IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Performance and Security Evaluations of Identity-and Pairing-based Digital Signature Algorithms on Windows, Android, and Linux Platforms: Revisiting the Algorithms of Cha and Cheon, Hess, Barreto, Libert, McCullagh and Quisquater, and Paterson and Schuldt

SHENG ZHONG [1] (Student Member, IEEE), WEI REN [1,2,3] (MEMBER, IEEE), TIANQING ZHU[4,7] (MEMBER, IEEE), YI REN [5] (MEMBER, IEEE), AND KIM-KWANG RAYMOND CHOO [1,6] (SENIOR MEMBER,IEEE).

[1]China University of Geosciences, Wuhan, 430074 P.R. China (e-mail: chongshengcug@qq.com, weirencs@cug.edu.cn)
[2]Hubei Key Laboratory of Intelligent Geo-Information Processing, China University of Geosciences (Wuhan), Wuhan, P.R. China
[3]Guizhou Provincial Key Laboratory of Public Big Data GuiZhou University, Guizhou, P.R. China
[4]School of Software, University of Technology Sydney, Australia (email: tianqing.zhu@uts.edu.au)
[5]School of Computing Science, University of East Anglia, Norwich, NR4 7TJ, UK (email: e.Ren@uea.ac.uk)
[6]Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249-0631, USA
[7]School of Mathematics and Computer Science, Wuhan Polytechnic University, Wuhan 430023, China

The research was financially supported by Major Scientific and Technological Special Project of Guizhou Province (20183001), the Open Funding of Guizhou Provincial Key Laboratory of Public Big Data with No. 2017BDKFJJ006, Open Funding of Hubei Provincial Key Laboratory of Intelligent Geo-Information Processing with No. KLIGIP2016A05, and the National Natural Science Foundation of China under Grant 61502362.

ABSTRACT Bilinear pairing, an essential tool to construct efficient digital signatures, has applications in mobile devices and other applications. One particular research challenge is to design cross platform security protocols (e.g. Windows, Linux, and other popular mobile operating systems) while achieving an optimal security-performance tradeoff. That is, how to choose the right digital signature algorithm, for example on mobile devices while considering the limitations on both computation capacity and battery life. In this study, we examine the security-performance tradeoff of four popular digital signature algorithms, namely: CC (proposed by Cha and Cheon in 2003), Hess (proposed by Hess in 2002), BLMQ (proposed by Barreto, Libert, McCullagh and Quisquater in 2005), and PS (proposed by Paterson and Schuldt in 2006), on various platforms. We empirically evaluate their performance using experiments on Windows, Android, and Linux platforms, and find that BLMQ algorithm has the highest computational efficiency and communication efficiency. We also study their security properties under the random oracle model and assuming the intractability of the CDH problem, we reveal that the BLMQ digital signature scheme satisfies the property of existential unforgeable on adaptively chosen message and ID attack. The efficiency of PS algorithm is lower, but it is secure under the standard model.

INDEX TERMS Identity-based Signature, Pairing-based Signature, Windows, Linux, Android

I. INTRODUCTION

THE U.S. government issued the first electronic signature law in the world on June 20, 2000, and several years later (i.e. April 1, 2005), the government of China officially issued and implemented "electronic signature law of the People's Republic of China", and article 14th explicitly stipulates that "reliable electronic signatures have the same legal effects with handwritten signatures or seals". Electronic signatures are also increasingly commonly used in our society (e.g. online shopping transactions using our mobile devices and applications), with security protocols such as digital schemes playing an important role [1]–[3].

Earlier digital signature algorithms are mostly based on public key cryptography [4]–[7], where a user's public key needs to be validated by a CA (Certificate Authority) [8]–[11]. In 1984, Shamir proposed the concept of the cryptosys-

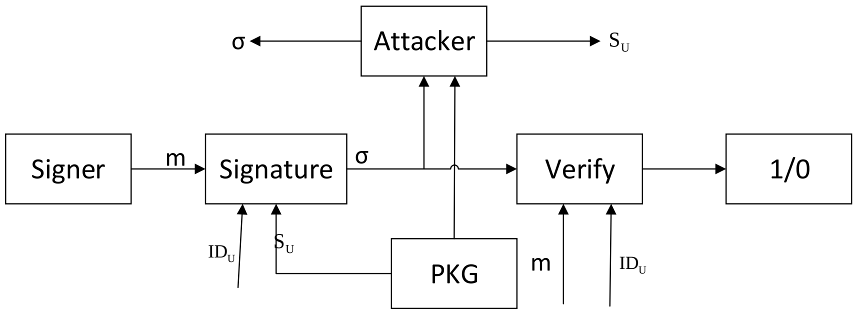

FIGURE 1: A typical identity-based digital signature algorithm

tem based on identity, which simplified key management [12]–[14]. Specifically, it uses bilinear pairing to solve the discrete logarithm problem on elliptic curves. In 2000, Joux proposed bilinear pairing for the identity based cryptosystem [15], and a year later, Boneh proposed a short signature system based on bilinear pairing. Bilinear pairing (e.g. Weil pairing and Tate pairing) allows one to construct more efficient cryptographic protocols. An identity-based signature algorithm generally includes the following four steps, as shown in Figure 1.

