Pairing-Based Cryptography on the Internet of Things: A Feasibility Study

Ioanna Karantaidou(✉), Spyros T. Halkidis, Sophia Petridou, Lefteris Mamatas, and George Stephanides

Department of Applied Informatics, University of Macedonia, Thessaloniki, Greece
{dai16090,halkidis,spetrido,emamatas}@uom.edu.gr, steph@uom.gr

Abstract. Pairing-based cryptography (PBC) has recently received much attention, since the mathematical building block of pairings paved the ground for devising efficient cryptographic protocols exploiting an old inspiration, i.e., to produce the public key of an entity based on its *identity*. The so-called Identity-Based Cryptography (IBC) simplifies key management procedures, since it does not require certificate-based infrastructures. Moreover, it is an elliptic curve cryptosystem which entails that it offers the same security levels as other public key systems with much smaller key lengths. The above characteristics make it an attractive solution for resource-constrained environments such as the Internet of Things (IoT), where strong confidentiality and signature schemes are necessary. In this article, we conducted feasibility tests of pairing-based cryptography for middle-class IoT devices, such as the Raspberry Pi 3 platform.

Keywords: Pairing-based cryptography · Identity-based encryption
Short signatures · Internet of things

1 Introduction

The Internet of Things (IoT) has overwhelmed the cyber-physical world with billions of interconnected, fixed or mobile, devices ranging form wearables to smartphones [1]. Providing access anytime, anywhere, anyhow, the IoT has the potential to enable innovative application in many domains such as home or building automation, automotive, transportation surveillance and health-care. However, along with its scale, the IoT augments the security concerns due to the ubiquitous nature of the IP-things which are sending private data to back-end systems, e.g., edge or core cloud, and servers [2].

Let us consider a smart health-care system where patients' wearables can either communicate directly with a hospital's IoT infrastructure or send collected data to intermediate devices, e.g., a Raspberry Pi, which gathers and forwards them to the cloud. In such a scenario, either the links, i.e., Internet connections,

K. R. Chowdhury et al. (Eds.): WWIC 2018, LNCS 10866, pp. 219–230, 2018.
https://doi.org/10.1007/978-3-030-02931-9_18

or the nodes, i.e., collaborating devices, can be untrusted. Thus, strong cryptography is required to provide efficient confidentiality and authentication solutions. At the same time, the resource-constrained nature of the IoT environment, both in terms of low rate communication links and hardware-limited devices, strives for lightweight cryptographic protocols.

Recently, pairing-based cryptography (PBC) has received much attention [3], since pairings are revealed to be the mathematical tool which makes possible the Shamir's inspiration of certificate-less Public Key Cryptography (PKC) [4]. In 1984, Shamir proposed to authenticate an entity using some form of its *identity*, e.g., the email address, instead of its certificate. The main advantage of Identity-Based Cryptography (IBC) is that enables message encryption without the need of previously distributed keys. Such a facility is attractive in IoT use-cases where keys' pre-distribution is impractical or raises security concerns. For example, when the same key is shared among all "things", the impairment of a single device exposes the security of the whole network; or when a dedicated key is established for each couple of "things", the solution is not scalable. Moreover, the IBC provides the feature of including date information to the *identity* which entails revocation support without the use of certificate revocation lists (CRLs) [3].

Two fundamental protocols, which addressed in 2001 the issue of devising efficient IBC schemes, are the Boneh and Franklin's identity-based encryption (IBE) protocol [5] and the Boneh, Lynn and Shacham's (BLS) short signature scheme [6]. IBC is public key cryptography in the sense that roughly a public key is used for encryption and a private key for decryption. However, instead of producing the public key from the private, IBC defines that public keys are issued upon pre-existing identifiers. This entails no need of Certification Authorities (CAs). Instead, a Private Key Generator (PKG) authenticates the receiver, generates his private key and provides public system parameters to the sender.

