DOI: 10.31276/VJSTE.64(4).03-09

Implementation of Boneh - Lynn - Shacham short digital signature scheme using Weil bilinear pairing based on supersingular elliptic curves

Nhu-Quynh Luc*, Quang-Trung Do, Manh-Hung Le

Academy of Cryptography Techniques

Received 4 May 2022; accepted 14 July 2022

Abstract:

One option for a digital signature solution for devices with low memory and low bandwidth transmission over channels uses a short digital signature scheme based on Weil bilinear pairing aimed at short processing times, fast computation, and convenient deployment on applications. The computational technique of non-degenerate bilinear pairings uses supersingular elliptic curves over a finite field F_{p^l} (where p is a sufficiently large prime number) and has the advantage of being able to avoid Weil-descent, Menezes-Okamoto-Vanstone (MOV) attacks, and attacks by the Number Field Sieve algorithm. Compared to Elliptic Curve Digital Signature Algorithm (ECDSA) digital signature schemes, generating a digital signature for a Boneh-Lynn-Shacham (BLS) scheme using Weil bilinear pairing on a supersingular elliptic curve is simple. In this study, the authors replace non-degenerate bilinear pairing calculations on a supersingular elliptic curve with a Weil pairing with $P \epsilon E(F_p)$, $Q \epsilon E(F_{p^l})$ and a higher security multiplier $\alpha=12$ in the BLS short digital signature scheme. The execution time of the BLS short digital signature program showed improvement compared to the commercial ECDSA digital signature scheme.

Keywords: **digital signature, ECDSA, elliptic curve cryptography, tate pairing, Weil pairing.**

Classification number: **1.2**

Introduction

Information exchange between devices and applications requires security and authentication with high reliability per the demanding strict standards of this digital era. New requirements for digital signature solutions such as short digital signatures, fast processing speeds, message authentication without transmissions, and digital signature on short message and low bandwidth channel transmissions are essential for today's applications [1-5]. To date, short digital signature solutions and signature authentication using the calculation of an elliptic curve, such as ECDSA, Elliptic Curve-based Schnorr Digital Signature Algorithm (ECSDSA), or Edwards-Curve Digital Signature Algorithm (EdDSA) have been applied widely in commercial products [1, 2, 6-9]. Among these, the digital signature solution with a short digital signature using the calculation of Weil and Tate bilinear pairing of the authors Boneh, Lynn, Schacham (2001) (denoted by the BLS short digital signature scheme) proves to meet the requirements [2, 10].

The BLS scheme uses a special supersingular curve with p=3, which raises the security level of the BLS scheme to be equivalent to the Digital Signature Algorithm (DSA) using a 1024-bit prime number [11-13]. The BLS short digital signature scheme is secure against attack with selected messages (according to a random oracle model), given that "Computational Diffie-Hellman based on an elliptic curve over finite field F_{p^l} (where p is a sufficiently large prime number) being difficult to solve" [1, 2]. The advantage of the BLS scheme when generating a digital signature is its simplicity as both the digital signature and signature verification processes use a non-degenerate bilinear pairing (Weil and Tate bilinear pairings) on the elliptic curve [2, 6, 10, 14-18]. Since this non-degenerate bilinear pairing calculus technique uses a supersingular elliptic curve over finite field F_p, such that both generic discrete log algorithm in $E(F_p)$ and the Number Field Sieve in F_{p^l}* are intractable, it is resistant to some Weil descent and MOV attacks [11, 12], as well as attacks by the Number Field Sieve algorithm [19-21]. Several publications have shown that elliptic curve cryptography (ECC) built on non-degenerate bilinear pairing could be a secure cryptosystem for today's applications with one particular development being the supersingular isogeny Diffie-Hellman (SIDH) [7, 22, 23].

This solution aims towards short processing time, fast computation, and convenient deployment on applications, making it fit for devices with low memory and transmission over low bandwidth channels. The authors have used computational techniques of Weil non-degenerate bilinear pairing (with a higher security multiplier α=12) in building a BLS short digital signature scheme based on a supersingular elliptic curve with functions for key generation, digital signature, and signature verification.

*Corresponding author: Email: quynhln@actvn.edu.vn

Related works on the BLS short digital signatures scheme

Mathematical basis of Weil and Tate pairing based on Supersingular Elliptic curves

Torsion points play an important role in the calculations of Weil and Tate bilinear pairings on elliptic curves and usually torsion points are points of finite order [1, 7].

Definition 1: Given an elliptic curve E over a field K and a positive integer n. Then, the set of n-torsion points is defined as the set $E[n] = \{P \in E(\overline{K}) | nP = \infty\}$ [1].

Since the characteristic of K is not divisible by n, the equation $x^n=1$ does not have multiple solutions, but has n solutions in $\overline{K}$ and μ_n is a cyclic group of order n. An element $\zeta \in \mu_n$ satisfies $\zeta^k=1$ if and only if n is divisible by K, then ζ is called a primitive root of degree n [1].

Definition 2: Let there be an elliptic curve E over K and n be an integer not divisible by the characteristic of K such that $E[n] \subseteq E[K]$. Then, the Weil pairing is the mapping e_n:E[n]×E[n]→μ_n [2].

