On the Feasibility of Attribute-Based Encryption on Internet of Things Devices

Attribute-Based Encryption (ABE) could be an effective cryptographic tool for the secure management of Internet of Things (IoT) devices, but its feasibility in the IoT has been under-investigated thus far. This article explores such feasibility for well-known IoT platforms, namely, Intel Galileo Gen 2, Intel Edison, Raspberry Pi 1 Model B, and Raspberry Pi Zero, and concludes that adopting ABE in the IoT is indeed feasible.

Moreno Ambrosin
University of Padua

Arman Anzanpour
University of Turku

Mauro Conti
University of Padua

Tooska Dargahi
CNIT (Consorzio Nazionale Interuniversitario per le Telecomunicazioni)

Sanaz Rahimi Moosavi

Amir M. Rahmani

Pasi Liljeberg
University of Turku

The Internet of Things (IoT) is a growing trend populating the world with billions of interconnected devices that relate to physical things, ranging from wearable sensors to smartphones and smart cars.[1] Although the IoT has the potential to enable innovative new services and simplify communication between people and objects, it also brings new security and privacy challenges. For example, consider an IP-enabled sensor in a smart healthcare system that transmits patients' medical data to a remote healthcare server. In this scenario, the conveyed medical data could be routed through an untrusted network or stored in an untrusted cloud service, potentially exposing privacy-sensitive data to cyberattacks.

Besides generic IoT security and privacy issues, the concept of distributed IoT introduces additional context-specific challenges.[1] Devices not only send their data to the cloud, but they can also form an Intranet of Things, communicating with each other and with other IoT systems. For example, in a smart healthcare system, devices in a patient's smart house might need to interact directly with a hospital's IoT system. However, either of the collaborating entities could be untrusted, or the transmitted data might need to be revealed only to some selected parties. These challenges call for efficient authentication and fine-grained access control mechanisms that require advanced cryptographic methods. Furthermore, an important aspect to consider when it comes to resource-constrained IoT devices is providing flexible key management protocols, which has motivated researchers to develop efficient security solutions for IoT systems.[2]

In recent years, several security protocols have adopted Attribute-Based Encryption (ABE) as a building block in different distributed environments,[3] such as the IoT,[4] cloud services,[5] and medical systems.[6] ABE is a public key scheme in which both encryption and decryption are based on high-level data access policies. Considering the aforementioned

0272-1732/16/$33.00 © 2016 IEEE Published by the IEEE Computer Society

requirements in distributed and heterogeneous IoT scenarios, ABE provides a more efficient access control mechanism compared to conventional cryptographic algorithms.[3,6,7] Specifically, it allows fine-grained access control based on recipients' attributes, scales independently from the number of authorized users, is resilient against collusion attacks, and does not require key sharing or key management algorithms between the participating parties (the data owner does not need to identify the destination client). However, in spite of its noteworthy advantages, a proper key revocation algorithm is still a challenge in ABE and an ongoing research effort that is beyond this article's scope.[3] More relevant to our work, ABE suffers from high computational overhead.[6,8] The literature is still missing a proper assessment of ABE's efficiency on resource-constrained devices, which are widely used in the IoT domain.

To shine a light on ABE's feasibility in the IoT, we perform a comprehensive analysis of the cost of ABE operations on resource-constrained devices. Similar to our previous work,[7] which investigated ABE's feasibility on smartphone devices, in this article, we implement the original Key-Policy Attribute-Based Encryption (KP-ABE)[9] and Ciphertext-Policy Attribute-Based Encryption (CP-ABE)[10] on widely used IoT-enabling devices. Our work focuses on the evaluation of encryption and decryption (hereafter called "cryptographic operations") on four boards: Intel Galileo Gen 2, Intel Edison, Raspberry Pi 1 Model B, and Raspberry Pi Zero. Due to space limitations, we report only the results for CP-ABE, but we noticed that the KP-ABE experiments have similar quantitative behavior to the CP-ABE results. Supported by our observations from thorough experimental results, we provide evidence of the feasibility of adopting ABE on resource-constrained devices. Moreover, we present a smart healthcare use case application to evaluate the feasibility of using ABE in real-world IoT scenarios.