1) System set up
   A trusted Private Key Generater (PKG) performs this algorithm, which receives parameter 1^k (k is a safety parameter) as the input. The corresponding output will be the master key s and system parameter $params$. PKG keeps s secret, and makes $params$ publicly available.
2) Key extract
   PKG executes this algorithm to generates the user's key. Given ID_U (i.e. user's identity) as the input, and the PKG calculates the user's private key S_U, and sends it to the user in a secure manner.
3) Sign
   Upon input the system parameter $params$, user's identity ID_U, message m to be signed, and private key S_U; the corresponding output is the signature σ.
4) Verify
   Upon input the system parameter $params$, user's identity ID_U, message m, and signature σ; the corresponding output is 1/0, which represents whether the signature σ is valid for the message m and the user's identity ID_U.

Despite the increasing role of digital signatures in our society and the increasing number of digital signatures proposed in the literature, there has been no prior attempt to objectively examine digital signature algorithms and evaluate them for their suitability for different operating systems (e.g. Windows, Linux and Android), particularly using implementations [16]–[18].

In this research, we evaluate four popular digital signature algorithms, namely: CC (proposed by Cha and Cheon in 2003 [26]), Hess (proposed by Hess in 2002 [27]), BLMQ (proposed by Barreto, Libert, McCullagh and Quisquater in 2005 [28]), and PS (proposed by Paterson and Schuldt in 2006 [29]), using implementations on the PBC Library. Specifically, we measure their computation and communication costs across four different platforms: Windows, Android, and Linux. Then, we evaluate their security.

In the next section, we revisit the four digital signature algorithms.

II. REVISITING THE FOUR ALGORITHMS

A. CC ALGORITHM

The CC digital signature algorithm [26] consists of the following four steps.

1) System set up
   Sets G_1 to be a cyclic addition group generated by p, and the order is p. Sets G_T to be a cyclic multiplication group with the same order p. Sets $e : G_1 \times G_1 \to G_T$ to be a bilinear pairing. Defines two secure Hash functions $H_1 \colon \{0,1\}^* \to G_1$ and $H_2 \colon \{0,1\}^* \times G_1 \to Z_p^*$. PKG generates a master key $s \in Z_p^*$, and calculates $P_{pub} = sP$. PKG opens system parameters $\{G_1, G_T, p, e, P, P_pub, H_1, H_2\}$, and keeps the master key s secret.
2) Key extract
   (1) Gives user U an identity ID_U.
   (2) PFG calculates the user's private key $S_U = sQ_U$. Thus, $Q_U = H_1(ID_U)$ is the user's public key.
3) Sign
   (1) Selects $r \in Z_p^*$ randomly.
   (2) Calculates $V = rQ_U$, $h = H_2(m, V)$ and $W = (r + h)S_U$.
   (3) The signature of the message m is $\sigma = (V, W)$.
4) Verify
   The verifier calculates $h = H_2(m, V)$, and verifies whether the equation $e(P, W) = e(P_pub, V + hQ_U)$ is correct in order to verify if the signature σ is the legitimate signature of message m and the identity ID_U. If the verification is successful, then the signature is valid. Otherwise, the signature is invalid.

B. HESS ALGORITHM

The Hess digital signature algorithm [27] consists of the following steps.

1) System set up
Sets G_1 to be a cyclic addition group generated by p, and the order is p. Sets G_T to be a cyclic multiplication group with the same order p. Sets $e : G_1 \times G_1 \to G_T$ to be a bilinear pairing. Defines two secure Hash functions $H_1 \colon \{0,1\}^* \to G_1$ and $H_2 \colon \{0,1\}^* \times G_1 \to Z_p^*$. PKG generates a master key $s \in Z_p^*$, and calculates $P_{pub} = sP$. PKG opens system parameters $\{G_1$, G_T, p, e, P, P_pub, H_1, $H_2\}$, and keeps the master key s secret.
2) Key extract
(1) Gives user U an identity ID_U.
(2) PKG calculates the user's private key $S_U = sQ_U$. Thus, $Q_U = H_1(ID_U)$ is the user's public key.
3) Sign
(1) Selects $r \in Z_p^*$ and $P_1 \in G_1^*$ randomly.
(2) Calculates $T = (P_1, P)^r$.
(3) Calculates $h = H_2(m, T)$.
(4) Calculates $W = rP_1 + hS_U$.
(5) The signature of the message m is $\sigma = (h, W)$.
4) Verify
The verifier calculates $T = e(W, P)e(Q_U, -P_pub)^h$ to verify if the signature σ is a legitimate signature of the message m and the identity ID_U. After that, determines whether the equation $h = H_2(m, T)$ is correct. If yes, then the signature σ is valid, and outputs 1. Otherwise, the signature σ is invalid, and outputs 0.

