute{m}$ from the requester, he generates the blind signature $\acute{s}$ by following these steps, which are also shown in Figure 1:

1. Signer calculates $\acute{r}$ from the elliptic point $\acute{R}$, as shown in Eq. (4).
2. The private key of the signer, d, was generated in the initialization phase.
3. k is a random integer that was generated in the initialization phase.
4. $\acute{s}$ is calculated as shown in Eq. (10):

$$\acute{s} = d\acute{r} + k\acute{m}\;\; (mod\, n). \tag{10}$$

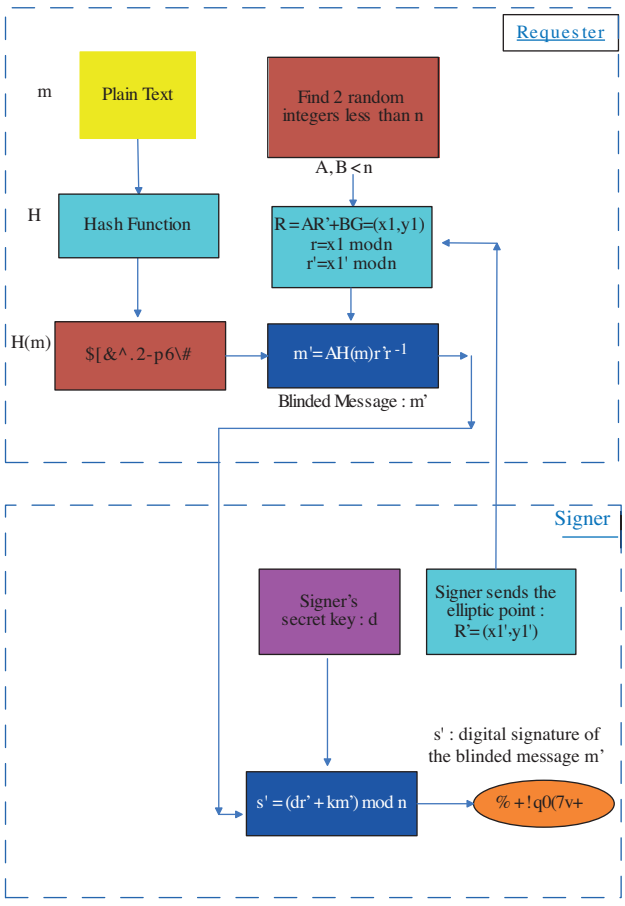

Figure 1. "Blinding" and "Signing" phases of the proposed BDS scheme.

3.4. Unblinding phase

When the requester receives the blind digital signature $\acute{s}$ from the signer, the unblinding operation is needed to obtain the digital signature $(s,\ R)$ on message m, as shown in Figure 2.

1. Requester calculates $\acute{r}$ from the elliptic point $\acute{R}$, as shown in Eq. (4).

2. Requester verifies whether $\acute{r}$ and $\acute{s}$ are in the range of (1, n-1). If so, the requester generates the digital signature $(s,\ R)$ of the signer on message m, as shown in Eq. (11):