Expressive Encryption with ABE

In KP-ABE, each user's key represents an access policy, such as `(Dev_family=Board_XYZ ∧ Dev_role=Role_1) ∨ (Release_Date >2013)`, in which `Dev_family` and `Dev_role` represent string attributes, `Release_Date` represents a numeric attribute, and ∧ and ∨ are the AND and OR Boolean operators, respectively. Figure 1a shows a KP-ABE example in which a data owner encrypts the data specifying a list of attributes. If the data owner assigns the following set of attributes to the ciphertext `{ Dev_family =Board_XYZ, Dev_role=Role_1}` or `{ Release_Date=2014}`, the user will be able to decrypt the ciphertext: in these cases, the access policy associated to the user's secret key can be satisfied by the attributes assigned to the ciphertext.

Unlike KP-ABE, CP-ABE "enforces" the access policy directly on the data: each user's key is associated with a set of attributes, and a user can decrypt a ciphertext if his or her attributes satisfy the defined access policy on the data. Figure 1b illustrates an example of the CP-ABE; the data owner encrypts the data specifying the access policy `(Dev_family=Board_XYZ ∧ Dev_role=Role_1) ∨ (Release_Date>2013)` as part of the encryption. A user will be able to decrypt the ciphertext if his or her secret key is associated with a set of attributes that can satisfy the access policy.

Several factors influence ABE's performance in real-world applications, such as the desired security level, the underlying device's capacity (that is, available memory and CPU speed), and the number and type of attributes used in the access policy definition. The number of attributes, in particular, plays a fundamental role in ABE performance: encryption in CP-ABE requires computation of two exponentiations for each attribute in the resulting access policy. Similarly, KP-ABE encryption requires two exponentiations for each attribute enforced on the ciphertext. Decryption complexity in CP-ABE is upper bounded by l exponentiations and $2l$ pairing operations,[10] compared to only l exponentiations and pairing operations in KP-ABE; l is the number of attributes "matching" the access policy (in CP-ABE) or the key policy (in KP-ABE).

For a more complete evaluation of ABE, we also analyze the impact of using numeric attributes along with string attributes. We believe that, although the use of numeric attributes might be expensive, it provides

