Feasibility Characterization of Cryptographic Primitives for Constrained (Wearable) IoT Devices

Aleksandr Ometov†, Pavel Masek⋆, Lukas Malina⋆, Roman Florea†, Jiri Hosek⋆, Sergey Andreev†,
Jan Hajny⋆, Jussi Niutanen‡, and Yevgeni Koucheryavy†
†**Tampere University of Technology**, Finland, Tampere, Korkeakoulunkatu 10, FIN-33720
⋆**Brno University of Technology**, Czech Republic, Brno, Technicka 3082/12
‡**Intel Finland**, Tampere, Insinoorinkatu 7, FIN-33720
Contact e-mail: *aleksandr.ometov@tut.fi*

***Abstract*—The Internet of Things (IoT) employs smart devices as its building blocks for developing a ubiquitous communication framework. It thus supports a wide variety of application domains, including public safety, healthcare, education, and public transportation. While offering a novel communication paradigm, IoT finds its requirements closely connected to the security issues. The role of security following the fact that a new type of devices known as *wearables* constitute an emerging area. This paper delivers an applicability study of the state-of-the-art cryptographic primitives for wearable IoT devices, including the pairing-based cryptography. Pairing-based schemes are well-recognized as fundamental enablers for many advanced cryptographic applications, such as privacy protection and identity-based encryption. To deliver a comprehensive view on the computational power of modern wearable devices (smart phones, watches, and embedded devices), we perform an evaluation of a variety of them utilizing bilinear pairing for real-time communication. In order to deliver a complete picture, the obtained bilinear pairing results are complemented with performance figures for classical cryptography (such as block ciphers, digital signatures, and hash functions). Our findings show that wearable devices of today have the needed potential to efficiently operate with cryptographic primitives in real time. Therefore, we believe that the data provided during this research would shed light on what devices are more suitable for certain cryptographic operations.**

***Index Terms*—Bilinear Pairing, Cryptography, Group Signatures, IoT, Performance evaluation, Wearables**

I. Introduction

The Internet of Things (IoT) creates the means for interconnection of highly heterogeneous entities and networks bringing a variety of communication patterns, including Human-to-Human (H2H), Human-to-Machine (H2M), and Machine-to-Machine (M2M) communications. IoT in general and wearable technology in particular empower the industry to develop new technology in almost unlimited numbers. Today, the term *wearables* stands for connected devices that collect data, track activities and improve user experience across different application domains. From the IoT point of view, wearables could be characterized as networked "smart devices" equipped with microchips (System on the Chip, SoC), sensors, and wireless communications interfaces deployed in the immediate vicinity of their owner [1] (see also Fig. 1 indicating the devices used in everyday life of tomorrow).

New findings from the leading telecommunication players, such as Juniper [2] and Cisco [3], reveal that global retail revenue from smart wearable devices will treble by 2016, therefore reaching $53.2 billion by 2019, compared to the $4.5 billion at the end of 2015. The market over the following five years is expected to be substantially driven by the sales of smart watches and smart glasses. As it is common for new and highly innovative digital technologies, wearables will also challenge existing social and legal norms. In particular, wearable technologies raise a variety of privacy and safety concerns, which should be addressed immediately. Without strong security frameworks capable of being executed directly on wearable devices, attacks and malfunctions might overshadow any of the expected benefits.

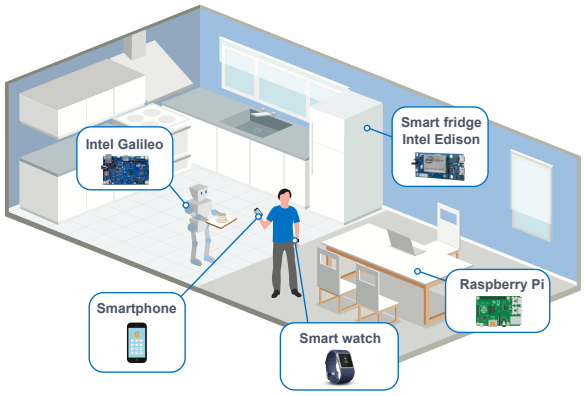


Fig. 1. Future secure smart home / IoT environment

Wearable devices can be secured by means of the public key cryptography, i.e., digital signature schemes providing user authentication and protecting data during their transmission over the medium. Information security specialists are targeting to design digital signature schemes that are (i) secure, (ii) computationally efficient, and (iii) have small communication overheads. Conventional digital signature schemes use standard operations and are based on mathematical assumptions, such as the discrete logarithm problem, the RSA problem, or integer factorization [4]. These conventional methods provide standard security properties, including authenticity, integrity, and non-repudiation.