$$s = \acute{s}\, r\, \acute{r}^{-1} + BH\,(m)\ (mod\, n). \tag{11}$$

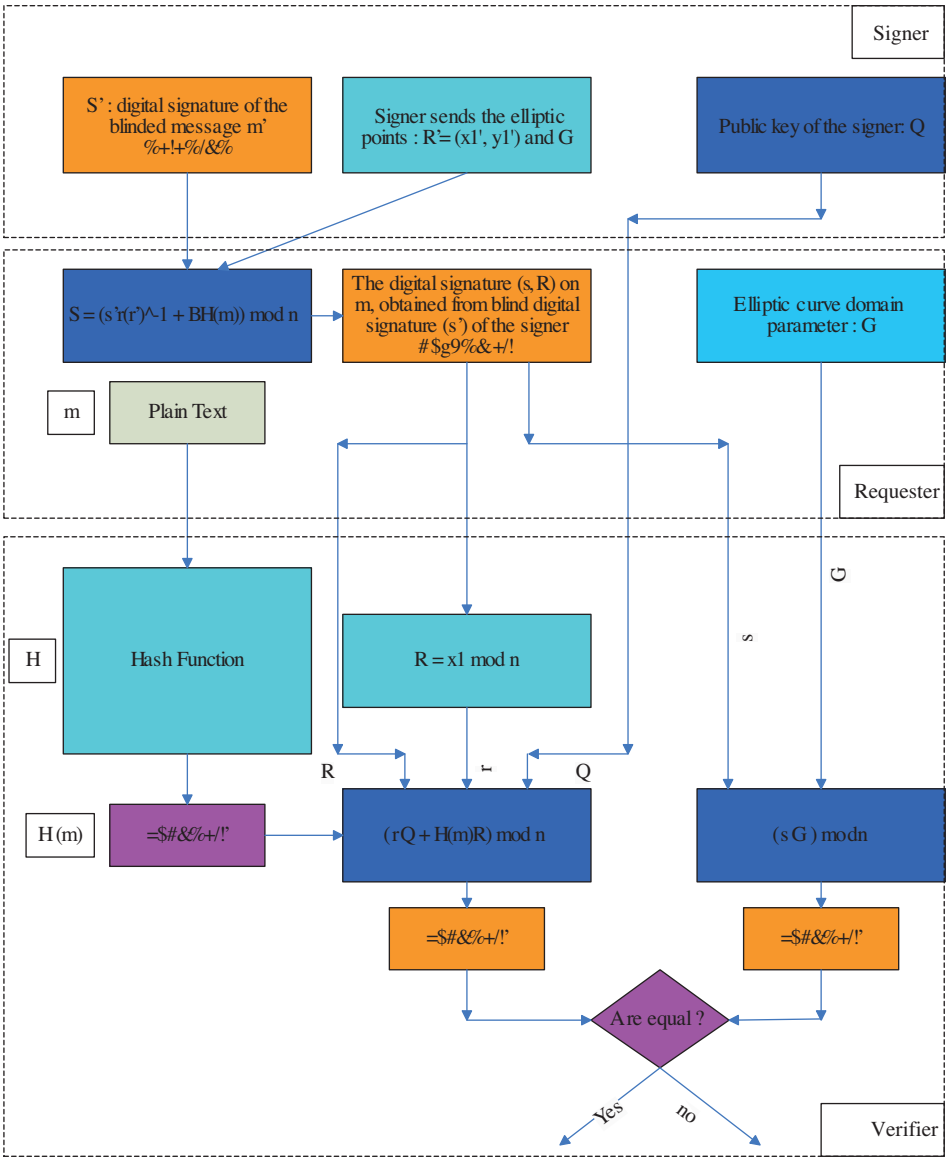

Figure 2. "Unblinding" and "Verifying" phases of the proposed BDS scheme.

3.5. Verifying phase

Any party that has the elliptic domain parameters T of the signer (refer to Eq. (1)) can verify the digital signature of $(s,\ R)$ on message m by following these steps, which are also shown in Figure 2:

$$u_1 = sG\,(mod\,n). \tag{12}$$

u_2 is calculated using the public key of the signer, Q:

$$u_2 = rQ + H\,(m)\,R\ (mod\,n). \tag{13}$$

If the statement of $u_1 = u_2$ is met, then the signature is verified as valid; otherwise, it is considered invalid.