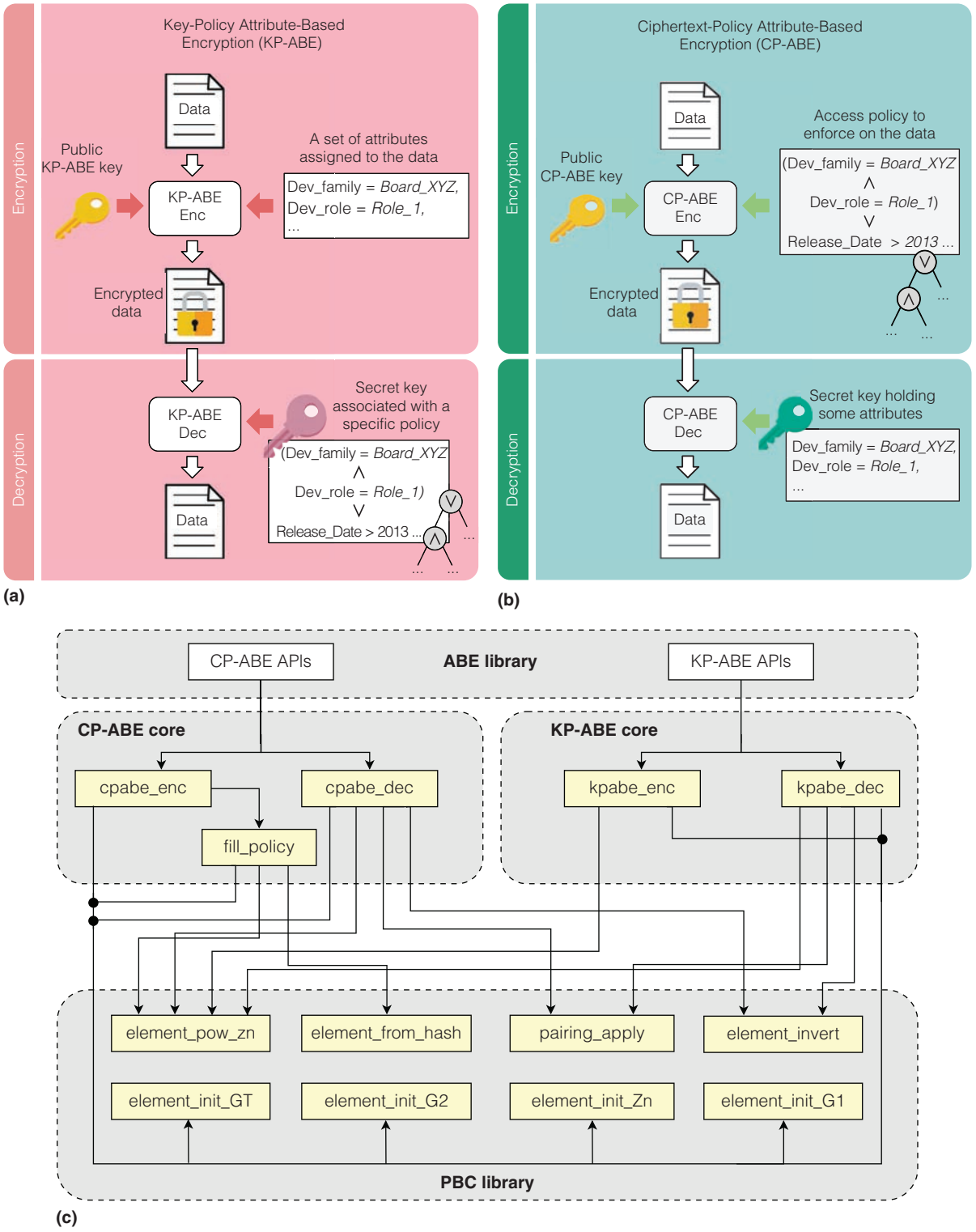


Figure 1. High-level overview of (a) Key-Policy Attribute-Based Encryption (KP-ABE) and (b) Ciphertext-Policy Attribute-Based Encryption (CP-ABE). (c) Simplified library structure.

additional expressiveness in policy definitions, especially in CP-ABE. As an example, there may be situations in which access to data should be restricted to a certain model of devices, released after a certain date (which can be represented as a 64-bit integer).

Feasibility of ABE on IoT Devices

Despite some researchers' argument about nonacceptable performance of ABE on mobile devices,[8] in our previous study, we implemented AndrABEn,[7] an ABE library for the Android operating system, and proved its efficiency. Similarly, in this section, we discuss the feasibility of ABE on resource-constrained IoT devices.

Before diving into the results of our experimental analysis, we clarify the concept of "feasibility," which we consider to be *latency,* as it has a direct impact on the consumed energy and is the most important discriminant factor in defining feasibility in this domain. The results from our study let us determine, at a high level, whether the use of ABE is feasible in specific applicative scenarios (such as video streaming and remote monitoring of healthcare appliances[11]), with respect to their latency requirements. We will present a smart healthcare use case example that uses CP-ABE for data encryption. Based on the use case's specific latency requirements, we can "tune" the adopted security level and determine the only reasonable number of attributes.

Experimental Setup

In our experiments, we adopt the same core C implementation of CP-ABE and KP-ABE that we used in previous work,[7] which implements the schemes in work by John Bethencourt and colleagues[10] and Vipul Goyal and colleagues,[9] respectively (the code is available at http://spritz.math.unipd.it/projects/andraben). Figure 1c presents a simplified representation of the library, showing its main dependencies with Ben Lynn's PBC (Pairing-Based Cryptography) library (http://crypto.stanford.edu/pbc/download.html), at a function-call level. For simplicity, we show only cryptographic operations. Although we are aware that more recent and improved ABE schemes exist,[3] we focused on the original schemes for their adoption in previous work[6,7] to maintain compatibility and comparability, and due to the availability of implementation libraries. We have chosen a set of middle-class IoT devices: low-cost, with a few megabytes of memory, network-enabled and compatible with a wide range of peripherals, to be used in different industrial or home automation applications.[12] The motivation behind our choice is to explore ABE's performance characteristics on IoT devices with diverse processing capabilities.