Motivated by the general advantage of IBC, i.e., it simplifies key management procedures of certificate-based public key infrastructures, in this paper, we evaluate the feasibility of the aforementioned pairing-based protocols on middle-class IoT devices, such as the Raspberry-Pi 3 platform, through experimentation. More precisely, we present the basic Boneh and Franklin's IBE scheme, namely *BasicIdent* and we implement its *FullIdent* version, which is Chosen-Ciphertext Secure, using the Relic-Toolkit library [7]. In addition we evaluate the performance of BLS short signature scheme in contrast with the Elliptic Curve Digital Signature Algorithm (ECDSA) [8]. Thus, our work focuses on encryption and decryption for IBE schemes, and on signing and verifying for signatures schemes. Feasibility is expressed in terms of protocols' execution time, memory usage and energy consumption.

Our contribution can be summarized as follows:

1. we conducted real experiments to measure the resource requirements of fundamental pairing-based crytposystems in terms of CPU time, memory and energy;
2. we implement the *FullIdent* IBE scheme inside the Relic-Toolkit library;

3. we compare the performance of *BasicIdent* and *FullIdent* IBE schemes, as well as of BLS and ECDSA signature schemes for different security levels;
4. we tested the feasibility of pairing-based algorithms for middle-class IoT devices, such as the Raspberry-Pi 3 platform.

The rest of the paper is organized as follows. A motivating use-case scenario along with pairings preliminaries are presented in Sect. 2. Section 3 reveals the algorithms and computational overhead of the Boneh-Franklin's protocols and the BLS signature scheme, while Sect. 4 provides a discussion over the experimental results. Section 5 is a brief overview of related studies. We conclude the article along with some future work insights in Sect. 6.

2 Pairing-Based Cryptography

2.1 Pairings on the IoT

To demonstrate the features and advantages of PBC in real-world IoT environments, we consider a smart health-care scenario, illustrated in Fig. 1. A patient, Peter, uses his smartphone to send some medical results to doctor David, who is on-call on the hospital Hippocrates during Friday. With traditional public key cryptography, Peter needs to know David's public key. How can Peter be sure that holds the valid David's key, and not some other key substituted by a malicious attacker? So far, certificates are the classical solution to authenticate David's key. In PBC, the string $ID = David||Hippocrates||Friday$ could be a form of doctor's *identity* for his public key. Thus, without the need of previously distributed keys, Peter can encrypt his private $Message$ using the ID and public parameters announced by the PKG, which is implemented in a base station of the hospital's infrastructure.

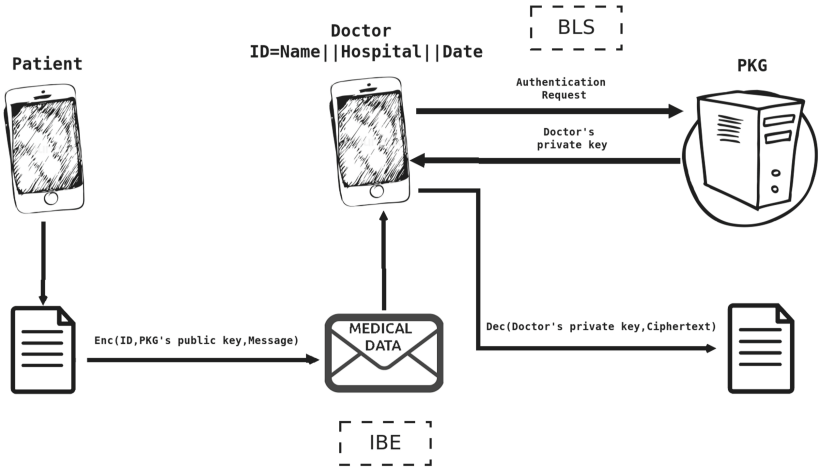


Fig. 1. An abstract view of pairing-based cryptography

Once David received the *Ciphertext* in his smartphone, he should be firstly authenticated and then proceed to data decryption. His ID can be exploited both by the signature scheme and the decryption function, since it constitutes part of his private key. The interesting feature is that the same doctor cannot access the data from his home or two days later, since his public key can be issued on-the-fly encoding information associated with his status. BLS instead of being able to authenticate a user upon his *identity*, it produces short signatures, i.e., 100s of bits in length. This is very attractive in low rates communications, since it entails shorter transmission time and fewer energy consumption in transmitters. Moreover, it is important when multiple authentication points exist. To clarify the details behind the IBE and BLS schemes used in this scenario, we provide a brief introduction to pairings.