C. BLMQ ALGORITHM

The BLMQ digital signature algorithm [28] consists of the following four steps.

1) System set up
Sets G_1 to be a cyclic addition group generated by $p > 2^k$(k is a safe parameter.), and the order is p. Sets G_T to be a cyclic multiplication group with the same order p. Sets $e : G_1 \times G_1 \to G_T$ to be a bilinear pairing. Defines two secure Hash functions $H_1 \colon \{0,1\}^* \to Z_p$ and $H_2 \colon \{0,1\}^* \times G_T \to Z_p^*$. PKG generates a master key $s \in Z_p^*$, and calculates $P_{pub} = sP$. Sets $g = e(p, p)$. PKG opens system parameters $\{G_1$, G_T, p, e, P, P_{pub}, g, H_1, $H_2\}$, and keeps the master key s secret.
2) Key extract
(1) Gives user U an identity ID_U.
(2) PKG calculates user's private key $S_U = \frac{1}{H_1(ID_U)+s}P$.
3) Sign
(1) Selects $r \in Z_p^*$ randomly.
(2) Calculates $x = g^r$.
(3) Calculates $h = H_2(m, x)$ and $V = (r + h)S_U$.
(4) The signature of the message m is $\sigma = (h, V)$.
4) Verify
To verify if the signature σ is a valid signature of the message m and the identity ID_U, the verifier checks if

$$h = H_2(m, e(V, H_1(ID_U)P + P_{pub})g^{-h})$$

is correct. If yes, then it is determined that $sigma$ is valid and 1 is given as the output; otherwise, the signature is invalid.

D. PS ALGORITHM

The PS algorithm [29] is the result of modifying Waters' encryption scheme. In the PS system, the length of the identity and the message is n_u bits and n_m bits. In order to construct a more flexible signature system, we can use two Hash function $H_1 \colon \{0,1\}^* \to (0,1)^{n_u}$ and $H_2 \colon \{0,1\}^* \to (0,1)^{n_m}$. Map any length of identity and message to a bit string with specified length. The PS digital signature system consists of the following four steps.

1) System set up
Sets G_1 to be a cyclic addition group generated by P, and the order is p. Sets G_T to be a cyclic multiplication group with the same order p. $e : G_1 \times G_1 \to G_T$ is a bilinear pairing. PKG selects a master key $s \in Z_p^*$ randomly. Calculates $P_1 = sP$. PKG selects $P_2 \in G_1$, $u', m' \in G_1$, vector $U = (u_i)$ and $M = (m_i)$ randomly. The lengths of U and M are respectively n_u and n_m. PKG opens system parameters $\{G_1$, G_T, n_u, n_m, p, e, P,P_1, P_2, u', U, m', $M\}$, and keeps the master key s secret.
2) Key extract
Gives the user U an identity ID_U. $u[i]$ represents the i^{th} bit of ID_U. $\mu \in \{1, \cdots, n_u\}$ is a set of i that satisfies $u[i] = 1$. In other words, if the i^{th} bit of ID_U is 1, then add i to the set μ. Otherwise, do nothing. PKG chooses $r \in Z_p$ randomly, and calculates U's private key $S_U = (S_1, S_2) = (sP_2 + r(u' + \sum_{i \in u} u_i), rP)$.
3) Sign
$m[j]$ represents the j^{th} bit of the messages m. $M \subseteq \{1, \cdots, n_m\}$ represents a set of j that satisfies $m[j] = 1$. To sign a message m, the signer first selects $r' = Z_p$ randomly, and then calculates $\sigma = (V, W, Z) = (sP_2 + r(r' + \sum_{i \in u} u_i) + r'(m' + \sum_{j \in M} m_j), rP, r'P)$.
4) Verify
To verify whether the signature σ is a valid signature of message m and the identity ID_U, the verifier checks if

$$e(V, P) = e(P_2, P_1)e(u' + \sum_{i \in u} u_i, W)e(m' + \sum_{j \in M} m_j, Z)$$

holds. If yes, then σ is valid and 1 is given as the output; otherwise, the signature is not validate and outputs 0.

III. PERFORMANCE AND SECURITY ANALYSIS

In this section, we analyze the performance and security of the four digital signature algorithms.