For our evaluation, we used the following settings:

- Intel Edison board: Silvermont Dual Core Intel Atom (500 MHz) + Intel Quark (100 MHz), 632 total Dhrystone MIPS (DMIPS), 256 Mbytes of memory, Yocto Linux OS, 1,335.84 mW baseline power.
- Intel Galileo Gen 2: Intel Quark X1000 (400 MHz), 500 total DMIPS, 1 Gbyte of memory, Yocto Linux OS, 7021.44 mW baseline power.
- Raspberry Pi 1 Model B: ARM1176 JZF-S (700 MHz), 875 total DMIPS, 512 Mbyte of memory, Raspbian OS, 2358.4 mW baseline power.
- Raspberry Pi Zero: ARM1176JZF-S (1,000 MHz), 1,250 total DMIPS, 512 Mbyte of memory, Raspbian OS, 1504 mW baseline power.

We evaluated the cryptographic operations' performance by varying the assured security level—that is, the number of bits that are used as primitives in cryptographic operations. Longer primitives lead to higher security levels. We considered three security levels (consistent with previous work[7,8]), equivalent to the security provided by AES symmetric encryption using key lengths of 80, 112, and 128 bits (corresponding to 1,024, 2,048, and 3,072 bits in RSA, respectively). To eliminate the impact of ciphertext size on execution time, we used a symmetric key to encrypt the plaintext and measured the performance of cryptographic operations of such a key. We considered policies with different numbers of attributes, ranging from 1 to 30, a range that represents a reasonable choice in real scenarios, while being consistent with related work.[6–8] Because all the devices run operating systems that support

multitasking, we report the average execution time for each board collected over several simulations, minimizing the impact of any background tasks on the results.

Evaluation and Discussion

Figures 2 and 3 show the execution time, memory usage, and energy consumption of CP-ABE on the considered devices, with varying numbers of attributes and security levels (confidence intervals are included in the figures but are not visible because they are too small). As expected, increasing the number of attributes leads to increased execution time and memory usage (and consequently, increased energy consumption). Similarly, a higher security level leads to increased workload on the tested devices.

The memory usage footprint is similar for all the boards, ranging between 14 and 15 Mbytes using a small or medium number of attributes. Security level does not significantly impact memory usage, which is instead affected by the number of adopted attributes.

In terms of execution time and energy consumption, Raspberry Pi 1 and Raspberry Pi Zero have similar behavior and show the best performance, whereas Intel Galileo shows the worst performance. For example, considering an 80-bit security level and 30 attributes, it takes approximately 5 seconds for encryption, and approximately 3.6 and 2.9 seconds for decryption, on Raspberry Pi 1 and Raspberry Pi Zero, respectively. With Intel Galileo, the execution time is approximately 15 and 13 seconds for encryption and decryption, respectively. For comparison, note that establishing a TLS (version: 1.2; cipher: `ECDHE-RSA-AES128-GCM-SHA256`; key length: 2048) session with `www.google.com:443`, on Intel Edison, requires on average 0.206 seconds. In the same setting, energy consumption of decryption and encryption on Raspberry Pi 1 and Raspberry Pi Zero are approximately 0.5 and 0.8 J, respectively, whereas Intel Galileo requires approximately 3.7 and 4.3 J, for decryption and encryption, respectively.

Our study provides a clear estimate of how the security level and number of attributes contribute to overall performance, and offers a caveat for choosing them. In general, the performance penalty is higher when the security level, rather than the number of attributes, is increased. For stronger security (that is, moving from 80 to 128 bits), the number of considered attributes must be reduced, on average, by 10 times. As an example of the tradeoff between security and the number of attributes, CP-ABE encryption with 15 attributes and a 112-bit security level shows an average execution time of 9.68 seconds and energy consumption of 1.75 J. Similar performance can be achieved with a security level of 128 bits using policies with fewer than five attributes. A notable insight from our experimentation is this Pareto-space of combinatorial choices of platform, security levels, and attributes.

We further analyzed the overhead of our implementation at a function-call level—that is, we measured the timing overhead introduced by each function in CP-ABE cryptographic operations on the Intel Edison board. In general, the encryption routine spends almost 91 percent of the time executing (multiple times) two functions from the PBC library: `element_from_hash`, to convert and hash value into a group element, and `element_pow_zn`, to perform exponentiation in Z_N. Decryption depends almost entirely on the `pairing_apply` function (almost 97 percent overhead).