Given $T \in E$[n], there exists a function f such that $div(f)=n[T]-n[\infty]$. Then choose $T \in E[n^2]$ with $nT=T$, there exists g such that $div(g)=\sum_{R \in E[n]}([T+R]-[R])$. For $S \in E[n], P \in E[\overline{K}]$, then $g(P+S)^n=f[n(P+S)]=f(nP)=g(P)^n$. Thus $\frac{g(P+S)}{g(P)} \in \mu_n$ and $\frac{g(P+S)}{g(P)}$ do not depend on P. Hence, the Weil pairing is $e_n(S,T) = \frac{g(P+S)}{g(P)}$.

Definition 3 [2]: Let p be a prime power, and E/F_p an elliptic curve with m points in $E(F_p)$. Let P in E/F_p be a point of primer order q where $q^2 \nmid m$. We say that the subgroup ⟨P⟩ has a security multiplier α, for some integer α>0, if the order of p in F_q^* is α. In other words:

$q | p^\alpha - 1$ and $q \nmid p^k - 1$ for all $k = 1, 2, \ldots, \alpha - 1$.

The security multiplier of $E(F_p)$ is the security multiplier of the largest prime order subgroup in $E(F_p)$.

Theorem 1 [2, 7, 17, 24]: Let E be an elliptic curve defined over a field F_p. Let n be an integer so that $n|(q-1)$. The elements of $E(F_p)$ of n are denoted by $E(F_p)[n]$ in dividing order, and let $\mu_n=\{x \in F_p | x^n=1\}$. Assume $E(F_p)$ contains an element of order n. Then, there exists a non-degenerate bilinear mapping:

$$\begin{cases} \langle .,. \rangle_n : E(F_p)[n] \times E(F_p)/nE(F_p) \to F_p^\times/(F_p^\times)^n \\ \tau_n : E(F_p)[n] \times E(F_p)/nE(F_p) \to \mu_n \end{cases}$$

The first pairing is called Tate-Lichtenbaum pairing. The second one, τ_n, is called the modified Tate-Lichtenbaum pairing [2, 7, 17, 24]. Each element in $E(F_p)/nE(F_p)$ has the form $Q+nE(F_p)$, so it is usually written as $\langle P,Q \rangle_n$ and $\tau_n(P,Q)$ instead of $\langle P,Q+nE(F_p) \rangle_n$ and $\tau_n(P,Q+nE(F_p))$. Since $F_p^\times$ is a cyclic group of order n, the $\frac{p-1}{n}$ powers of $\langle P,Q \rangle_n$ and $\tau_n(P,Q)$ give an isomorphism $F_p^\times/(F_p^\times)^n \to \mu_n$. Hence

$$\tau_n(P,Q) = \langle P,Q \rangle_n^{\frac{p-1}{n}} \quad (1)$$

Compute the Tate pairing according to Miller's algorithm [3, 7, 17, 24]:

Given an elliptic curve E over F_p; P, Q are points with prime order n and $P,Q \in E(F_p)$. Draw the line n_1 through P and Q, which intersects E at another point called R_1. Draw the vertical line n_2, which is the line connecting R_1 and the point ∞. The line n_2 intersects E at the third point, which is R_2 ($R_2=P+Q$). The lines n_1 and n_2 are functions on E and have a main divisor [2]:

$$\begin{cases} div(n_1) = [P] + [Q] + [R_1] - 3[\infty] \\ div(n_2) = [R_1] + [R_2] - 2[\infty] \end{cases}$$

Divisor [Q']-[S] will be equivalent to D_Q=[Q]-[∞], so S is chosen at random. Calculate gD_P at D_Q, where at each step in the algorithm T_1 is the point obtained by computing mP where m is an integer represented in binary of the binary expansion of n. Calculate f_1 to be the value at [Q']-[S] of the function f satisfying $m([P]-[\infty])=[T_l]-[\infty]+div(f)$. At the end of the algorithm the value reaches $T_l=\infty, f_l=gD_P$. It follows that f_1 is the value at [Q']-[S] of the function gD_P satisfying $m([P]-[\infty])=div(gD_P)$ as required by the definition of the Tate pairing. For $P \in E(F_p), Q \in E(F_{p^l})$ the Tate pairing is calculated according to the formula $\langle P,Q \rangle_n$ and the modified Tate-Lichtenbaum pairing is calculated by formula (1) with powers $(p^l-1)/n$ [1, 3, 7].

Algorithm 1: Miller's algorithm for computation with Tate bilinear pairings [2, 7]

Input: Let the elliptic curve E over the field F_p. Two points P and Q on E are points of order n.

Output: The value f_1 satisfies the definition of a Tate pairing (Theorem 2).

1. Randomly select $S \in E(F_{p^l})$ and calculate $Q'=Q+S \in E(F_{p^l})$.
2. Let $l=[log_2(n)]-1, T_l=P, f_l=1$
3. While $l \geq 1$ do
   - Write equations for the lines n_1 and n_2 with the multiplication of T_1.
   Calculate $T_l=2T_l, f_l=f_l^2((n_1(Q')n_2(S))/(n_2(Q')n_1(S))$
   - If the l^{th} bit of n is 1, then
   write equations for the lines n_1 and n_2 with the addition of points of T_1 and P.
   Calculate $T_l=T_l+P, f_l=f_l^2((n_1(Q')n_2(S))/(n_2(Q')n_1(S))$
   - Decrease l.
4. Return f_1

The input is an elliptic curve E chosen as a supersingular curve E over the field F_p, $p>3$ (the curve E over the field F_p is said to be supersingular if the curve E satisfies $E[P]=[\infty]$); The subgroup $E(F_p)[n]$ has an influence on the computation in Miller's algorithm, so the number of iterations is $[log_2(n)]$ [2, 7]. For Tate pairing, it is necessary to pay attention to the field characteristic of 2,3 and make sure the order of the group $E(F_p)$ is appropriate, so choose the prime number n as the largest prime divisor of the group order $E(F_p)$. In Miller's algorithm, integer n is calculated by Schoof's algorithm and using the point multiplication algorithm kP [1, 4, 16, 25-27].