In this paper, we expand our vision not only on classical cryptography but also on pairing-based algorithms by evaluating their usability for wearables and other constrained IoT devices (smart watches, smart phones, and embedded devices, see Section III). In particular, pairing-based cryptography is often used in modern solutions to implement privacy-enhancing features that are difficult to achieve with

978-1-5090-1941-0/16/$31.00 ©2016 IEEE

conventional asymmetric cryptography. Using bilinear pairing operations, it is possible to design schemes like group signatures [5], [6], anonymous attribute -based credentials [7], [8], or identity-based encryption [9]. Some of those mechanisms are particularly important for the IoT system operation, such as efficient revocation of invalid devices based on dynamic accumulators [10] and identification of attackers. These would be difficult to construct without pairing-based cryptography. Inspired by that, we analyze and evaluate the most common personal / wearable devices – starting from the conventional smartphones (Samsung Galaxy S4, Apple iPhone 6, etc.) to the embedded devices (Intel® Edison, Raspberry Pi 1 Model B, Raspberry Pi 2 Model B) to smart watches (Sony Smart Watch 3, Apple Watch) – with a particular focus on their ability to execute both standard and advanced cryptographic operations. As pairing-based cryptography primitives have not been rigorously evaluated on these devices so far, we also address this type of security functionality.

The rest of the paper is organized as follows. Section II provides the description of related work dealing with the classical cryptography and *pairing-based cryptography*. Further, in Section III we discuss the selected devices for our test scenarios that are employed and characterized in Section IV. Finally, the lessons learned and conclusions are summarized in Section V.

II. Classical and Pairing-based Cryptography

In our work, we consider the classical cryptographic primitives (calculations using big integer as multiplication, division, power functions and schemes for classical elliptic curves) for constructing the RSA signature, hash functions (SHA1, SHA-256), and block cipher (AES). These are well-known and well-analyzed construction blocks and we target to assess their execution time on low-power and constrained IoT devices.

A thorough overview is given in [11], where implementation of 12 lightweight and standard block ciphers in ATMEL AVR ATtiny45 has been described. Further, in [12], the authors focus on benchmarking the modern hash functions, where 15 hashes were evaluated on 8-bit micro-controllers. However, there has been very limited work documenting classical cryptography implemented on modern wearable devices e.g., smart watches.

To construct novel cryptographic primitives with advanced security properties, *Pairing-Based Cryptography* (PBC) has been developed. PBC-based solutions have exploded since 2000, when Joux [13] presented the three-party one-round Diffie-Hellman protocol based on bilinear pairings. The bilinear pairings enable efficient design of many protocols with enhanced properties, such as one-round three-party key agreement, identity-based encryption, and group signatures [5].

1) Bilinear Pairing Operations: A bilinear pairing function maps two elements of groups G_1 and G_2 onto the third cryptographic group G_T, i.e., $e : G_1 \times G_2 \rightarrow G_T$. The pairing function e must be computable, non-degenerative and bilinear. The pairing operations work with pairing-friendly Elliptic Curves (EC), including MNT curves (Miyaji-Nakabayashi-Takano) [14], BN curves (Barreto-Naehrig) [15]. The pairing operations can be symmetric ($G_1 = G_2$) or asymmetric ($G_1 \neq G_2$). Symmetric and/or asymmetric bilinear pairing operations can be computed by pairing algorithms, such as Weil, Tate, Ate, Eta, or O-Ate. Symmetric pairings are usually more computationally efficient than asymmetric pairings [16].