2.2 Preliminaries on Pairings

Pairings are mathematical tools, widely used to implement the Shamir's idea for IBE and signature schemes that replace the public key of an entity with basic information about it, e.g., an *identity* string [4]. In Cryptography, a pairing, also called a (nondegenerate or admissible) bilinear pairing, is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ where $\mathbb{G}_1, \mathbb{G}_2$ are additive groups and $\mathbb{G}_T$ is a multiplicative group, all of prime order p. We use groups in which the discrete logarithm problem is believed to be adequately hard.

A pairing satisfies the following conditions:

1. **bilinearity:** i.e., $\forall P \in \mathbb{G}_1, Q \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$, it holds that $e(aP, bQ) = e(P, Q)^{ab}$. $aP = P + P + \cdots + P$ (a times) and it corresponds to the scalar multiplication in the additive group.
2. **non-degeneracy:** i.e., $\forall P \neq \mathcal{O}_{\mathbb{G}_1}$ and $Q \neq \mathcal{O}_{\mathbb{G}_2}$, $e(P, Q) \neq 1_{G_T}$, which expresses the fact that the map does not send all pairs to the neutral (identity) element of $\mathbb{G}_T$. $\mathcal{O}_{\mathbb{G}_1}, \mathcal{O}_{\mathbb{G}_2}$ and 1_{G_T} correspond to the neutral elements of the groups $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$, respectively.

In addition, pairings should be efficiently computable and are usually constructed on elliptic curves over finite fields. A pairing environment is considered as a tuple $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e)$, where P_i is a generator of the group $\mathbb{G}_i$. When $\mathbb{G}_1 = \mathbb{G}_2$, the pairing is called symmetric. This type was well-used at the dawn of PBC, but it has been gradually dismissed, since it mostly uses supersingular curves over small characteristic finite fields, i.e., very large groups for certain security levels. Thus, for efficiency reasons asymmetric pairings (i.e., $\mathbb{G}_1 \neq \mathbb{G}_2$) are more attractive in practice [9].

3 Confidentiality and Authentication Using Pairings

3.1 Identity-Based Encryption

An IBE scheme is a group of four algorithms:

- **Setup:** Run by the PKG. According to a security parameter, computes a master key that is kept secret and a system's public key P_{pub}, published as a parameter. Outputs the system parameters.
- **Extract:** Run by the PKG, takes as input the master key and an entity's identity $ID \in \{0,1\}^*$, outputs the entity's private key d_{ID}. Private keys are generated using the master key. It is the PKG's role to generate and distribute them to the communicating entities.
- **Encrypt:** Takes as input the system's public key P_{pub}, an identity ID and a message M and outputs the ciphertext C.
- **Decrypt:** Takes as input the system's public key P_{pub}, the entity's private key d_{ID} and a ciphertext C and outputs the message M or a message of failure.

3.1.1 Boneh-Franklin IBE Schemes

BasicIdent and CCA Security. The basic IBE scheme, named *BasicIdent* proposed by Boneh and Franklin in [5] is secure against eavesdropping, however it is susceptible to Adaptive Chosen Chiphertext Attacks (CCA). In a CCA, an adversary may make adaptively queries and obtain decryptions of ciphertexts different than the target ciphertext. The attack is successful when the adversary manages to obtain some information about the plaintext that corresponds to the target ciphertext.

A ciphertext C produced by the *BasicIdent* protocol has the following form: $C = (C_1, C_2) = (C_1, M \oplus H)$, where M is the message and H is the output of a hash function, both in binary format. As we can see, the ciphertext is malleable. The adversary can flip one specific bit of C_2, make a query with $(C_1, C_2') = (C_1, M' \oplus H)$ and obtain the decryption of M'. By flipping again the same bit, the message M is recovered.

FullIdent. This scheme, also proposed by Boneh and Franklin in [5], uses the Fujisaki-Okamoto transformation [10] which makes any cryptographic scheme CCA-secure. Two more cryptographic hash functions are added, the structure of the encrypted message is altered and one check at the end of the decryption process determines if the decrypted message is accepted or not. The original scheme contains symmetric pairings. In a more recent approach, the scheme is slightly adjusted to make use of asymmetric pairings [11].