According to Algorithm 1, calculating the Tate pairing $\langle P,Q \rangle_n$, (with $P \in E(F_p)$, $Q \in E(F_{p^l})$) on security applications, the line coefficients n_i belongs to the subfield of F_p, the finite field is used to calculate the value of f_1 with a large length field. At that time, the attacker who wants to attack the Miller algorithm must solve the problem "The point P to be found belongs to $E(F_p)$ when knowing the public point Q belongs to $E(F_{p^l})$, then finding the point P is more complicated" [2, 23]. Formula $e_n(P,Q) = \frac{\langle P,Q \rangle_n}{\langle Q,P \rangle_n}$ is often used to calculate Weil

pairing [3, 7]. In addition, the Weil pairing is also calculated according to the formula $e_n(P,Q) = \frac{f_P(R)f_Q(P)}{f_P(Q+R)f_Q(\infty)}$ but it is not favourable [1, 3, 7]. So, the Weil pairing is considered as another way of calculating the Tate pairing when the conditions for the Weil pairing occur.

When $P \in E(F_p)$, $Q \in E(F_{p^l})$, both Tate and Weil pairing calculations are time consuming. Therefore, the calculation time for the required Weil pairing takes twice as much as the calculation of the Tate pairing. In this study, the authors have replaced the non-degenerate bilinear pairing calculations on the supersingular elliptic curve with the Weil pairing in the BLS short digital signature scheme. Then, the performance of the BLS short digital signature scheme is evaluated by comparison with the classic ECDSA scheme commonly used today.

Building a BLS short digital signature scheme based on the non-degenerate bilinear pairing of supersingular elliptic curves

The BLS key generation scheme

With the BLS short digital signature scheme, the curve E used is $y^2=x^3+Ax+B \bmod p$. The input for key generation consists of a set of parameters *(A, B, p, q, l, P)* denoted *BTS-BLS* (Table 1) [2]. This parameter set is used by the author for all key generation, digital signatures, and signature verification processes of the BLS short digital signing scheme.

Table 1. Parameter sets used in the BLS short digital signature scheme.

Parameters	Functions
A, B	The coefficients of the supersingular elliptic curve equation
p	Modulo
q	Greatest prime divisor of $\#(E/F_{p^l})$
l	Key length belongs to F_{p^l}
Point $P \in E/F_{3^l}$	Base point with order *q*

In Algorithm 2, the generated key pair consists of the public key *PK* and the private key *SK* in which the public key is the parameter set *PK=(l, q, P, R)* and the private key $SK=x$, with x is a random number belonging to Z_p^* (with a large enough prime p). When generating the key for the BLS short digital signatures scheme, the BLS scheme only uses the *kP* point multiplication algorithm and choses a random number belonging to Z_p^*. This shows that the key generation process for the BLS short digital signatures scheme is efficient and simple.

Algorithm 2: Generate keys for the short digital signature scheme BLS [2, 6]

- Input: Let l, the curve (E/F_{p^l}) and q is the greatest prime divisor of $\#(E/F_{p^l})$, the point P has order q
- Processing steps: Chosen random number $x \in Z_p^*$ and alculate $R \leftarrow xP$
- Output: The public key $PK=(l, q, P, R)$ and the private key $SK=x$

The BLS short digital signature scheme

According to Algorithm 3, the signing process of the BLS short digital signatures scheme also uses the input parameters of the supersingular elliptic curve E on the field F_{p^l}; the parameters of the curve used for digital signature are the number of the corresponding *BTS-BLS* tuple in the key generation scheme for the BLS scheme.

Algorithm 3: The BLS short digital signature [2, 6, 7]

- Input: message $M \in \{0,1\}^*$, private key $SK=x$
- Parameter set: *BTS-BLS*
- Processing steps:

+ Using $MaptoGroup_{h'}$ algorithm [2], map message M to point $P_M=(x_M,y_M) \in \langle P \rangle$ belonging to E/F_{p^l}

+ Calculate $S_M=xP_M$

- Output: signature $\sigma=x_{S_M} \in F_{p^l}$ of the point $S_M=(x_{S_M},y_{S_M})$

In this algorithm, embedding the message M to be signed into a point $P_M=(x_M,y_M) \in E/F_{p^l}$ and using the kP multiplier algorithm to create a signature for the message M is necessary. The message M, before embedding into a point $P_M \in E/F_{p^l}$ will be hashed using a hash function [5]. The mapping of this hash value to a component x_M coordinate of point P_M is accomplished using the $MapToGroup_{h'}$ algorithm [2, 6, 7]. Thus, the process of creating a digital signature of the BLS short digital signature scheme is more complicated than that of the key generation algorithm of the ECDSA scheme [16, 28, 29]. In the BLS short digital signature scheme, the signature generation process requires the use of a cryptographic hash function and the technique of embedding the message into a point of the curve. This keeps the value of the digital signature generated by the BLS short digital signature scheme small.