2) Pairing-Based Cryptographic Schemes: In general, pairing functions can reduce a problem that is in one group by solving it in a different group where the problem is easier to solve. This property enables to design new cryptographic schemes such as identity and attribute encryption, group signatures or three-party key establishment. In addition, the use of elliptic curves allows some pairing-based signature schemes to produce shorter signatures than the conventional digital signature schemes like RSA [17]. For example, the pairing-based short signature scheme BLS [18] employing the Weil pairing [19] produces 20 B only signatures. Due to space limitations, only a short overview is provided in this section; for more information related to pairing-based cryptography, please follow [20], [21].

3) Optimization of Pairing-Based Cryptographic Schemes: There are only few studies addressing the pairing-based cryptography on smartphones. For example, the work in [22] presents several ways for the efficient implementation of pairing-based cryptographic protocols on restricted devices. The BBS04 scheme that requires one online bilinear pairing during the signing phase is used to illustrate the below optimization approaches. The paper presents the pros and cons behind the approaches applied to the BBS04 scheme. The first approach uses the "Shamir's trick" [23], which reduces the time complexity of scalar multiplication (or modular exponentiation). The second method replaces an expensive pairing operation by less complex operations. The pre-computation of pairing operations is utilized, but the final efficiency depends on the number of components in the multi-exponentiations. The third approach delegates the computation of bilinear pairings to a more computationally stronger entity. Only the public parameters can be delegated and the communication between a constrained device and said entity must be fast.

The work in [24] presents the first Java wrapper of a pairing-based cryptography library. The authors implement jPBC that is a Java port of the PBC library written in C. As an example, the implementation of the BLS signature scheme [18] is described. The paper contains benchmarks of bilinear symmetric pairings and exponentiation operations for two devices (Samsung I9000 Galaxy S and a PC machine). However, the paper does not offer any performance results for asymmetric pairing operations that are required by many PBC signature schemes.

The challenge of PBC scheme optimization conventionally refers to decreasing the number of pairing operations. Optimization techniques, such as a pairing precipitation and pairing collapsing, can thus be applied to the PBC mechanisms. In addition, the verification phase of signature and group signature schemes can be optimized by using a batch verification trick [25]. The batch verification method can be used only in the case, when a verifier is able to verify more signatures in one batch. Some PBC schemes on constrained devices are able to

delegate expensive pairing operations to more powerful nodes. But, this trick can only be done if the pairing operation does not map secret and private parameters as inputs. More about the optimization tricks can be found in [4].

4) Performance Requirements of Pairing-Based Cryptographic Schemes: The bilinear pairing operations are considered to be more computationally expensive than other modular arithmetic operations, such as scalar multiplication and modular exponentiation. The computation time of a pairing operation depends on the type of a pairing algorithm, the type and the length of EC parameters, the implementation of the PBC methods, as well as on the hardware and software specifications of a device. For example, according to MIRACL benchmarks, one 512-bit pairing operation by the Tate algorithm takes 20 ms and one modular exponentiation with 1024-bit numbers takes 8.8 ms [26]. Unfortunately, some devices, such as smartphones and smart cards, need more time for computing a single pairing operation. For example, the results in [24] indicate that one symmetric pairing operation takes about 254 ms on smartphone (Samsung I9000 Galaxy S). Moreover, the results in [4] show that asymmetric pairing operations are more time consuming and require around 3 seconds on current Android smartphones. However, no similar study is presently available for general public in the area of wearable devices.

III. Selected Wearable Devices

Broadly, terms "wearable technology", "wearable devices", and "wearables" all refer to electronic technologies or computers that are incorporated into items of clothing and accessories, which can be worn on the body [1]. Following the fact that the computational performance is constantly growing (see Table I for a comparison of hardware parameters), contemporary wearable devices are becoming able to perform similar computing tasks as handheld devices or even laptop computers.

For the purposes of our research work, we have selected today's *pioneers* as well as already widely used devices from three main categories: (i) smartphones, (ii) smart watches, and (iii) embedded devices, see Fig. 2.

As representatives of the first group, we chose devices built on two main mobile platforms: Android and iOS. More specifically, we used Samsung Galaxy S4 (SGH-I337) and Jiayu S3 Advanced (JY-S3), both running Android 4.4.2, Apple iPhone 4s (MD128CS/A) running the iOS 7.1.2, and Apple iPhone 6 (MG4F2CN/A) with the latest iOS 9.1.