The *FullIdent* is the following scheme:

- **Setup:** The PKG chooses a random $s \in \mathbb{Z}_p^*$ and keeps it as a master key. Then computes $P_{pub} = sP_1$ with P_1 being a generator for $\mathbb{G}_1$. The system parameters $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e)$, P_{pub} are published. The following cryptographic hash functions are defined: $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_2^*$, $H_2 : \mathbb{G}_T \rightarrow \{0,1\}^l$, $H_3 : \{0,1\}^l \times \{0,1\}^l \rightarrow \mathbb{Z}_p{}^*$ and $H_4 : \{0,1\}^l \rightarrow \{0,1\}^l$.

- **Extract:** For an identity ID, $Q_{ID} = H_1(ID) \in \mathbb{G}_2^*$ is computed and $d_{ID} = sQ_{ID} \in \mathbb{G}_2^*$ is the recipient's private key.
- **Encrypt and Decrypt:** The encryption and decryption algorithms are given by Algorithms 1 and 2, respectively.

Algorithm 1. The FullIdent encryption algorithm

Input: Message M with length l, system key P_{pub}, identity ID.
Output: Ciphertext $C = (U, V, W)$.
1: Compute $Q_{ID} = H_1(ID)$
2: Choose a random string σ with length l
3: Set $r = H_3(\sigma, M)$
4: $C = (rP_1, \sigma \oplus H_2(g_{ID}^r), M \oplus H_4(\sigma))$, where P_1 is a generator of $\mathbb{G}_1$ and $g_{ID} = e(P_{pub}, Q_{ID}) \in \mathbb{G}_T$.

Algorithm 2. The FullIdent decryption algorithm

Input: Ciphertext $C = (U, V, W)$, entity's private key d_{ID}.
Output: Message M or Failure message.
1: Compute $\sigma = V \oplus H_2(e(U, d_{ID}))$
2: Compute $M = W \oplus H_4(\sigma)$
3: Set $r = H_3(\sigma, M)$
4: Check that $U = rP_1$. If not, the ciphertext is rejected. Alternatively, the algorithm outputs the message M.

Based on the properties of pairings, the protocol is correct since:

$$\begin{aligned} e(U, d_{ID}) &= e(rP_1, sQ_{ID}) = e(P_1, Q_{ID})^{sr} \\ &= e(sP_1, Q_{ID})^r = e(P_{pub}, Q_{ID})^r = g_{ID}^r \end{aligned}$$

The performance of the scheme depends on the cost of pairing calculations. One pairing is calculated by the sender in step 4 of Algorithm 1 and one by the recipient in step 1 of Algorithm 2. The exponentiation in step 4 of the encryption algorithm also demands notable calculation. This entails that the encryption is more time demanding compared to the decryption, and the experimental results derived confirm it.

3.2 BLS Short Signature Scheme

The BLS signature scheme produces short length signatures and it makes use of pairings only during the verification process. More precisely, this signature scheme is a group of four algorithms:

- **Setup:** The system parameters $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, P_1, P_2, e)$ and one cryptographic hash function $H : \{0,1\}^* \rightarrow \mathbb{G}_1$ are defined and published.
- **Key Generation:** Run by the signer. A random $x \in \mathbb{Z}_p^*$ is picked as the private key. The public key pk is computed as $pk = xP_2 \in \mathbb{G}_2$ and published.

- **Signing:** Takes as input a message M and the private key x and produces a signature $\sigma \in \mathbb{G}_1$ as $\sigma = xH(M)$.
- **Verification:** Takes as input a message M, a public key pk and a signature σ. Checks the equality $e(\sigma, P_2) = e(H(M), pk)$. If it holds, it returns a succesful verification message.

In the next section we compare the performance of *BasicIdent* and *FullIdent* IBE schemes, and we evaluate BLS in contrast to the ECDSA signature scheme. We tested the feasibility of pairing-based algorithms in terms of CPU time, memory and energy requirements for different security levels.