The BLS signature verification scheme

In Algorithm 4, signature verification of the BLS scheme is done using the same set of input parameters of the curve as above Table 1. To verify the digital signature, first one must check whether the obtained signature belongs to the curve. Secondly, two values of Weil pairings will be computed, as the first one is being calculated from the base point and the digital signature, and the second one from the public key and the message M. If these two values are equal or the inverse of the first value is equal to the second value, then the signature is valid.

Algorithm 4: The BLS signature verification [2, 6, 7]

- Parameter set: *BTS-BLS*
- Input: The public key $PK=(l, q, P, R)$, the message $M \in \{0,1\}^*$, and the signature σ
- Output: The signature σ is valid or invalid
- Processing steps:

Step 1: Check the condition that the signature σ is the coordinates x_{S_M} of the point $S_M=(x_{S_M},y_{S_M}) \in E/F_{p^l}$. If such a point does not exist, the signature is invalid.

Step 2: Calculate $u \leftarrow e[P,\phi(S)]; v \leftarrow e[R,\phi(h(M))]$, where e is a non-degenerate bilinear mapping (Weil pairing) on the curve $E/F_{p^{6l}}$ and $\phi: E \rightarrow E$ is a Frobenius endomorphism.

Step 3 (check condition u, v): If $u=v$ or $u^{-1}=v$, then the signature is valid, otherwise the signature is invalid.

The correctness of the BLS short digital signature verification algorithm (algorithm 4) is confirmed in step 3 of the algorithm, whether the signature is valid or not. Specifically, with (σ, y) and $(\sigma, -y)$ being two points on E/F_{p^l}, where σ is the x coordinate, one of the two

points can be point S_M or can be used to generate digital signatures in the BLS short digital signatures scheme. From $(\sigma,y)=-(\sigma,-y)$ on the curve, then $e(P,\phi(-S))=e(P,\phi(-S))^{-1}$. Therefore, the $u=v$ condition is to check that $(P, R, h(M), S)$ is a Diffie-Hellman tuple, while the $u^{-1}=v$ condition is to check that $(P, R, h(M), -S)$ is a Diffie-Hellman set [6, 7].

Theoretical model to prove the security of the BLS short digital signature scheme

In Ref. [2], a secure proof theory for the BLS short digital signatures scheme was propose. The theoretical model that proves the security of BLS is based on the difficulty level of the Hidden Field Equation (HFE), co-CDH (Computational co-Diffie-Hellman), co-DDH (Decision co-Diffie-Hellman), and GDH (Gap Diffie-Hellman groups) problems. It is shown that when an isomorphism $\psi:G_2\rightarrow G_1$ exists, the short digital signatures scheme BLS is vulnerable to the discrete log problem by MOV attacks [11, 12], and attacks by the Number Field Sieve algorithm [19-21] on the extended field F_{p^l}.

For Co-GDH signatures from elliptic curves [2], the security level of the BLS short digital signatures scheme is equivalent to the difficulty of the co-CDH (Computational co-Diffie-Hellman) problem on (G_1,G_2). In other words, it is the computational requirements of a discrete log in G_1 or the computation of a discrete log in $F^*_{p^l}$. According to [2], when the BLS scheme uses a special supersingular curve with $p=3$, the security level of the BLS scheme is equivalent to DSA using a 1024-bit prime (MOV attack [11-13]. This is a weakness of the BLS short digital signatures scheme when the number p is small. To use the BLS schema in this case, we would have to use a curve $E(F_{3^l})$ where 3^{6l} is much larger than 1024 bits.

In the case of a BLS schema using a non-supersingular curves over fields of high characteristic with the security multiplier α=6, [2] shows that with l=159-bit (Signature size $[log_2q]$ of the BLS scheme) is equivalent to "DLog Security $[log_2p]$ of 158 bits" and "MOV Security $[6log_2q]$ of 954 bits". Signatures using this curve are 168 bits while the best algorithm for co-CDH on $E(F_p)$ requires either (Formula (1) in [2]) a generic discrete log algorithm taking time approximately 2^{83}, or (Formula (2) in [2]) a discrete log in a 1008-bit finite field of large characteristic.

Finally, consider the BLS schema in the case of higher security multipliers (Definition 3). D. Pointcheval, J. Stern (2000) [30] proposed certain Abelian varieties. However, to obtain security comparable to DSA using a 2048-bit prime with α=6, we get signatures size l=342 bits. Then, with α=12, the signature is shorter but the security level is guaranteed (equivalent to 2048-bit discrete-log security) [31]. The result is an n-bit signature where the pairing reduces the discrete log problem to a finite field of size approximately $2^{7.5n}$.