To provide a comprehensive evaluation at par with the selected smartphones, we also employed smart watches running Android Wear and Apple WatchOS. The utilized devices are correspondingly Sony Smart Watch 3 (SWR-50) with Android Wear 5.1.1 and Apple Watch 42mm Sport edition with WatchOS 2.0.

Following the fact that most of today's embedded devices (often named the IoT development boards) are intended to be used also as wearables, we decided to additionally evaluate the well-known examples from this class: Intel® Edison [27], Raspberry Pi 1 (Model B), and Raspberry Pi 2 (Model B). Both Raspberry Pi devices run the latest version of Raspbian

Fig. 2. Wearable devices used in our performance evaluation.

OS (Jessie, v 8.0) together with the latest version of Oracle JDK (1.8.0-b132). Edison features a Ubilinux 3.10.17-yocto-standard-r2 build equipped with JDK (1.8.0_66-b17)[1]. In more detail, Edison is a small-sized computing module aiming to enable the next generation of wearables and IoT devices, where size and power consumption are extremely important factors. In addition, Edison may be attached to a number of different extension boards, for example, to enable Arduino compatibility. Hence, Edison empowers a range of different use cases, whereas Raspberry Pi might be more suitable for graphics and multimedia related applications and products.

IV. Performance Evaluation

To adequately evaluate the performance of the devices listed in Table I, we decided to implement the above described security primitives in a unified framework. For Raspberry Pi, Android, and Android Wear devices, it has been executed as a standalone Java application. To run the framework on Apple devices (iPhone 4s, iPhone 6, and Apple Watch), we have ported the logic and created a standalone application written in Objective-C programming language. To make our assessment conditions even more equivalent, we terminated all unnecessary background processes and enabled the flight mode whenever possible. To execute our application on the restricted Intel® Edison board, we followed the manual [28] to prepare a Linux build equipped with JRE. Further, an executable jar file was designed, deployed, and executed on the device.

We split all of the tested devices based on their performance metrics into two groups: *Smart devices* and *IoT boards*. As the main evaluation criteria to characterize this equipment, we have selected the security primitive execution time. This is due to the unification and well-acceptance of this approach in addition to the fact that some of the devices are hardware restricted and, therefore, could not provide any other valuable and unified evaluation metric. Further, we compared the classical cryptographic primitives and the bilinear pairing on both

[1]Ubilinux stands for embedded Linux distribution based on Debian Wheezy and enables to run JVM / JDK. The targeted application domain for those platforms is embedded devices with limited memory and storage capacity; the image is currently available for Intel® Galileo Gen 1 / Gen2 and Edison.

TABLE I
SELECTED DEVICES WITH THEIR CORRESPONDING SPECIFICATIONS

Device	Type	SoC	Processor	RAM
Apple Watch	Smart Watch	APL0778	520 MHz Single-core Cortex-A7	512 MB
Sony SmartWatch 3 SWR50	Smart Watch	BCM47531	1.2 GHz Quad-Core ARM A7	512 MB
Apple iPhone 4s	Smartphone	APL A5	800 MHz Dual-Core Cortex A9 64bit	512 MB
Apple iPhone 6	Smartphone	APL A9	1.5 GHz Dual-Core Cortex A57 64bit	1 GB
Samsung I9500 Galaxy S4	Smartphone	APQ8064T	1.6 GHz Dual-Core Cortex-A15	2 GB
Jiayu S3 Advanced	Smartphone	MT6752	1.7 GHz Octa-Core 64bit Cortex A53	3 GB
Intel® Edison	IoT Development Board	Atom + Quark	500 MHz Dual-Core Intel® Atom™ CPU, 100 Mhz MCU	1 GB
Raspberry Pi 1 model B	IoT Development Board	BCM2835	700 MHz Single-Core ARM Cortex-A6	512 MB
Raspberry Pi 2 model B	IoT Development Board	BCM2836	900 MHz Quad-Core ARM Cortex-A7	1 GB

Intel® Edison development board and "off-the-shelf" devices available on today's market. The following results have been obtained as an average of 1000 executions for each operation to achieve statistically-reliable data.