3.6. Correctness proof of the proposed scheme

We begin by expanding u_2, defined in Eq. (13), by substituting Q with dG according to Eq. (6):

$$u_2 = rdG + H\,(m)\,R\ \ (mod\,n). \tag{14}$$

Since from Eq. (7) we know that $R = A\acute{R} + BG$, then we can expand Eq. (14) as follows:

$$u_2 = rdG + H\,(m)\,A\acute{R} + H\,(m)\,BG\ (mod\,n)\,. \tag{15}$$

Using Eq. (3), we substitute $\acute{R}$ with kG and we get:

$$u_2 = rdG + H\,(m)\,AkG + H\,(m)\,BG\ (mod\,n) \quad . \tag{16}$$

Now, by expanding u_1, defined in Eq. (12), we need to achieve the same expression shown in Eq. (16). Since from Eq. (11) we know that $s = \acute{s}\,r\,\acute{r}^{-1} + BH\,(m)\ (mod\,n)$, then Eq. (12) becomes:

$$u_1 = \acute{s}\,r\,\acute{r}^{-1}G + BH\,(m)\,G\,(mod\,n). \tag{17}$$

By substituting $\acute{s}$ with $d\acute{r} + k\acute{m}\ (mod\,n)$ from Eq. (10), Eq. (17) results in:

$$u_1 = d\acute{r}\,r\,\acute{r}^{-1}G + k\acute{m}\,r\,\acute{r}^{-1}G + BH\,(m)\,G\,(mod\,n). \tag{18}$$

By rearranging Eq. (18) we get:

$$u_1 = rdG\acute{r}\acute{r}^{-1} + k\acute{m}\,r\,\acute{r}^{-1}G + H\,(m)\,BG\,(mod\,n). \tag{19}$$

From Eq. (9), substituting $\acute{m}$ with $A\,H\,(m)\ \acute{r}\,r^{-1}\,(mod\,n)$ in Eq. (19) results in:

$$u_1 = rdG\acute{r}\acute{r}^{-1} + k\,A\,H\,(m)\ \acute{r}\,r^{-1}r\,\acute{r}^{-1}G + H\,(m)\,BG\,(mod\,n). \tag{20}$$

From modular arithmetic, we know that $\acute{r}\acute{r}^{-1} = 1\,(mod\,n)$ and $rr^{-1} = 1\,(mod\,n)$. By substituting these into Eq. (20) we get:

$$u_1 = rdG + k\,A\,H\,(m)\,G + H\,(m)\,BG\,(mod\,n). \tag{21}$$

Eq. (21) is the same expression shown in Eq. (16). Therefore, we have proven that $u_1 = u_2$ by showing that Eqs. (16) and (21) are equal to the same expression.

4. Simulation results and discussions

In the applications, the key length of the algorithm is determined according to the desired security level. Today, it is most practical to use a key length of between 160 and 192 bits for ECC systems. In the case of RSA, the key length is 1024 bits for commercial applications and 2048 bits for more critical applications (where more security is needed). These key lengths correspond to the 192-bit and 224-bit ECC key lengths, respectively [22].

While the security of Chaum's BDS scheme [1] is based on the difficulty of the factorization problem [23], the security of Camenisch et al.'s BDS scheme [17] is based on the difficulty of the discrete logarithm problem [24]. On the other hand, the security of our BDS scheme relies on the elliptic curve discrete logarithm problem, which is considered to be much more difficult than either of the other problems [12].

In our work, to provide comparisons for the reader, the implementation and simulation of both the BDS schemes of [1] and [17] have been accomplished. A 1024-bit RSA key length is chosen for the implementation of [1] and a 1024-bit DSA key length is chosen for the implementation of [17]. To provide further comparison for the reader, we have issued our scheme with a variety of NIST-suggested elliptic curves (NIST192, NIST224, NIST256, NIST384, and NIST521). This means that the key length of our scheme changes depending on the curve (192 bits, 224 bits, 256 bits, 384 bits, and 521 bits, respectively). For example, if the NIST192 elliptic curve is chosen for our scheme, then the key length is apparently 192 bits.

The test-bed system consists of a 1733-MHz processor with 512 MB of DDR-2 533-MHz RAM. Implementation is based upon the C programming language. For elliptic curve arithmetic operations, Miracle Library is used [25].