Results and discussion

Architectural design of BLS short digital signature scheme

Figure 1 details the implementation steps of the key generation algorithm of the BLS schema, the diagram shows that the key generation modulo is simply designed using only a random function and multiplication points (kP) on the elliptic curve. The key generation modulo then will generate the private key and the public key, which

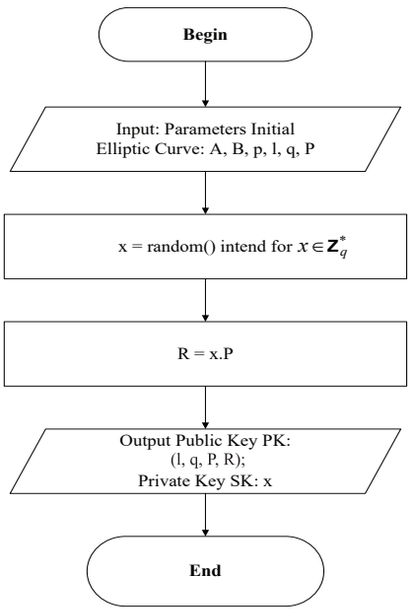

Fig. 1. Scheme of the BLS key generation.

are saved as "bls_private.key" and "bls_public.key," respectively. After executing the key pair generation, the program modulo will issue a notice about the key pair generation time.

Figure 2 shows details the steps of implementing the DSA of BLS schema with a digital signature called "bls_signature.sig". First, when performing a digital signature according to the BLS scheme, the message to be signed, M, will be passed through a secure hashing algorithm that outputs a summary (hash value) [5]. This summary is combined with the private key (the key generated by the BLS key generator modulo), which is then fed into the digital signature program modulo, which results in the digital signature bls_signature. The digital signature program can sign data files of any content with text

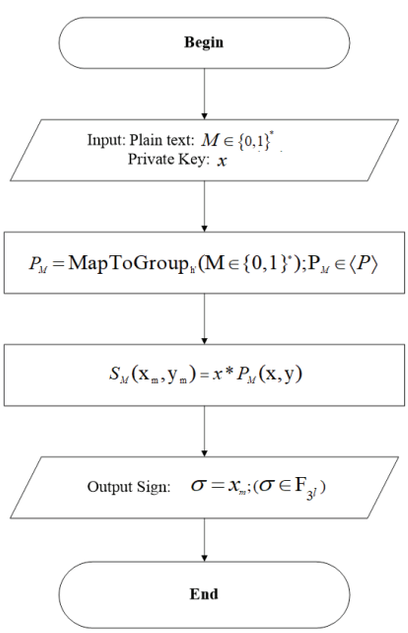

Fig. 2. BLS digital signature scheme.

file formats, image files, audio files, video files, etc. When performing digital signature, the program will create a digital signature file (bsl_signature.sig) and output the execution time of the digital signature process.

Figure 3 details the implementation of the signature verification algorithm steps of the BLS short digital signature scheme. The program verifies the content of the signed data file and calculates the signature verification time of the BLS short digital signature scheme. The received message is passed through the hashing algorithm that obtains the hash value. The process of checking the digital signature of the BLS scheme is done by calculating and checking the input parameters of the hash digest, digital signature, and public key. If the conditions are satisfied, then the signature is valid.

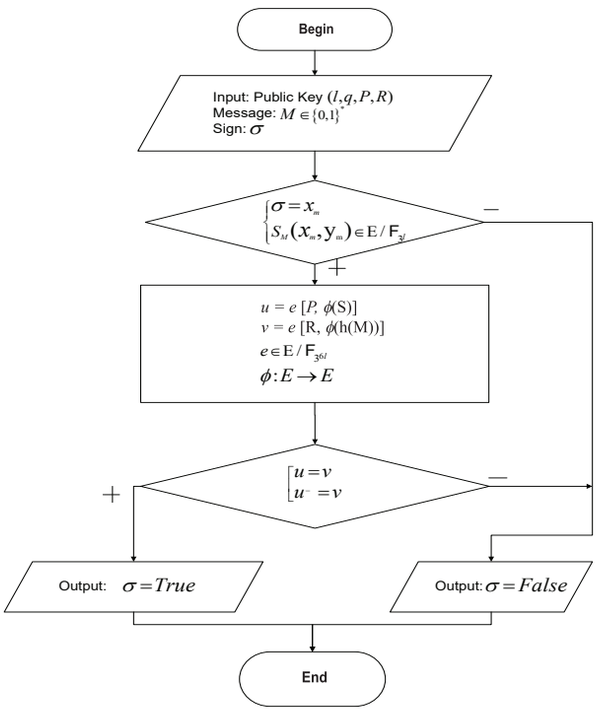

Fig. 3. The BLS signature verification.

Results of the short digital signature program BLS

In this study, the authors have built a program with 3 main modules: key generation, digital signature and signature verification according to BLS scheme. First, the key generation modulo generates a public key and a private key, then the digital signature modulo performs digital signature with the newly generated private key in the key generation modulo. Finally, the signature verification modulo will perform the signature verification with the public key. In addition, in order to facilitate the performance evaluation of the BLS short digital signature scheme, the authors also built a program following the ECDSA digital signature scheme including the key generation module, digital signature module, and signature verification module [16, 28, 29]. Comparisons of key generation, digital signature, and signature verification program against the BLS and ECDSA scheme were executed on the computer using Intel(R) Core i5-4200U, CPU @ 1.60GHz, up to 2.30 GHz; RAM: 4.00 GB.