A. Classical Cryptographic Primitives Evaluation

First, Fig. 3 indicates the average time overhead for encryption and decryption operations of the conventional non-optimized RSA schemes with correspondence to different decimal digits. Public and Private keys were generated using OpenSSL with default parameters. Adopting a security value of 1024 or 2048 bits and default public exponent (3 bytes) (which is reasonable for the constrained wireless devices [29]), the RSA Encryption operation remains under 1 ms on a typical Android smartphone, around 2.5 ms for a Smart Watch, and less than 12 ms on Intel® Edison and Raspberry Pi 2.

Decryption time looks less optimistic and, therefore, for an Android phone it takes around 25 ms, but up to 100 ms for an iPhone. Similar behavior is observed for Android Wear and Apple Watch – here, the values are 35 ms and 200 ms correspondingly. On the IoT boards, the execution may take up to half a second, which may still be feasible for delay-tolerant applications. Concerning smart devices, we can state that Sony Watch is demonstrating high performance even though it is not classified as a standalone device. Interestingly, here and further on, iPhone 4s is sometimes showing better results than iPhone 6 or Apple Watch, which may be due to the lack of the power consumption optimization feature on the version of iOS that was introduced only starting 9.0.1. Hence, CPU utilization is able to approach 90 %, while for the latest models it remains well below 50 %.

Taking into account such basic operation as Hashing function, we evaluate the execution of SHA1 and SHA2 (SHA-256) on all of the devices. The corresponding results are summarized in Fig. 4. We can conclude that for all of our test devices SHA1 and SHA2 are hardware optimized and mainly depend on the utilized equipment. As an example of the data encryption, we used AES 128 cipher. The corresponding results still follow the execution time pattern of public-key cryptosystems and hashing functions for all of our devices.

B. Pairing-based Primitives Evaluation

Further, we present the results for bilinear pairing operations. The curve types A and D (175) are assessed utilizing

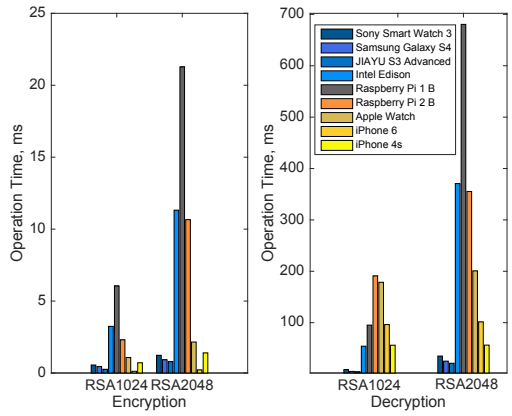


Fig. 3. RSA execution time.

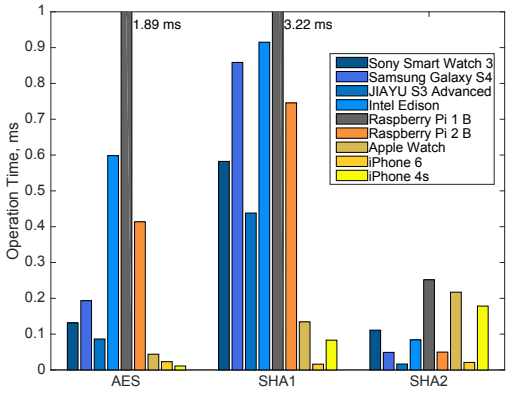
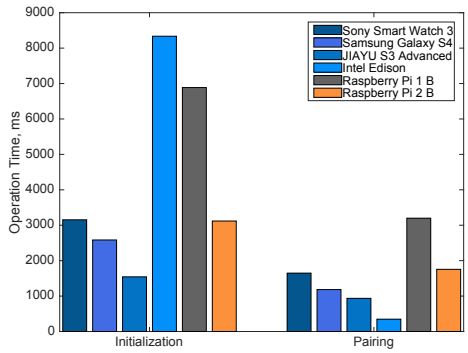