Numeric Attributes in ABE

According to CP-ABE's original design,[10] access policies are expressed as a conjunction of Boolean predicates—such as A (that is, A = true), or $A < N$, where $N \in N$—and are represented as trees. Leaf nodes of such trees (for example, *A, B,* and *C* in Figure 4a) are attributes, whereas inner nodes represent logical threshold gates of the form K of N, meaning that, for a set of attributes to satisfy the subtree rooted in such a gate, the set must (recursively) satisfy at least K of the N subtrees of the inner node. A leaf node—that is, an attribute—is satisfied by a key, if such an attribute is associated with the key.

Consider the example in Figure 4a. The policy `(A∧B)∨C` is translated into a tree with three leaves and two inner threshold gates. The ∧ Boolean operator is translated into a 2-of-2 gate (that is, both subtrees connected to this gate must be true for this gate to be considered true), whereas the ∨ operator is a 1-of-2 gate (if at least one of the connected

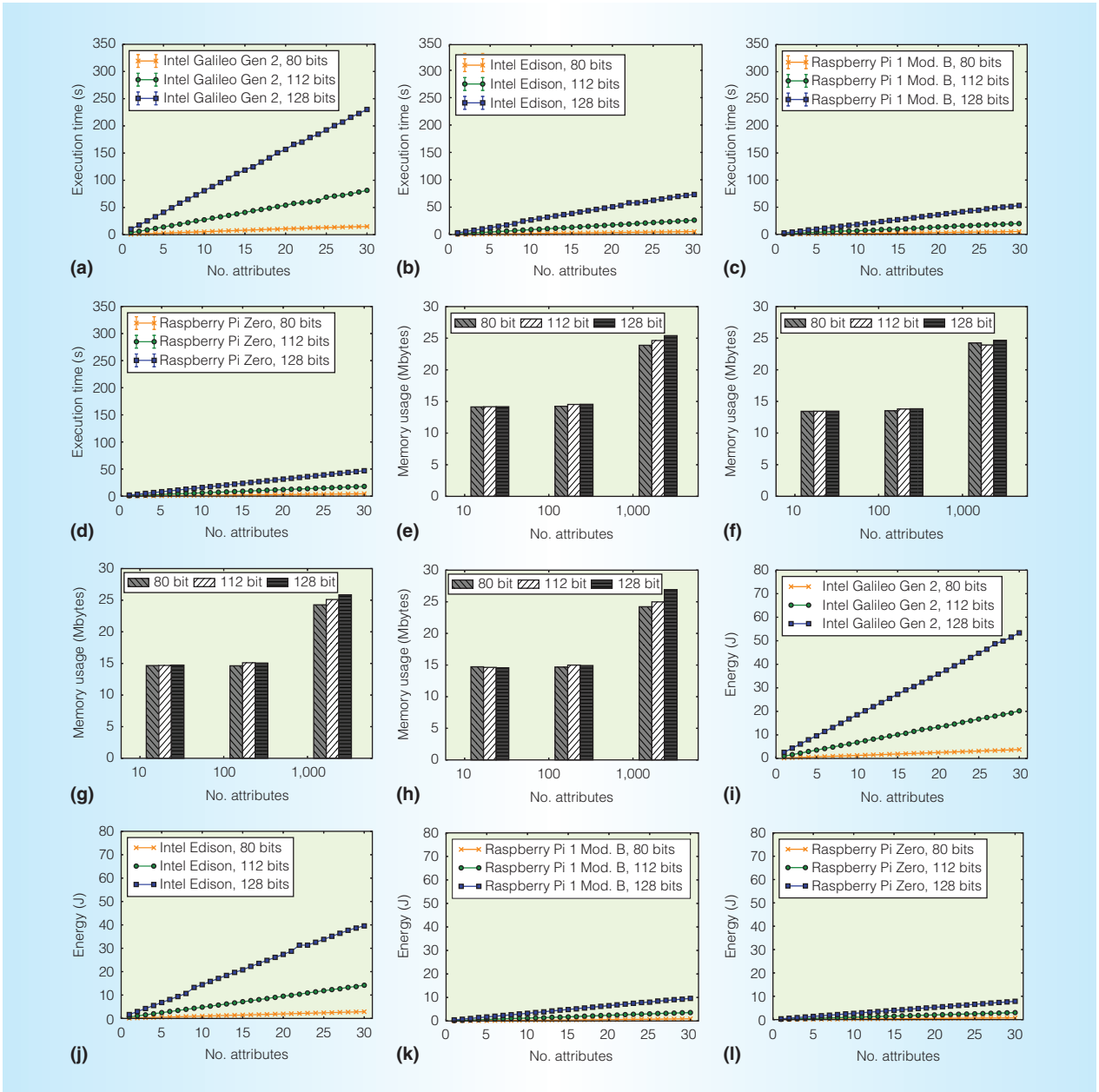


Figure 2. Execution time, memory, and energy consumption for CP-ABE encryption. Execution time for (a) Intel Galileo Gen 2, (b) Intel Edison, (c) Raspberry Pi 1, and (d) Raspberry Pi Zero; memory for (e) Intel Galileo Gen 2, (f) Intel Edison, (g) Raspberry Pi 1, and (h) Raspberry Pi Zero; and energy for (i) Intel Galileo Gen 2, (j) Intel Edison, (k) Raspberry Pi 1, and (l) Raspberry Pi Zero.

nodes to this gate is true, this gate will be considered true).

According to Bethencourt and colleagues,[10] a numeric attribute, such as $A = 9$, can be translated into a set of simple attributes indicating the value of each single bit in the attribute's binary representation. For example, using a 64-bit representation for an integer, the attribute $A = 9_{10} = 1001_2$ is translated into

A:xxxx…1xxx, A:xxxx…x0xx,
A:xxxx…xx0x, A:xxxx…xxx1,
A:eq_09, A:gt_2^02, A:lt_2^04, …

This represents the binary translation of 9 (`x` is a wildcard bit value), plus an attribute for exact matching (`A:eq_09`). It also represents other attributes—for example, the ones of the form `A:lt_2^N (A<2N)` and