TABLE 1: Performance Cost Comparison

System	Calculation Costs										Communication Costs
	Sign					Verify					
	Add	PM	Exp	Mul	P	Add	PM	Exp	Mul	P	
CC	0	2	0	0	0	1	1	0	0	2	$2\|G_1\|$
Hess	1	2	1	0	1	0	0	1	1	2	$\|G_1\| + \|Z_p\|$
BLMQ	0	1	1	0	0	1	1	1	1	1	$\|G_1\| + \|Z_p\|$
PS	$\frac{n_m}{2}+1$	2	0	0	0	$\frac{n_u+n_m}{2}$	0	0	2	4	$3\|G_1\|$

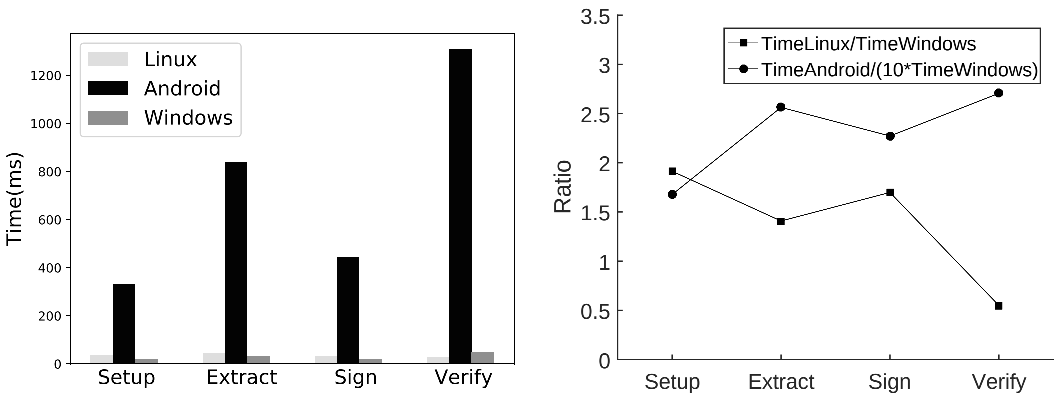

FIGURE 2: Performance of CC under Windows, Android, and Linux

A. PERFORMANCE COMPARISON: COMMUNICATION COST

1) CC digital signature algorithm
At the signing phase, it performs two point multiplication operations in G_1. At the verify phase, it performs one add operation and one point multiplication operation in G_1, and two pairing operations. Thus, the communication cost is $2|G_1|$.
2) Hess digital signature algorithm
At the signing phase, it performs one add operation and two point multiplication operations in G_1, one exponential operation in G_T and one pairing operation. At the verify phase, it performs one exponential operation and one point multiplication operation in G_T and two pairing operations. Thus, the communication cost is $|G_1| + |Z_p|$.
3) BLMQ digital signature algorithm
At the signing phase, it performs one point multiplication operation in G_1 and one exponential operation in G_T. At the verify phase, it performs one add operation, one point multiplication operation in G_1, one exponential operation and point multiplication operation in G_T and one pairing operation. Thus, the communication cost is $|G_1| + |Z_p|$, which is the same as of Hess algorithm.
4) PS digital signature algorithm At the sign phase, there are $\frac{n_u+n_m}{2}+1$ times add operations and two point multiplication operations in G_1 because there is no limit to the numbers of users' ID and the messages to be signed. At the verify phase, there are $\frac{n_u+n_m}{2}$ times add operations in G_1, two multiplication operations in G_T and four pairing operations. Thus, the communication cost is $3|G_1|$.

Table 1 gives the summary of theoretic performance analysis of four algorithms.

B. PERFORMANCE ACROSS THREE PLATFORMS

Figures 2 to 5 present the performance comparison of the four algorithms under three different operating systems, namely: Windows, Android, and Linux.

From the figures, it is clear that the efficiency of the four algorithms on Android is much lower than that on Windows. There are two main reasons for this. One is that Android on mobile is less efficient than Windows on a conventional and more powerful computer, and the other reason is that Java language is less efficient than C language.

1) We also remark that the CPU of an Android (mobile) device generally requires lower power consumption, and the ARM CPU is often used. Thus, the number of instruction sets inside an ARM CPU is less than a computer with an Intel X86 CPU. Also, even for CPUs with the same frequency, there are differences in the performance capability between different floating point operations, ranging from thousands to tens of thousands of times. An Android device's GPU is usually integrated with the CPU at the same SoC (system on chip), which is equal to Intel HD Graphics. The computers that we used for the evaluations have discrete graphics. While both mobile devices and computers are both multi-core, multiple CPUs on mobile devices are used to deal with different things and a computer's multi-core processor refers to centralized multiple core computations on a CPU, and deals with the same thing