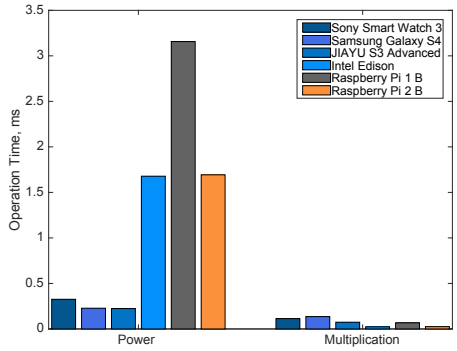
Fig. 4. Hashing and AES execution times.

jPBC-benchmark framework [24]. To this end, Fig. 5 shows one pairing operation with curve A. The most efficient device here is Intel Edison with JDK 1.8.0 that computes a single pairing operation in 580 ms. At the same time, Intel Edison needs over 8 seconds for the pairing initialization. Moreover, the modular exponentiation (EC point addition) operation takes about 4x more time than that on Android devices. EC multiplication follows a similar pattern. Our results confirm that some curve operations on smartphones and smart watches are more

efficient than on the single-board computers (Raspberry Pi 1/2 B). Further, Fig. 6 depicts the results of the PBC operations with D curves. Interestingly, Android devices with a more powerful CPU take as much time for one pairing operation as the less powerful devices (Intel Edison, Raspberry Pi). Hence, JRE seems to be more efficient than the Android platform. Our results indicate that optimized PBC schemes with only few pairing operations (i.e. < 2), several exponentiation, and scalar multiplications can be deployed in the security layers of non-real time IoT applications that run on current smartphones and wearables.



(a) Initialization and Pairing procedures.



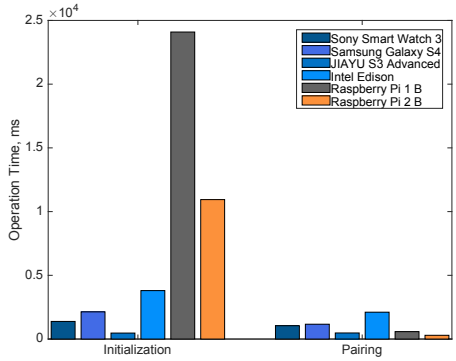
(b) EC point addition (Power) and EC point multiplication

Fig. 5. Execution time – Curve A.

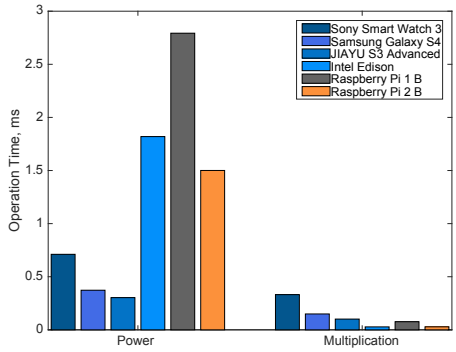
To provide a clear viewpoint of testing, Table II contains the information of which devices best match which cryptographic operations – with respect to HW parameters in Table I.

TABLE II
SUITABILITY OF WEARABLES FOR CRYPTOGRAPHIC OPERATIONS OVER RELATIVELY ACCEPTABLE TIME

Device	Cryptographic operations
Apple Watch	SHA 1/2; Curve operations
Sony SmartWatch 3	RSA 1024, 2048 E/D; Curve operations
Intel® Edison	RSA 2048 E/D; AES; SHA 2
Raspberry Pi 1 model B	RSA 1024 E/D
Raspberry Pi 2 model B	RSA 1024, 2048 E/D; AES; SHA 1



(a) Initialization and Pairing procedures.



(b) EC point addition (Power) and EC point multiplication

Fig. 6. Execution time – Curve D.

V. LESSONS LEARNED AND CONCLUSIONS

In this section, we discuss the important aspects that we faced during the implementation of our experimental framework, as well as outline our conclusions and future steps. In the process of developing the said framework, we have addressed a number of challenges regarding the very different requirements by the selected operating systems (i.e., Android, iOS, Android Wear, Apple WatchOS).