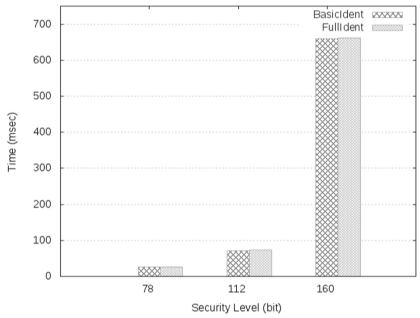
4 Experimental Results

We conducted our experiments on the Raspberry Pi 3 platform which has a 4 core ARM-Cortex, 1.2 GHz processor, 1 GB Memory and Raspbian GNU/Linux 8 OS. We use the elliptic curves $BN12$ and $BLS12$ from the Barreto-Naehrig and Barreto-Lynn-Scott family, respectively [12]. The curves are defined over a finite field $\mathbb{F}_q$ and are of the form $E/\mathbb{F}_q : y^2 = x^3 + b$ with $b \in \mathbb{F}_q$, embedding degree $k = 12$ and a sextic twist for faster computations. We provide results for the security levels 78, 112 and 160-bits which are defined in line with the q length. Pairing-based protocols use the efficient Optimal Ate pairing [3].

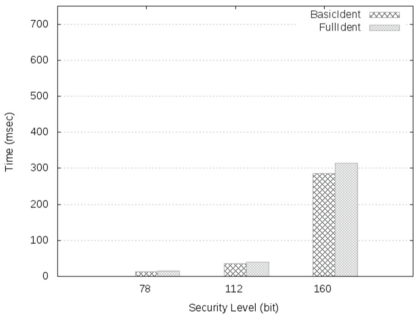
Feasibility in our study is expressed in terms of protocols' execution time, memory usage and energy consumption. The statistical evaluation of our results indicates a small standard deviation in case of IBE schemes. Therefore, the corresponding graphs depict mean values. In case of signature schemes, the standard deviation is non-negligible and, thus, both mean value and standard deviation are calculated over 1000 samples.

Figure 2 compares the encryption and decryption algorithms of the *BasicIdent* and *FullIdent* protocols' implementation in Relic-Toolkit library.

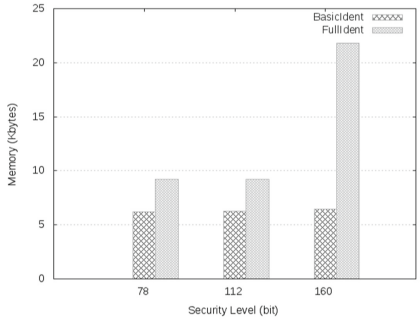
- **Execution Time:** In both protocols, encryption is more demanding than decryption, due to the overhead of the exponentiation g_{ID}^r in step 4 of Algorithm 1. The hash function H_1 that maps a value to a curve's element also consumes considerable time. It is interesting that the *FullIdent* protocol is chosen ciphertext secure with almost non time-overhead. Using hash functions and XOR operations it secures the *BasicIdent* efficiently. A slight deterioration in decryption is owed to the scalar multiplication rP_1 and an equality check in the additive group $\mathbb{G}_1$. The parameter that has serious impact on time is the security level. Moving from 112 to 160 bits adds around 550 ms in the encryption process and almost 250 ms in the decryption process. This is because pairing computations and scalar multiplications become time consuming in large groups of elliptic curve elements.
- **Energy Consumption:** Energy consumption is proportional to the execution time, since $E = P \times T$, where P is the power consumption and T is the algorithm's execution time. To obtain this measurement we put a USB detector between the power supply and the Raspberry Pi, and a constant



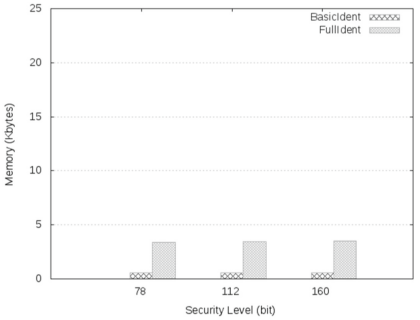
(a) BasicIdent and FullIdent encryption protocols



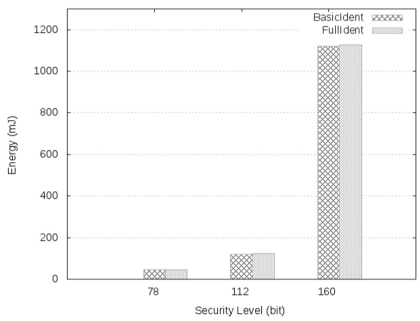
(b) BasicIdent and FullIdent decryption protocols