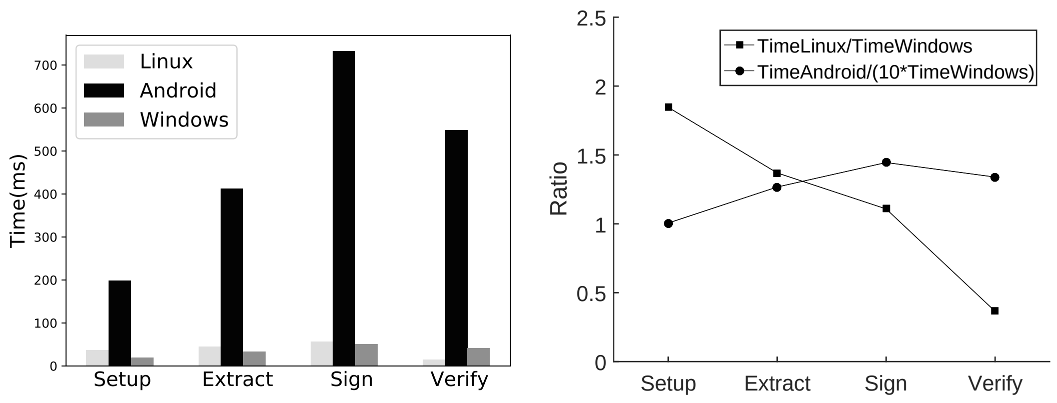

FIGURE 3: Performance of Hess under Windows, Android, and Linux

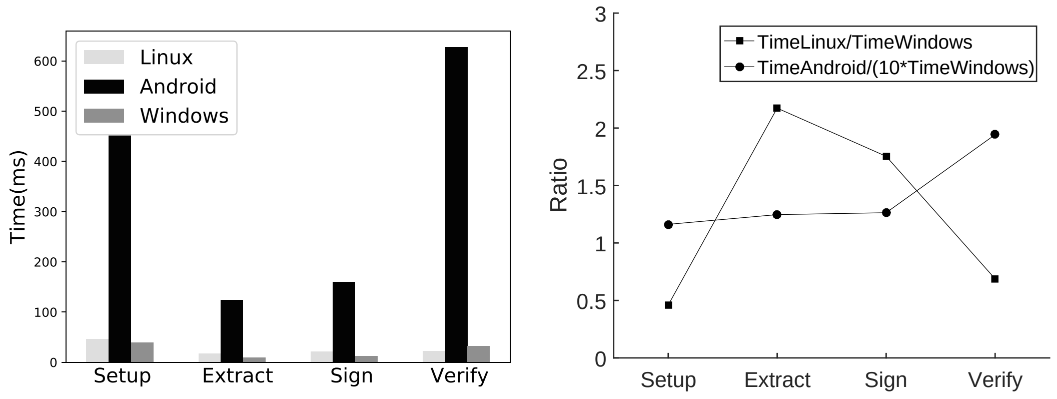

FIGURE 4: Performance of BLMQ under Windows, Android, and Linux

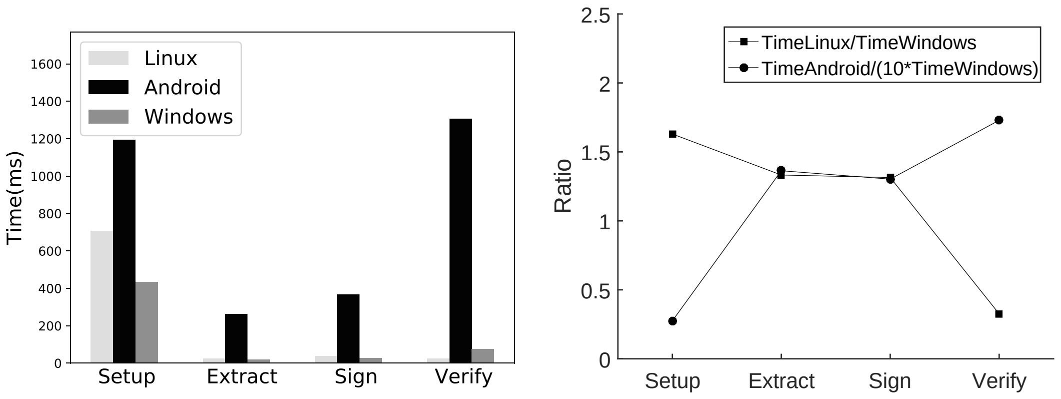

FIGURE 5: Performance of PS under Windows, Android, and Linux

by cooperating with each other (i.e. parallel processing).

2) C language is compiler language. At compile time, the file written in compiler language is compiled into machine language, which is time consuming. However it is very fast when it is running. Java, on the other hand, is an interpretive language. At compile time, it transforms the file written in Java into Java byte code. Java byte code is performed by Java virtual machine (JVM). However, JVM is implemented in C language, which means that there is another intermediate layer. Interpretation of a program is very slow. Sometimes, the high-level language's interpretation of a source program is 100 times slower than the machine code program. In addition, some mechanisms of JVM are time consuming (e.g. garbage collection, and library search and loading). However, in our evaluations, we did not include the compilation process.

We also observed that the first three phases of the four algorithms on Linux require more time than those on Windows.

Linux and Windows are both full multitasking operating systems. But as we know, Linux is faster than Windows. When Windows is installed, it also installs many other components and services; thus, the system becomes bloated.