In particular, we had to adapt the jPBC library to run not only on smartphones but also on wearable devices. After the actual implementation, we learned that the same cryptographic primitives (i.e., the same application) may be optimized in completely different ways on similar devices. Further, a deeper study in the area of Apple development brought us to a number implementation challenges. For example, due to the lack of integrated information security libraries, we developed most of the primitives from scratch, thus solving many platform-dependent issues. However, as our future step, we plan to develop a pairing-based framework in Objective-C, that is, to enable its operation on iOS devices as well. Also the question of power consumption (CPU and memory) which is superficially mentioned in literature will be covered in further results.

Our main and the most essential learning while working with the pairing-based solutions is such that pairings consume from several hundreds of milliseconds to few seconds on

current handheld and IoT devices. We identified the most resource-consuming operation of pairing-based cryptography, that is, the bilinear pairing operation. The time necessary to compute this operation is several orders longer than that for the other operations on the elliptic curve. Therefore, in practical implementation, we highly recommend using cryptographic schemes [30], [31], in which an IoT device executes only basic operations on the curve and offloads the pairing operations to some central device with more computation power. On the other hand, some smartphone applications that send data in real-time and must secure data integrity and authenticity (e.g. for remote control systems) should avoid using the PBC schemes. These applications need to be secured by classical cryptographic primitives (SHA2, AES, RSA) that take only several milliseconds.

Finally, we can conclude that modern wearable electronics has already reached the computational power of a two-year-old smartphone and, thus, IoT world fulfills the security requirements of today. Constrained but powerful IoT devices, like Intel Edison, are designed so that the energy consumption is minimized. Due to that fact, the computational power is somewhat lowered, but this class of devices appears to be an attractive enabler for the required levels of information security. Importantly, the Raspberry Pi board, which is often nicknamed "a tiny and affordable computer" is demonstrating more modest performance results comparing to a small Edison chip designed specifically for the IoT-centric use cases.

ACKNOWLEDGMENT

The described research was supported by the Academy of Finland, the Ministry of Interior under grant VI20162018003, the National Sustainability Program under grant LO1401, and the Foundation for Assistance to Small Innovative Enterprises (FASIE) within the program "UMNIK" under grant 8268GU2015 (02.12.2015).

For the research, infrastructure of the SIX Center was used. We would like to thank to Intel Finland for the possibility to evaluate Intel Edison IoT development boards.

REFERENCES

[1] A. D. Thierer, "The Internet of Things and Wearable Technology: Addressing Privacy and Security Concerns without Derealing Innovation," *Rich. JL & Tech.*, vol. 21, pp. 6–15, 2015.

[2] M. S. Whitcup and K. LaMattina, "Juniper – What is Inhibiting Growth in the Medical Device Wearable Market?." http://bit.ly/1Dfffbf, September 2014.

[3] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2014–2019," February 2015.

[4] L. Malina, J. Hajny, and V. Zeman, "Usability of pairing-based cryptography on smartphones," in *Proc. of 38th International Conference on Telecommunications and Signal Processing (TSP)*, pp. 617–621, IEEE, 2015.

[5] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," in *Advances in Cryptology–CRYPTO 2004*, pp. 41–55, Springer, 2004.

[6] L. Nguyen and R. Safavi-Naini, "Efficient and provably secure trapdoor-free group signature schemes from bilinear pairings," in *Advances in Cryptology-ASIACRYPT 2004*, pp. 372–386, Springer, 2004.

[7] C. Paquin and G. Zaverucha, "U-prove cryptographic specification v1. 1," tech. rep., Microsoft Technical Report, http://connect.microsoft.com/site1188, 2011.

[8] J. Hajny, P. Dzurenda, and L. Malina, "Attribute-based credentials with cryptographic collusion prevention," *Security and Communication Networks*, 2015.

[9] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Advances in Cryptology–CRYPTO 2001*, pp. 213–229, Springer, 2001.

[10] J. Camenisch and A. Lysyanskaya, "Dynamic accumulators and application to efficient revocation of anonymous credentials," in *Advances in Cryptology—CRYPTO 2002*, pp. 61–76, Springer, 2002.