In order to compare the time consumptions of the algorithms, the clock command of the C programming language has been used. It gives the time that is spent on the processor between 2 events. For the same plain text message (m consists of 431 bytes), the time (in seconds) spent on the processor for the relevant algorithms is given in Figure 3.

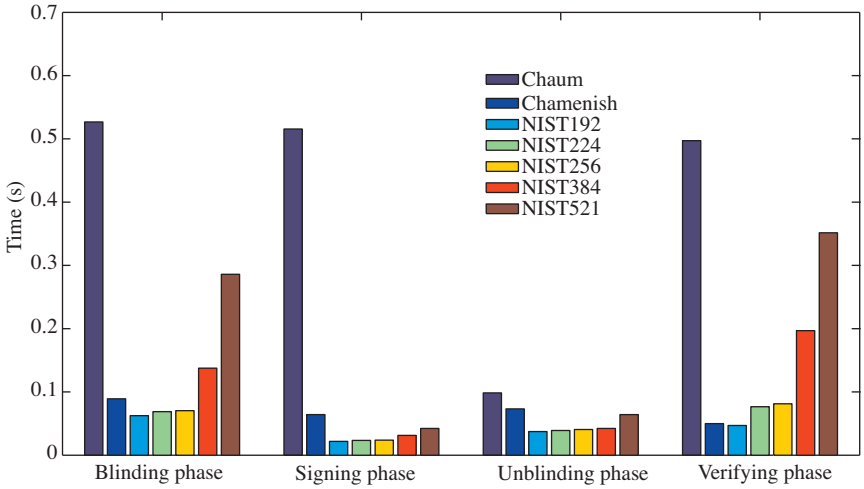

Figure 3. Comparisons of processing time for various BDS schemes classified according to phases.

Figure 3 is sorted according to the phases (blinding, signing, unblinding, and verifying) of the BDS schemes, while Figure 4 is sorted according to the types of the BDS schemes (Chaum's [1], Camenisch et al.'s [17], and our scheme with the following elliptic curves: NIST192, NIST224, NIST256, NIST384, and NIST521).

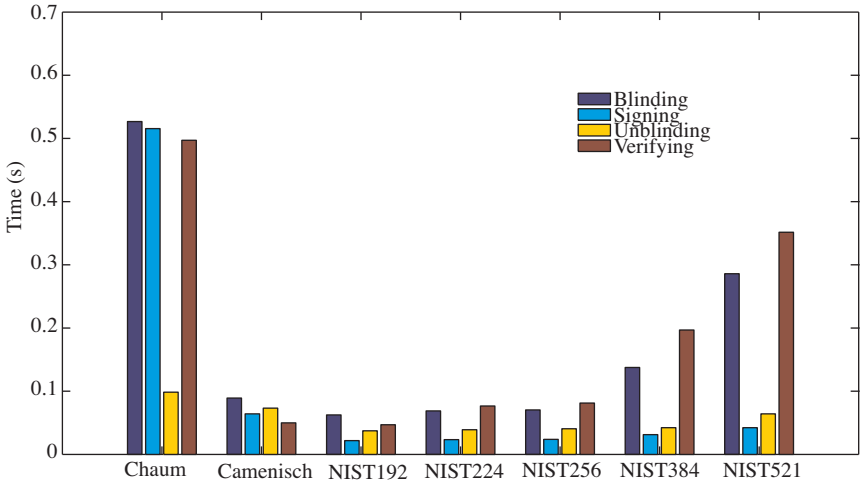

Figure 4. Comparisons of processing times for various BDS schemes classified according to schemes.

Table 2 gives the processing time (s) of our scheme compared to other schemes, when the NIST192 elliptic curve is used for our scheme. Table 3 gives the performance improvement (%) of our scheme compared to other schemes, when the NIST192 elliptic curve is used for our scheme. In this case, it is clear that in terms of the processing time, our scheme outperforms Chaum's scheme [1] by about 96% and Camenisch et al.'s scheme [17] by about 66%.

Table 2. Processing time (s) of BDS schemes.