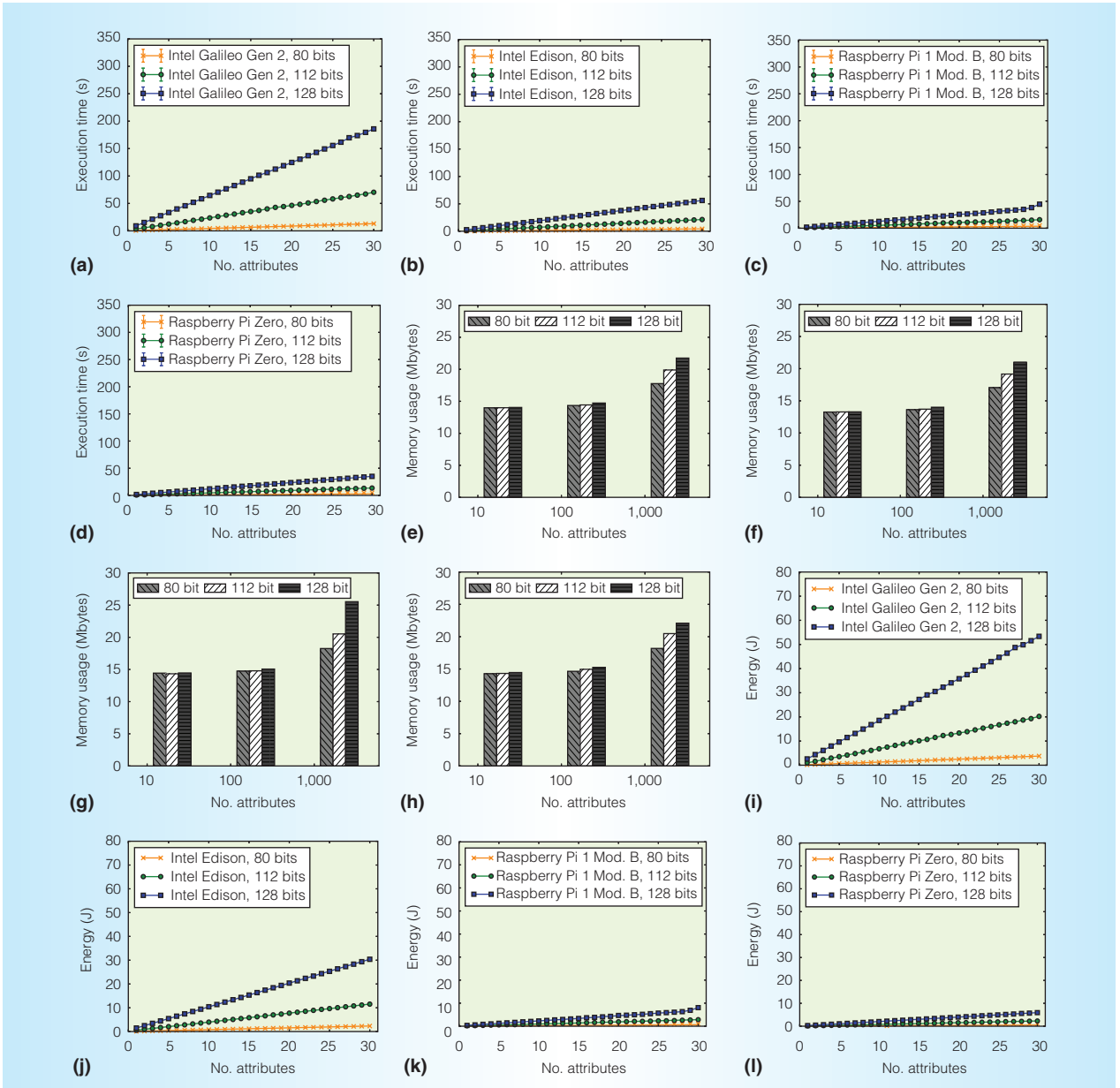


Figure 3. Execution time, memory, and energy consumption for CP-ABE decryption. Execution time for (a) Intel Galileo Gen 2, (b) Intel Edison, (c) Raspberry Pi 1, and (d) Raspberry Pi Zero; memory for (e) Intel Galileo Gen 2, (f) Intel Edison, (g) Raspberry Pi 1, and (h) Raspberry Pi Zero; and energy for (i) Intel Galileo Gen 2, (j) Intel Edison, (k) Raspberry Pi 1, and (l) Raspberry Pi Zero.

`A:gt_2^N (A>2N)`, which are "compressed" representations of the remaining bits, required due to the 64-bit representation of a numeric attribute.

Single numeric clauses can be converted into access tree structures of simple attributes. Figure 4b shows the translation of $A < 11$. As we can see, even simple access control policies involving numeric attributes generate quite complex trees and consequently impact the performance of cryptographic operations. To better understand such an impact, we measured the execution time of CP-ABE encryption using simple policies in the form $A < 2^X$, where X ranges from 1 to 24. Figure 4c presents our results, experimented on a