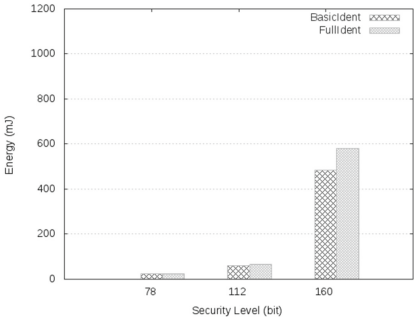
(c) BasicIdent and FullIdent encryption protocols



(d) BasicIdent and FullIdent decryption protocols



(e) BasicIdent and FullIdent encryption protocols



(f) BasicIdent and FullIdent decryption protocols

Fig. 2. Execution time, memory usage and energy consumption of pairing-based encryption and decryption on the Raspberry Pi 3 platform

$P = 1702\,mW$ was observed due to the protocols' execution. During encryption, lower security levels demand less than $200\,mJ$, while 160 bit security level increases the energy demands at $1100\,mJ$ for both *BasicIdent* and *FullIdent* protocols. Decryption is more efficient, since it requires less than $100\,mJ$ in low security levels and around $500-600\,mJ$ for 160 bit security. These levels of energy consumption indicate that the execution of protocols is feasible in our platform.

– **Memory:** The main part of the memory being used is reserved for the integration of the Relic-Toolkit library in Contiki. Thus we measured the memory overhead caused by the protocols themselves. We observe that, during the encryption, the *FullIdent* protocol is more memory-demanding compared to the *BasicIdent* especially in higher security levels. Both protocols decrease their requirements in decryption below $5\,KB$. It can be noted that the memory used in decryption seems quite indifferent of the security level.

Figure 3 compares the signing and verification algorithms of the BLS and ECDSA protocols' implementation in Relic-Toolkit library. Both protocols use elliptic curves, but only BLS is pairing-based.

– **Execution Time:** BLS and ECDSA exhibit similar performance in signature production. This is also confirmed from Table 1, where mean values and standard deviations have been recorded (due to space limitations similar statistics for verification process are omitted). Signing includes hash functions' computations and scalar multiplications in both protocols. BLS verification is slower due to the calculation of two pairings. Keeping in mind that this process is typically executed in base stations, BLS is a good candidate given the advantage of short signatures which have impact on time and energy transmission.
– **Energy Consumption:** Both protocols demand lower than $20\,mJ$ during signing for low security levels. This raises to $95\,mJ$ for the BLS protocol and to $80\,mJ$ for the ECDSA when the security level ups to 160 bit. Verification by the BLS is more demanding than ECDSA in every security level, with a noticable difference at high security.
– **Memory:** The BLS protocol seems to be memory efficient, since it requires around $3.5\,KB$ and $1.5\,KB$ during signing and verification process, respectively, irrelevantly to the security level.