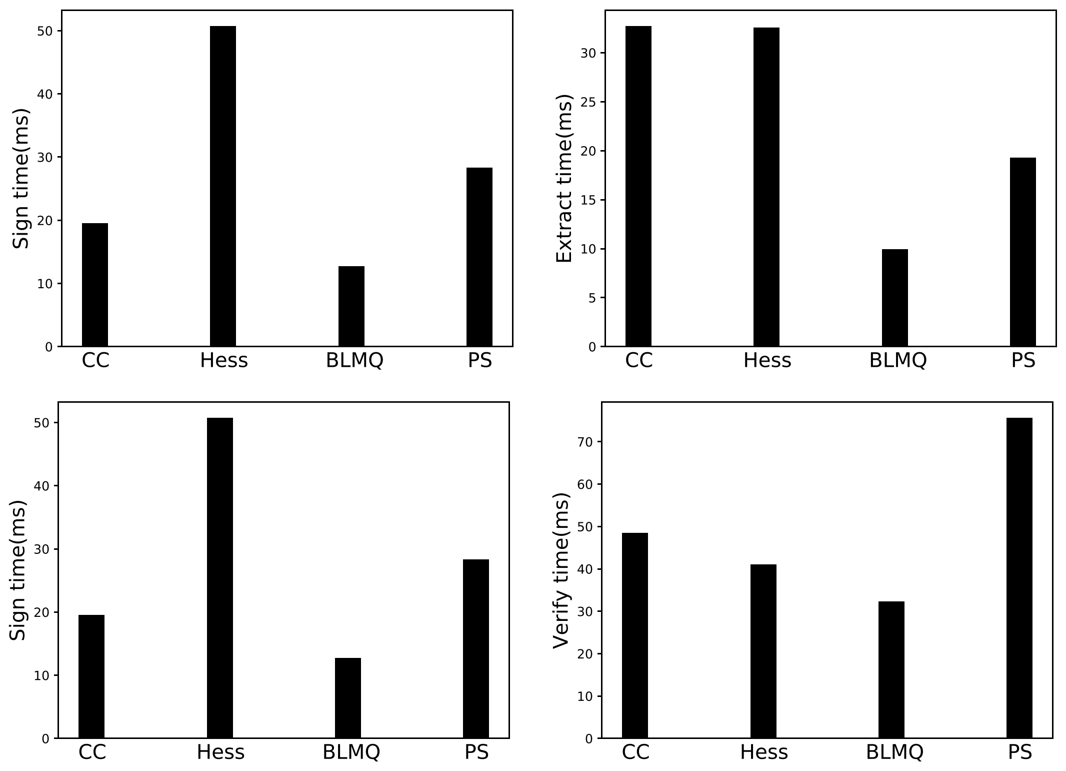

FIGURE 6: Performance of the four signature algorithms on each phase

Whereas when Linux is installed, it only installs basic software and services. When running the same program, the CPU usage of Linux is higher than that of Windows. So under the same hardware conditions, Linux is more efficient than Windows. But as shown in Figures 2 to 5, the efficiency of Linux on the first three phases of the four algorithms is lower than that of Windows. It is due to the need to use the rand function. On Windows, the maximum value that can be returned by the rand function is 32767. However on Linux, the largest number that can be returned by the rand function is 2147483647. In other words, the number of their internal loops and the magnitude of their calculations are different.

Now let's analyze the differences of the four algorithms in the same phase, say on Windows. Figure 6 presents the performance of the four signature algorithms on each phase. We can see that the PS algorithm requires a much longer time than the other three algorithms on the setup phase, with CC, Hess and BLMQ have almost the same time cost. On the extract phase, both CC and Hess algorithms require the most time cost, and the BLMQ algorithms have the lowest time cost. On the sign phase, BLMQ has the lowest time cost. On the verify phase, PS has the most time cost. In general, the PS algorithm has the lowest efficiency, mainly due to the time costs incurred during Setup and Verify phases. BLMQ appears to have the highest efficiency on all phases.

As for communication efficiency, it is system-independent and only relevant to algorithms. Figure 7 shows the communication costs for each algorithm, which suggests that both BLMQ and Hess algorithms have the lowest communication costs.

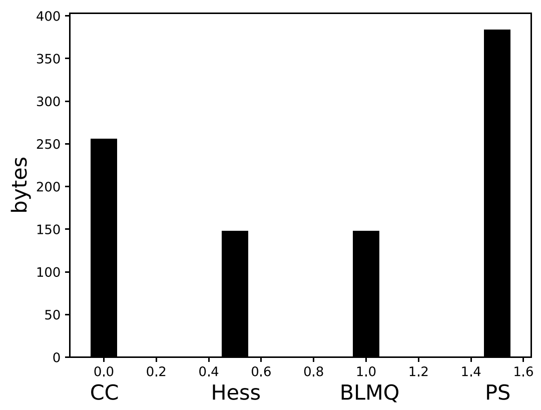

FIGURE 7: Communication cost comparison of the four signature algorithms

C. SECURITY ANALYSIS

In the random oracle model and assuming intractability of the CDH problem, it is proven that the CC digital signature algorithm satisfies the property of existential unforgeable on adaptively chosen message and ID attack.