[11] T. Eisenbarth, Z. Gong, T. Güneysu, S. Heyse, S. Indesteege, S. Kerckhof, F. Koeune, T. Nad, T. Plos, F. Regazzoni, *et al.*, "Compact implementation and performance evaluation of block ciphers in attiny devices," in *Progress in Cryptology-AFRICACRYPT 2012*, pp. 172–187, Springer, 2012.

[12] J. Balasch, B. Ege, T. Eisenbarth, B. Gérard, Z. Gong, T. Güneysu, S. Heyse, S. Kerckhof, F. Koeune, T. Plos, *et al.*, *Compact implementation and performance evaluation of hash functions in attiny devices*. Springer, 2013.

[13] A. Joux, "A one round protocol for tripartite Diffie–Hellman," in *Algorithmic number theory*, pp. 385–393, Springer, 2000.

[14] A. Miyaji, M. Nakabayashi, and S. Takano, "New explicit conditions of elliptic curve traces for FR-reduction," *IEICE transactions on fundamentals of electronics, communications and computer sciences*, vol. 84, no. 5, pp. 1234–1243, 2001.

[15] P. S. Barreto and M. Naehrig, "Pairing-friendly elliptic curves of prime order," in *Selected areas in cryptography*, pp. 319–331, Springer, 2006.

[16] S. Chatterjee and A. Menezes, "On cryptographic protocols employing asymmetric pairings–the role of ψ revisited," *Discrete Applied Mathematics*, vol. 159, no. 13, pp. 1311–1322, 2011.

[17] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[18] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in *Advances in Cryptology—ASIACRYPT 2001*, pp. 514–532, Springer, 2001.

[19] V. S. Miller, "The weil pairing, and its efficient calculation," *Journal of Cryptology*, vol. 17, no. 4, pp. 235–261, 2004.

[20] S. D. Galbraith, K. G. Paterson, and N. P. Smart, "Pairings for cryptographers," *Discrete Applied Mathematics*, vol. 156, no. 16, pp. 3113–3121, 2008.

[21] R. Dutta, R. Barua, and P. Sarkar, "Pairing-based cryptographic protocols: A survey," *IACR Cryptology ePrint Archive*, vol. 2004, p. 64, 2004.

[22] S. Canard, N. Desmoulins, J. Devigne, and J. Traoré, "On the implementation of a pairing-based cryptographic protocol in a constrained device," in *Pairing-Based Cryptography–Pairing 2012*, pp. 210–217, Springer, 2013.

[23] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to elliptic curve cryptography*. Springer Science & Business Media, 2006.

[24] A. De Caro and V. Iovino, "jPBC: Java pairing based cryptography," in *Proc. of IEEE Symposium on Computers and Communications (ISCC)*, pp. 850–855, IEEE, 2011.

[25] A. L. Ferrara, M. Green, S. Hohenberger, and M. Ø. Pedersen, "Practical short signature batch verification," in *Topics in Cryptology–CT-RSA 2009*, pp. 309–324, Springer, 2009.

[26] I. Elashry, Y. Mu, and W. Susilo, "Jhanwar-Barua's Identity-Based Encryption Revisited," in *Network and System Security*, pp. 271–284, Springer, 2014.

[27] Intel® Edison, "One Tiny Platform, Endless Possibility." http://www.intel.de/content/www/de/de/do-it-yourself/edison.html, 2015.

[28] Intel® Developer Zone, "Installing the Eclipse* IDE - Install the Intel IoT Developer Kit version of Eclipse." https://software.intel.com/en-us/installing-the-eclipse-ide, 2015.

[29] M. Brown, D. Cheung, D. Hankerson, J. L. Hernandez, M. Kirkup, and A. Menezes, "PGP in Constrained Wireless Devices," in *USENIX Security Symposium*, 2000.

[30] R. Spreitzer and J.-M. Schmidt, "Group-signature schemes on constrained devices: the gap between theory and practice," in *Proceedings of the First Workshop on Cryptography and Security in Computing Systems*, pp. 31–36, ACM, 2014.

[31] J. Camenisch and A. Lysyanskaya, "Signature schemes and anonymous credentials from bilinear maps," in *Advances in Cryptology–CRYPTO 2004*, pp. 56–72, Springer, 2004.