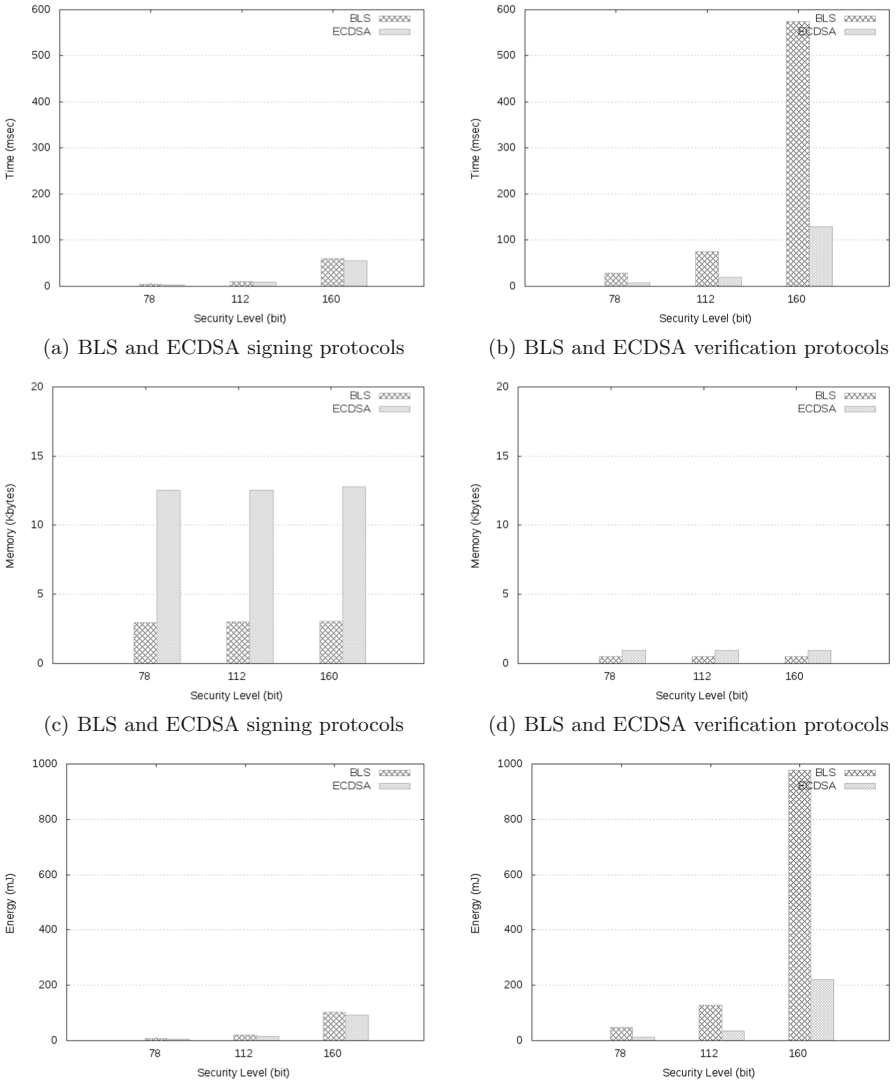
Table 1. Mean CPU time and standard deviation for signature schemes (msec)

Security level	Mean BLS	Std. Dev. BLS	Mean ECDSA	Std. Dev. ECDSA
78	3.189	0.047	3.036	0.233
112	10.349	0.742	8.0217	0.341
160	69.252	4.429	54.681	4.322

To summarize, all protocols evaluated through experimentation are shown to be feasible in middle-class IoT devices, such as the Raspberry Pi 3. Moreover, pairing-based protocols have features that make them attractive for such devices, e.g., short signatures.

5 Related Work

There is a large body of work on security and privacy issues for the IoT environment; we refer to [2] for a survey. Broadly, the Advanced Encryption Standard



(a) BLS and ECDSA signing protocols

(b) BLS and ECDSA verification protocols

(c) BLS and ECDSA signing protocols

(d) BLS and ECDSA verification protocols

(e) BLS and ECDSA signing protocols

(f) BLS and ECDSA verification protocols

Fig. 3. Execution time, memory usage and energy consumption of pairings and not-pairings signing and verification on the Raspberry Pi 3 platform

(AES) is used in association with public-key cryptosystems to provide confidentiality, while elliptic curves' signing schemes are dominating, since an 160-bit ECC key is roughly equivalent to an 1024-bit RSA key. A step forward, our work is motivated by a recent report of the National Institute of Standards and Technology (NIST) about PBC [3]. Two seminal cryptographic schemes that use pairings on elliptic curves are introduced in 2001; i.e., the Boneh and Franklin's

IBE protocol [5] and the BLS short signature scheme [6]. Our work elaborates on the *BasicIdent* and *FullIdent* IBE protocols, and the BLS scheme in an effort to experimentally evaluate them on middle-class IoT devices. To the best of our knowledge, this is the first feasibility test of the aforementioned fundamental protocols on the Raspberry Pi 3 platform.

In [13], Szczechowiak et al. explore the application of PBC on Wireless Sensor Networks (WSNs) and their results show that pairings can be implemented in real motes, although at a high computational cost. In a more theoretical approach [14], Mandal et al. discuss decisions regarding pairings, arithmetic field, curves and key-size which influence the feasibility of PBC on WSN. A pairing-based protocol for Home Area Networks was proposed in [15], while Oliveira et al. introduced the TinyPBC, a WSN authentication scheme based on pairings [16]. Private mutual authentication and private service discovery in mobile IoT are addressed in [17] and two new protocols are introduced. This work as well as the study of Attribute-Based Encryption (ABE) for access control in IoT [18] use pairings as cryptographic primitives and Raspberry platform, among others, as IoT platform for protocols' evaluation.