From the key extract stage, we know that the user's private key is generated by PKG and sent to the user. In other words, PKG is fully aware of the user's private key. So in many cases

PKG is assumed to be fully secure and trusted, by default, which may not be realistic for all applications. Thus, to introduce an additional layer of security, the user can randomly select a secret value $t \in Z_p^*$ in the key extract stage. After that, the user calculates $T = t \times P_{pub}$ and $Q_U = H_1(ID_U, t)$, and then sends T and ID_U to PKG. PKG simply determines the uniqueness of the user's identity information ID_U. If ID_U is unique, then PKG should calculate the private key for the user, and sends the private key to the user. Users no longer need to select random number $r \in Z_P^*$, but calculate $V = t \times P_{pub}$, $h = H_2(m, V)$ and $W = (t + hS_U)$ directly. When verifying, verifiers only need to verify that $e(P, W) = e(V + hP_{pub}, Q_U)$ holds.

Similarly, it was proven that both Hess and BLMQ algorithms satisfy the property of existential unforgeable on adaptively chosen message and ID attack, in the random oracle model and assuming intractability of the CDH problem.

The PS digital signature algorithm was shown to satisfy the property of existential unforgeable on adaptively chosen message and ID attack. However, unlike the previous three algorithms, PS was proven secure under the standard model.

IV. CONCLUSION

In this research, we studied the performance of four popular digital signature algorithms, and in particular their performance on Windows, Linux, and Android, as well as their security.

Future research includes extending this study to cover a more comprehensive list of digital signature algorithms.

REFERENCES

[1] J. Heckeroth, CD. Boywitt Initial exploration of handwritten electronic signatures suitability for the examination of authenticity[J]. orensic Science International 275: 144-154, (2017).

[2] A. Srivastava Electronic signatures and security issues: An empirical study[J]. Computer Law and Security Review: The International Journal of Technology and Practice 25(5): 432-446, (2009).

[3] GG. Kuniholm Steering a course for risk assessment: The impact of new draft guidance, 21 CFR part 11, scope and application, on the electronic records; electronic signatures rule[J]. The Quality Assurance Journal 7(3): 182-186, (2003).

[4] HS. Lee, S. Lim An efficient incomparable public key encryption scheme[J]. Information Sciences 181(14): 3066-3072, (2011).

[5] CK. Li, DS. Wong Signcryption from randomness recoverable public key encryption[J]. Information Sciences 180(4): 549-559, (2010).

[6] K. Singh, CP. Rangan, AK. Banerjee Lattice-based identity-based resplittable threshold public key encryption scheme[J]. International Journal of Computer Mathematics 93(2): 289-307, (2016).

[7] KS. Booth Authentication of signatures using public key encryption[J]. Communications of the ACM 24(11):772-774, (1981).

[8] M. Spalding Deciding Whether or not to use a Third Party Certificate Authority[J]. Network Security 2000(6): 7-8, (2000).

[9] S. Ray, GP. Biswas A Certificate Authority (CA)-based cryptographic solution for HIPAA privacy/security regulations[J]. Journal of King Saud University - Computer and Information Sciences 26(2): 170-180, (2014).

[10] A. Rajaram, S. Palaniswami A High Certificate Authority Scheme for Authentication in Mobile Ad hoc Networks[J]. International Journal of Computer Science Issues (IJCSI) 7(4): 291-298, (2010).

[11] R. Rivest, A. Shamir, LM. Adleman A method for obtaining digital signatures and public key cryptosystems[J]. Communications of ACM 21(2): 120-126, (1978).

[12] A. Shamir Identity-based cryptosystems and signature schemes[C]. Advances in Cryptology-CRYPTO' 84, LNCS 196, Springer-Verlag, Berlin 21(2): 47-53, (1984).

[13] A. Fiat, A. Shamir How to prove yourself: Practical solutions to identification and signature problems[C]. Advances in Cryptology-CRYPTO' 86. LNCS263, Springer-Verlag Berlin 263(1): 186-194, (1986).

[14] CC. Chang, CH. Lin An ID-based signature scheme based upon Rabin's public key cryptosystem. Proceedins 25th Annual [J]. IEEE International Camahan Conference on Security Technology 17(9): 139-141, (1991).

[15] A. Joux A one round protocol for tripartite Diffie-Hellman. Algorithmic Number Theory Symposium [J]. AnTS-IV, LNCS 1838, Springer-Verlag, Berlin 17(4): 263-276, (2004).

[16] P. Karimaghaee, N. Noroozi Frequency Weighted Discrete-Time Controller Order Reduction Using Bilinear Transformation[J]. Journal of Electrical Engineering 62(1): 44-48, (2011).

[17] S. Islam, G. Biswas An Efficient and Provably-secure Digital signature Scheme based on Elliptic Curve Bilinear Pairings[J]. Theoretical and Applied Informatics 24(2): 109-118, (2012).

[18] A. Bowers Representation of Extendible Bilinear Forms[J]. Mathematica Slovaca 65(5): 1123-1136, (2015).