Based on the security analysis and evaluation for such a BLS scheme, in this study the authors have selected the parameters for the supersingular elliptic curve over finite field F_p such that both a generic discrete log algorithm in $E(F_p)$ and the Number Field Sieve in $F_{p^l}^*$ are intractable, with p=7DDCA613A2E3DDB17 49D0195BB9F14CF44626303, the security multiplier α=12, and signature size l=159. The coefficients of the supersingular elliptic curve are A=-3, B=21C3F3AC7864D1F99273D0F828D3657D8CFD4E ($y^2=x^3+Ax+B$). This parameter set was evaluated by the National Institute of Standards and Technology (NIST, US Department of Commerce), which minimised the risk of being attacked [2, 6, 7, 28].

Table 2 details the execution time of the BLS key generation, digital signature, and signature verification computations. To check the correctness of the program, the authors tested the program with 2 scenarios, specifically:

Table 2. Results of digital signature and signature verification according to the BLS scheme.

Input data	Digital signature time (ms)	Signature verification time (ms)
535 KB	31	98
1.56 MB	119	161
9.47 MB	577	646
9.79 MB	618	671
25.5 MB	1643	1638

Scenario 1: The authors modified the contents of the input data files of the BLS short digital signature program, kept the key and signature, then checked the authenticity of the data. Fig 4 details the process of modifying the input data, where the results showed that the digital signature is invalid and the processing time was given (Fig. 5).

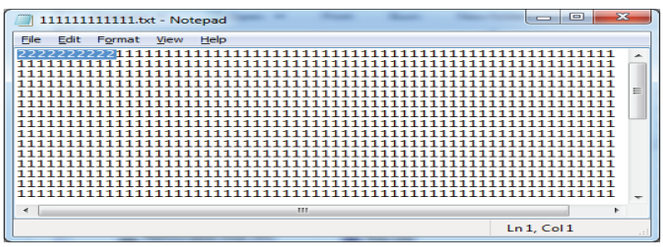

Fig. 4. Modification of the contents of the signed data file.

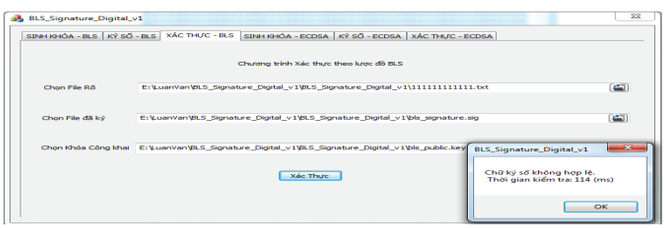

Fig. 5. Signature verification after the message was modified.

Scenario 2: The program generated an original signature (Fig. 6). Then, the author modified the signature (Fig. 7) but did not change the message and the public key. The data verification process for the modified signature resulted in an invalid signature (Fig. 8). Moreover, to evaluate the BLS short digital signature program performance, the

authors tested the digital signature and signature verification program according to the BLS short digital signature scheme with several data files of different lengths (Tables 2, 3).

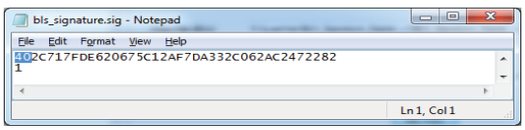

Fig. 6. Original unmodified signature.

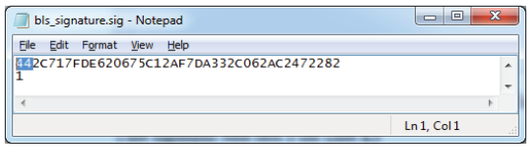

Fig. 7. Signature after modification.

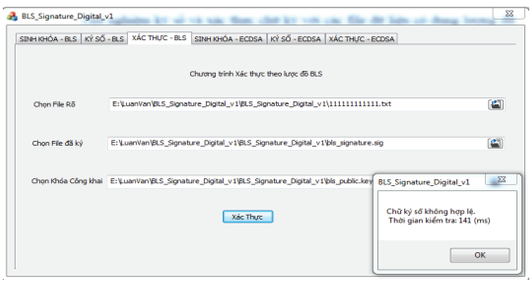

Fig. 8. Signature verification after the signature was modified.

Table 3. Runtime comparison of BLS scheme and ECDSA scheme.

Input data (mb)	Digital signature time (ms)			Signature verification time (ms)		
	BLS	*ECDSA*	*Diff. in %*	*BLS*	*ECDSA*	*Diff. in %*
1.02	108	350	69.14%	166	347	52.16%
1.56	131	523	74.95%	201	523	61.57%
2.00	186	720	74.17%	241	713	66.20%
3.68	298	1227	75.71%	335	1230	72.76%
4.07	313	1353	76.87%	350	1337	73.82%
5.03	376	1637	77.03%	418	1664	74.88%
6.01	450	1928	76.66%	473	1955	75.81%

Analysis and evaluation of the results achieved by the short digital signature program BLS

In previous publications, the authors evaluated the execution speed and occupied resources of the Tate pairing computation and kP point multiplication algorithm on a Spartan6 XC6SLX150T FPGA hardware platform [25, 32].

In this study, the authors tested the execution time of the program under two scenarios. The first was to evaluate the execution speed between the two program functions, i.e., digital signature and signature verification. Second, the authors evaluated the execution speed between the BLS short digital signature program and the ECDSA digital signature program. For each function of the program, the authors ran the test three times and took the average execution time.