6 Conclusions and Future Work

In this paper we evaluated the feasibility of fundamental PBC schemes and concluded that they can be adopted by the IoT resource-constrained devices. We implemented and evaluated the *FullIdent* IBE scheme in contrast to the *BasicIdent*. Results show that the overhead of using the extended CCA-secure is negligible. In addition, we compared the BLS short signature scheme with the well-known ECDSA. The BLS algorithm seems to be approximately equivalent to ECDSA in the signing process, while it is more time and energy consuming in the verification process.

Our future work plans include the feasibility study of pairing-based cryptography on low-class IoT platforms, e.g., Zolertia RE-Mote 2 devices, which are resource-constrained. Challenges derived by devices' communication and messages' exchange worth further research. Finally, we plan to evaluate more pairing-based protocols, which according to the bibliography are more efficient because they require a single pairing evaluation during decryption instead of one for encryption and a second for decryption.

References

1. Atzori, L., Iera, A., Morabito, G.: The Internet of things: a survey. Comput. Netw. **54**(15), 2787–2805 (2010)
2. Sicari, S., Rizzardi, A., Grieco, L.A., Coen-Porisoni, A.: Security, privacy and trust in the Internet of things: the road ahead. Comput. Netw. **76**, 146–164 (2015)
3. Moody, D., Peralta, R., Perlner, R., Regenscheid, A., Roginsky, A., Chen, L.: Report on pairing-based cryptography. J. Res. Natl. Inst. Stand. Technol. **120**, 11–27 (2015)

4. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985). https://doi.org/10.1007/3-540-39568-7_5
5. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_13
6. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_30
7. Aranha, D.F., Gouvêa, C.P.L.: RELIC is an Efficient Library for Cryptography (2013)
8. Johnson, D., Menezes, A., Vanstone, S.: The elliptic curve digital signature algorithm (ECDSA). Int. J. Inf. Secur. **1**(1), 36–63 (2001)
9. Galbraith, S.D., Kenneth, K.G., Smart, N.P.: Pairings for cryptographers. Discret. Appl. Math. **156**(6), 3113–3121 (2008)
10. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 537–554. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_34
11. Galindo, D.: Boneh-Franklin identity based encryption revisited. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 791–802. Springer, Heidelberg (2005). https://doi.org/10.1007/11523468_64
12. Barreto, P.S.L.M., Costello, C., Misoczki, R., Naehrig, M., Pereira, G.C.C.F., Zanon, G.: Subgroup security in pairing-based cryptography. In: Lauter, K., Rodríguez-Henríquez, F. (eds.) LATINCRYPT 2015. LNCS, vol. 9230, pp. 245–265. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-22174-8_14
13. Szczechowiak, P., Kargl, A., Scott, A., Collier, M.: On the application of pairing based cryptography to wireless sensor networks. In: Proceedings of the 2nd ACM Conference on Wireless network security, pp. 1–12 (2009)
14. Mandal, M., Sharma, G., Bala, S., Verma, A.K.: Feasibility of public key cryptography in wireless sensor networks. J. Theor. Phys. Cryptogr. **7**, 20–24 (2014)
15. Jacobsen, R.H., Mikkelsen, S.A., Rasmussen, N.H.: Towards the use of pairing-based cryptography for resource-constrained home area networks. In: Proceedings of the 2015 Euromicro Conference on Digital System Design. IEEE, Portugal (2015)
16. Oliveira, L.B., Scott, M., López, J., Dahab, R.: TinyPBC: pairings for authenticated identity-based non-interactive key distribution in sensor networks. Comput. Commun. **34**(3), 485–493 (2011)
17. Wu, D.J., Taly, A., Shankar, A., Boneh, D.: Privacy, discovery, and authentication for the Internet of things. In: Askoxylakis, I., Ioannidis, S., Katsikas, S., Meadows, C. (eds.) ESORICS 2016. LNCS, vol. 9879, pp. 301–319. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45741-3_16
18. Ambrosin, M., et al.: On the feasibility of attribute based encryption on Internet of things devices. IEEE Micro **36**(6), 25–35 (2016)