[19] ML. Das Key-escrow free multi-signature scheme using bilinear pairings[J]. Groups Complexity Cryptology 7(1): 47-57, (2015).

[20] SH Islam, GP. Biswas Certificateless short sequential and broadcast multisignature schemes using elliptic curve bilinear pairings[J]. Journal of King Saud University - Computer and Information Sciences 26(1): 89-97, (2013).

[21] S. Kwon An identity-based strongly unforgeable signature without random oracles from bilinear pairings[J]. Information Sciences 276: 1-9, (2014).

[22] KG. Paterson, G. Price A comparison between traditional public key infrastructures and identity-based cryptography[J]. Information Security Technical Report 8(3): 57-72, (2003).

[23] CH. Li, XF. Zhang, H. Jin, W. Xiang E-passport EAC scheme based on Identity-Based Cryptography[J]. Information Processing Letters 111(1): 26-30, (2010).

[24] Y. Mu, Z, Chen, F. Guo Multi-identity management for identity-based cryptography[J]. Journal of Discrete Mathematical Sciences and Cryptography 11(6): 639-672, (2008).

[25] M. Joye, G. Neven, A. Joux Introduction to Identity-Based Cryptography[J]. Cryptology and Information Security Series 2:1-12, (2009).

[26] JC. Cha, JH. Cheon An identity-based signature from Gap Diffie-Hellman groups[C]. Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography-PKC 2567: 18-30, (2003).

[27] F. Hess Efficient Identity Based Signature Schemes Based on Pairings[C]. Springer Berlin Heidelberg 2595: 310-324, (2002).

[28] PSLM. Barreto, B. Libert, N. McCullagh, JJ. Quisquater Efficient and procably-secure identity-based signatures and signcryption from bilinear maps[C]. Springer Berlin Heidelberg 3788: 515-532, (2005).

[29] KG. Paterson, JCN. Schuldt Efficient identity-based signatures secure in the standard model. Information Security and Privacy-ACISP 207-222, (2006).

SHENG CHONG is a master student at School of Computer Science, China University of Geosciences (Wuhan), China. Her research interests include information security and digital signature.

WEI REN currently is a Professor in School of Computer Science, China University of Geosciences (Wuhan), China. He was with Illinois Institute of Technology, USA in 2007 and 2008, School of Computer Science, University of Nevada Las Vegas, USA in 2006 and 2007, and Hong Kong University of Science and Technology, in 2004 and 2005. He obtained his Ph.D. degree in Computer Science from Huazhong University of Science and Technology, China. He published more than 70 refereed papers, 1 monograph, and 4 textbooks. He obtained 10 patents and 5 innovation awards. He is a senior member of China Computer Federation and a member of IEEE.

TIANQING ZHU received the B.Eng. and M.Eng. degrees from Wuhan University, China, in 2000 and 2004, respectively, and the Ph.D. degree in computer science from Deakin University, Australia, in 2014. She is currently a Senior Lecturer with the School of Software, University of Technology Sydney, Australia. From 2014 to 2018, she was a Lecturer with the School of Information Technology, Deakin University, Australia.Her research interests include privacy preserving, data mining, and network security.

YI REN obtained his Ph.D. in Information Communication and Technology from the University of Agder, Norway in 2012. He was with the Department of Computer Science, National Chiao Tung University (NCTU), Hsinchu, Taiwan, as a Postdoctoral Fellow and an Assistant Research Fellow from 2012 to 2017. He is currently a Lecturer in the School of Computing Science at University of East Anglia (UEA), Norwich, U.K. His current research interests include security and performance analysis in wireless sensor networks, ad hoc, and mesh networks, LTE, and e-health security. He received the Best Paper Award in IEEE MDM 2012.

KIM-KWANG RAYMOND CHOO (SM?15) received the Ph.D. in Information Security in 2006 from Queensland University of Technology, Australia. He currently holds the Cloud Technology Endowed Professorship at The University of Texas at San Antonio (UTSA), and has a courtesy appointment at the University of South Australia. In 2016, he was named the Cybersecurity Educator of the Year - APAC (Cybersecurity Excellence Awards are produced in cooperation with the Information Security Community on LinkedIn), and in 2015 he and his team won the Digital Forensics Research Challenge organized by Germany's University of Erlangen-Nuremberg. He is the recipient of the 2018 UTSA College of Business Col. Jean Piccione and Lt. Col. Philip Piccione Endowed Research Award for Tenured Faculty, ESORICS 2015 Best Paper Award, 2014 Highly Commended Award by the Australia New Zealand Policing Advisory Agency, Fulbright Scholarship in 2009, 2008 Australia Day Achievement Medallion, and British Computer Society's Wilkes Award in 2008. He is also a Fellow of the Australian Computer Society, an IEEE Senior Member, and an Honorary Commander of the 502nd Air Base Wing, Joint Base San Antonio-Fort Sam Houston.