Execution speed of digital signature and signature verification BLS: Table 2 details the execution time results of the digital signature modulo and BLS signature validation. Fig. 9 shows the corresponding graph comparing the running time between digital signature and signature verification. Experimental results of the BLS scheme show that the signing time is faster than the validation time. Theoretically, the digital signature of the BLS scheme uses one-point multiplication, while the validation uses two values of the Weil pairing for calculation. In the Weil non-degenerate bilinear pairing values calculation, a point multiplication is used for each value of u and v. Therefore, calculating u, v requires two-point multiplications, which makes the signature verification time longer than the digital signature time.

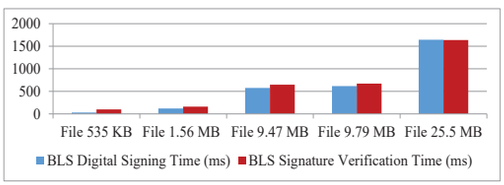

Fig. 9. Digital signature time and signature verification time of the BLS scheme.

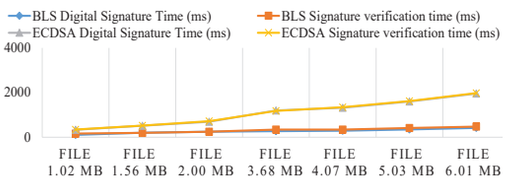

Fig. 10. Runtime comparison of BLS short digital signature and ECDSA schemes.

Execution speed of BLS short digital signature program and ECDSA digital signature program: Both the BLS and ECDSA digital signature schemes are designed with a 160-bit key-length key for the same data input. Table 3 and the diagram in Fig. 10 present the run-time details of the digital signature function for both the BLS and ECDSA short digital signature scheme.

Table 3 shows that the running speed of the BLS scheme's digital signature/signature verification algorithm (with a key length of 160 bits) is better than that of the ECDSA scheme. Specifically, BLS's digital signature generation performs at least 69% faster than that of ECDSA, while the signature verification process of BLS is at least 52% faster than ECDSA.

With the same key length (160 bits), the same digital signature, and signature verification data, the BLS short digital signature scheme had a faster execution time than the ECDSA scheme. Moreover, with the larger size of the input data file, the execution time of the BLS short digital signature scheme linearly increased with the input data file size as shown in Fig. 10. This can be explained by two main reasons:

For digital signature function: The number of operations used for the digital signature function of the BLS schema includes a mapping of a point on the curve and a point multiplication kP. Meanwhile, the number of operations used for the digital signature function of the ECDSA scheme includes one kP point multiplication, one inverse

operator modulo, and two scalar point multiplications. The DSA of the BLS scheme obviously requires less operations than ECDSA digital signature.

For the signature verification function: The number of operations using signature verification for the BLS scheme includes the Weil non-degenerate bilinear pairing value calculation that uses two points multiplications to calculate the two values u and v. Meanwhile, the number of operations used in the signature verification function of the ECDSA digital signing scheme includes one modulo inverse operator, two points multiplications, and two scalar multiplications. The larger number of operations makes the ECDSA scheme operate slower than the BLS scheme.

Conclusions

In this paper, the authors used the calculation technique of Weil non-degenerate bilinear pairing (with $P \in E(F_p)$, $Q \in E(F_{p^l})$ and a higher security multiplier $\alpha=12$) in building a BLS short digital signature scheme based on supersingular elliptic curves with key generation, digital signature, and digital verification functions. The set of supersingular elliptic curve parameters (with a sufficiently large prime p and a higher security multiplier $\alpha=12$) initialised for the selected BLS scheme ensures that the signature size is short and the security of the BLS scheme remains theoretically safe. The execution time of the BLS short digital signature program was much improved compared to the ECDSA digital signature scheme, which makes BLS short digital signature scheme a candidate for applications that require short processing time, fast computation, and for devices with low memory and low bandwidth transmission.

ACKNOWLEDGEMENTS

The authors are grateful to the Academy of Cryptography Techniques for supporting this work.

COMPETING INTERESTS

The authors declare that there is no conflict of interest regarding the publication of this article.

REFERENCES

[1] H. Cohen, et al. (2005), *Handbook of Elliptic and Hyperelliptic Curve Cryptography*, Chapman and Hall/CRC, DOI: 10.1201/9781420034981.

[2] D. Boneh, B. Lynn, H. Shacham (2001), "Short signatures from the weil pairing", *Advances in Cryptology - CRYPTO 2002*, **2248**, pp. 514-532.

[3] S. Wang (2017), *Efficient Computation of Miller's Algorithm in Pairing-Based Cryptography*, Electronic Theses and Dissertations, University of Windsor, 86pp.

[4] M. Masoumi, H. Mahdizadeh (2012), "Efficient hardware implementation of an elliptic curve cryptographic processor over GF(2^163)", *Int. J. Comput. Electr. Autom. Control Inf. Eng. 2012 Int.*, **6(5)**, pp.725-732.

[5] D. Moody, et al. (2015), "Report on pairing-based cryptography", *J. Res. Natl. Inst. Stand. Technol.*, **120**, DOI: 10.6028/jres.120.002.

[6] A. Markel, L. Nemirovskiy (2014), "Pairing-based short signatures", https://markel.co/projects/ecc/2/article.pdf.

[7] V.S. Miller (2004), "The Weil pairing, and its efficient calculation", *J. Cryptol.*, **17**, pp.235-261.

[8] J. Shallit, et al. (1999), "Handbook of applied crytography", *Am. Math. Mon.*, **106(1)**, DOI: 10.2307/2589608.