	Our scheme	Chaum's scheme	Camenisch et al.'s scheme
Blinding phase	0.0624	0.5267	0.0892
Signing phase	0.0218	0.5156	0.0641
Unblinding phase	0.0374	0.0984	0.0732
Verifying phase	0.0470	0.4971	0.0499

Table 3. Relative performance improvement (%) of our scheme compared to other schemes.

	Chaum's scheme	Camenisch et al.'s scheme
Blinding phase	88.15	30.04
Signing phase	95.77	65.99
Unblinding phase	61.99	48.90
Verifying phase	90.55	5.81

For all of the phases (blinding, signing, unblinding, and verifying), the fastest scheme is the one proposed in this study, which uses the NIST192 elliptic curve (in other words, the scheme that has a key length of 192 bits), and the slowest of all is Chaum's [1] scheme, which uses a 1024-bit RSA key length. It is important to mention that the key lengths for the considered schemes are selected to provide equal security levels. For example, it has been proven that the security levels of the 1024-bit key length RSA algorithm, 1024-bit key length DSA algorithm, and 160-bit key length ECC algorithm are the same [26,27]. The computational effort needed to factor a 1024-bit size integer using the general number field sieve method is 3×10^{11} million instructions per second years, whereas the same effort is needed to compute elliptic curve logarithms of the 160-bit size elliptic

point with the Pollard ρ-method [12]. In [12], it is suggested that a 192-bit size NIST elliptic curve is comparable to 1024-bit size RSA and DSA key lengths in terms of the intended cryptanalysis strength. Hence, we issued a 192-bit key length ECC algorithm, and, in this case, our scheme is not only faster but also more secure. Table 4 gives the comparable key sizes of the ECDSA and RSA/DSA algorithms in terms of the computational effort for cryptanalysis [28].

Table 4. Comparable key sizes in terms of the computational effort for cryptanalysis [28].

ECDSA (size of the prime field in bits)	RSA/DSA (modulus size in bits)
112	512
160	1024
224	2048
256	3072
384	7680
512	15,360

5. Conclusions and future remarks

In this study, we briefly introduced the concept of BDS, and later on, our contribution to the field was presented. Our proposed BDS scheme has lower complexity (i.e. in terms of computational load) and provides better security compared to those in [1] and [17].

Our proposed scheme uses ECC (ECDSA), providing all of its advantages over the other PKC algorithms. It offers smaller key lengths for desired security levels, along with high-speed cryptographic processes, leading to low-complexity hardware and software requirements [12]. These advantages are indispensable for applications where resource shortage is of prime importance, especially in mobile platforms. Our proposed scheme can be used in the applications where not only user anonymity but also processing time is critical under certain hardware constraints.

According to the results, our proposed scheme outperforms that in [1] by 96% and that in [17] by 66% in terms of processing time. Thus, our proposed scheme leads to an apparent improvement in BDS systems. Eventually, this enhancement will drastically reduce the total cost of the commercial systems that are using BDS.

The application of our scheme to smart cards, e-commerce, and e-voting is left as future work for us to consider.

Acknowledgments

We would like to give special thanks to our colleagues Dr Murad Khalid, Dr Hasari Çelebi, and Prof Ravi Sankar for their reviews and valuable comments regarding the publication of this work.

References

[1] D. Chaum, "Blind signatures for untraceable payments", Advances in Cryptology: Proceedings of CRYPTO '82, pp. 199–203, 1983.

[2] D. Chaum, "Blind signature system", Advances in Cryptology, p. 153, 1984.

[3] D. Chaum, A. Fiat, M. Naor, "Untraceable electronic cash", Proceedings in Advances in Cryptology, pp. 319–327, 1988.

[4] A. Juels, M. Luby, R. Ostrovsky, "Security of blind digital signatures", Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology, pp. 150–164, 1997.

[5] D. Pointcheval, J. Stern, "Provably secure blind signature schemes", Advances in Cryptology – Proceedings of ASIACRYPT, pp. 252–265, 1996.

[6] Y.C. Lai, M.S. Hwang, "A study on digital blind signature and its applications to electronic voting and electronic cash", MSc, Chaoyang University of Technology, 2002.

[7] D. Chaum, "Privacy protected payments: unconditional payer and/or payee untraceability", In: D. Chaum, I. Schaumuller-Bichl, Eds., Smart Card 2000, Amsterdam, Elsevier, pp. 69–93, 1989.

[8] N. Ferguson, "Single term off-line coins", Advances in Cryptology: Workshop on the Theory and Application of Cryptographic Techniques, pp. 318–328, 1994.

[9] C.I. Fan, C.L. Lei, "A multi-recastable ticket scheme for electronic elections", Advances in Cryptology, Vol. 1163, pp. 116–124, 1996.

[10] W.S. Juang, C.L. Lei, "A collision-free secret ballot protocol for computerized general elections", Computers and Security, Vol. 15, pp. 339–348, 1996.

[11] W.S. Juang, C.L. Lei, "A secure and practical electronic voting scheme for real world environments", IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences, Vol. E80-A, pp. 64–71, 1997.

[12] A. Lenstra, E. Verhuel, "Selecting cryptographic key sizes", Journal of Cryptography, Vol. 14, pp. 255–293, 2001.

[13] R.L. Rivest, A. Shamir, L.M. Adleman, Cryptographic Communications System and Method, US Patent 4,405,829, 1983.

[14] D.W. Kravitz, Digital Signature Algorithm (DSA), US Patent 5,231,668, 1993.

[15] D. Johnson, A. Menezes, S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)", International Journal of Information Security, Vol. 1, pp. 36–63, 2001.

[16] İ. Bütün, "Blind digital signature system development and implementation", MSc, Hacettepe University, Ankara, Turkey, 2006. Available at http://www.eng.usf.edu/∼ibutun/masters/ismail_butun_master_thesis_published.pdf.

[17] J.L. Camenisch, J.M. Piveteau, M.A. Stadler, "Blind signatures based on the discrete logarithm problem", Advances in Cryptology - EUROCRYPT, Vol. 950, pp. 428–432, 1995.

[18] M. Brown, D. Hankerson, J. López, A. Menezes, "Software implementation of the NIST elliptic curves over prime fields", Proceedings of the Cryptographer's Track at RSA Conference, Vol. 2020, pp. 250–265, 2001.

[19] NIST, "Digital Signature Standard", Federal Information Processing Standards Publication 186, 2000.

[20] Standards for Efficient Cryptography Group, "SEC 1: Elliptic Curve Cryptography", available at http://www.secg.org/, accessed October 2011.

[21] D. Eastlake, P. Jones, "US Secure Hash Algorithm 1 (SHA1)", RFC 3174, Internet Engineering Task Force, 2001.

[22] SoftForum, "PKI and Contents Protection", available at www.softforum.co.kr, accessed May 2010.

[23] N. Koblitz, A. Menezes, S. Vanstone, "The state of elliptic curve cryptography", Journal of Designs, Codes and Cryptography, Vol. 19, pp. 173–193, 2000.

[24] S. Pohlig, M. Hellman, "An improved algorithm for computing logarithms over GF(P) and its cryptographic significance (Corresp.)", IEEE Transactions on Information Theory, Vol. 24, pp. 106–110, 1978.

[25] MIRACL Elliptic Curve Library, Shamus Software, available at http://www.shamus.ie/index.php?page=elliptic-curves, accessed May 2010.

[26] M.J. Wiener, "Performance comparison of public-key cryptosystems", CryptoBytes, Vol. 4, pp. 1–5, 1998.

[27] A. Menezes, "Elliptic curve cryptosystems", CryptoBytes, Vol. 1, pp. 1–4, 1995.

[28] W. Stallings, Cryptography and Network Security: Principles and Practice, New Jersey, Prentice Hall, pp. 312–313, 2006.