[9] S.S. Dhanda, B. Singh, P. Jindal (2020), "Lightweight cryptography: A solution to secure IoT", *Wirel. Pers. Commun.*, **112(3)**, pp.1947-1980.

[10] P.S.L.M. Barreto, et al. (2002), "Efficient algorithms for pairing-based cryptosystems", *Advances in Cryptology - CRYPTO 2002*, **2442**, pp.354-369.

[11] A.J. Menezes, T. Okamoto, S.A. Vanstone (1993), "Reducing elliptic curve logarithms to logarithms in a finite field", *IEEE Trans. Inf. Theory*, **39(5)**, pp.1639-1646.

[12] J. Shikata, Y. Zheng, J. Suzuki (2000), "Realizing the Menezes-Okamoto-Vanstone (MOV)", *IECE Trans. Fundam.*, **E83-A(4)**, pp.756-763.

[13] R. Barbulescu, P. Gaudry, A. Joux, E. Thomé (2013), "A quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic", https://arxiv.org/abs/1306.4244.

[14] O. Abid (2012), "New digital signature protocol based on elliptic curves", *Int. J. Cryptogr. Inf. Secur.*, **2(4)**, pp.13-19.

[15] S. Koppula, J. Muthukuru (2016), "Secure digital signature scheme based on elliptic curves for internet of things", *Int. J. Electr. Comput. Eng.*, **6(3)**, DOI: 10.11591/ijece.v6i3.9420.

[16] M.A. Mehrabi, C. Doche, A. Jolfaei (2020), "Elliptic curve cryptography point multiplication core for hardware security module", *IEEE Trans. Comput.*, **69(11)**, pp.1707-1718.

[17] M.H.T. Tran, et al. (2017), "Multilinear mappings based on weil pairing over elliptic curves", *2017 4th NAFOSTED Conference on Information and Computer Science*, DOI: 10.1109/NAFOSTED.2017.8108053.

[18] D.P. Le, C.H. Tan (2013), "Improved Miller's algorithm for computing pairings on edwards curves", *IEEE Trans. Comput.*, **63(10)**, pp.2626-2632.

[19] O. Schirokauer, D. Weber, T. Denny (1996), "Discrete logarithms: The effectiveness of the index calculus method", *International Algorithmic Number Theory Symposium*, **1122**, DOI: 10.1007/3-540-61581-4_66.

[20] R. Padmavathy, C. Bhagvati (2010), "Solving the discrete logarithm problem for ephemeral keys in chang and chang password key exchange protocol", *J. Inf. Process. Syst.*, **6(3)**, pp.335-346.

[21] D. Hankerson, A.J. Menezes, S. Vanstone (2004), *Guide to Elliptic Curve Cryptography*, Springer, 312pp.

[22] D.B. Roy, D. Mukhopadhyay (2019), "High-speed implementation of ECC scalar multiplication in GF(p) for generic montgomery curves", *IEEE Trans. Very Large Scale Integr. Syst.*, **27(7)**, pp.1587-1600.

[23] C. Costello, P. Longa, M. Naehrig (2016), "Efficient algorithms for supersingular isogeny Diffie-Hellman", *Annual International Cryptology Conference*, **9814**, DOI: 10.1007/978-3-662-53018-4_21.

[24] M. Scott (2005), "Computing the tate pairing", *Cryptographers' Track at the RSA Conference*, **3376**, DOI: 10.1007/978-3-540-30574-3_20.

[25] L.N. Quynh, D.V. Son, M.A. Tuan (2017), "Enhancement of implementing cryptographic algorithm in FPGA built-in RFID tag using 128 bit AES and 233 bit kP multitive algorithm", *VNU J. Sci. Math. - Phys.*, **33(2)**, pp.82-87.

[26] I. Yavuz, S.B.Ö. Yalçin, Ç.K. Koç (2008), "FPGA implementation of an elliptic curve cryptosystem over GF(3^m)", *2008 International Conference on Reconfigurable Computing and FPGAs*, DOI: 10.1109/ReConFig.2008.66.

[27] J. López, R. Dahab (1999), "Fast multiplication on elliptic curves over GF(2m) without precomputation", *International Workshop on Cryptographic Hardware and Embedded Systems*, DOI: 10.1007/3-540-48059-5_27.

[28] National Institute of Standards and Technology (2013), *Digital Signature Standard (DSS)*, DOI: 10.6028/NIST.FIPS.186-4.

[29] D. Johnson, A. Menezes, S. Vanstone (2001), "The elliptic curve digital signature algorithm (ECDSA)", *Int. J. Inf. Secur.*, **1(1)**, pp.36-63.

[30] D. Pointcheval, J. Stern (2000), "Security arguments for digital signatures and blind signatures", *J. Cryptol.*, **13(3)**, pp.361-396.

[31] P.S.L.M. Barreto, B. Lynn, M. Scott (2003), "Constructing elliptic curves with prescribed embedding degrees", *International Conference on Security in Communication Networks*, **2576**, DOI: 10.1007/3-540-36413-7_19.

[32] L.N. Quynh, D.V. Son, M.A. Tuan (2019), "Performance of 697-bit Tate pairing based on Elliptic curve implementation for Spartan6 XC6vlx760-2ff1760 FPGA", *4th International Conference on Advanced Materials and Nanotechnology*, pp.166-169.