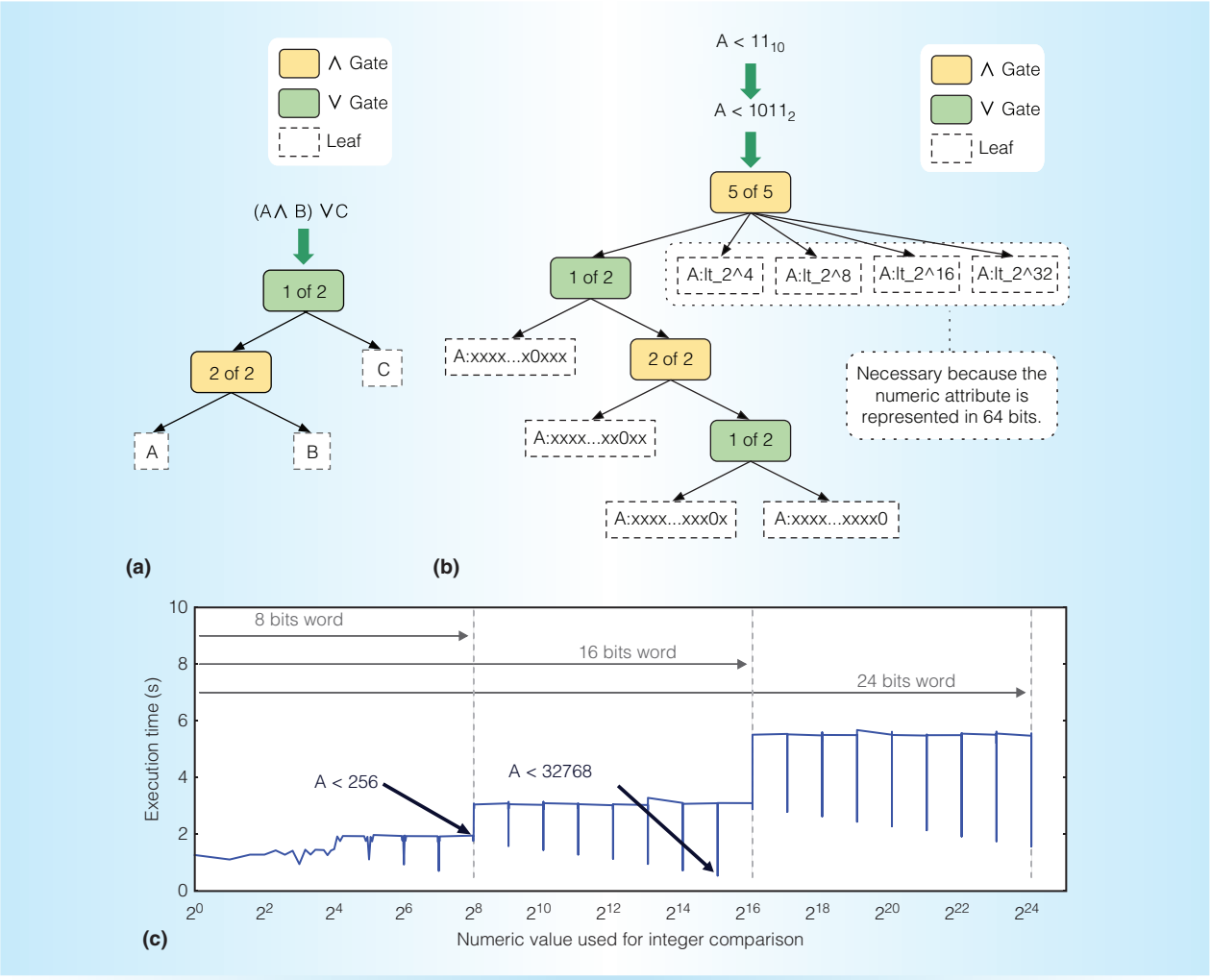


Figure 4. Access policy translation in CP-ABE. (a) Simple policy, (b) policy with numeric attributes, and (c) CP-ABE encryption time on a Raspberry Pi 1 access policy $A < N$, where N ranges from 2^0 to 2^{24}.

Raspberry Pi. We made two important observations:

- Encryption time (which depends on the size of the tree) does not grow directly with the size of the considered number, but rather with the "minimum number of bytes" necessary to represent the number.
- Numbers that are a power of 2 generate simpler access trees, with a consequent reduced encryption time. Moreover, for power of 2, the closer the most significant bit at 1 is to the size of the bit word in use (that is, 8, 16, 24, or 32), the simpler the corresponding access tree will be.

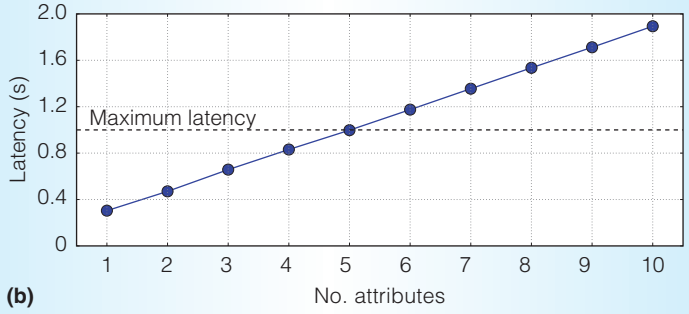
For example, in Figure 4c, the access policy $A < 256$ (2^8) generates an access tree with 11 leaves and 2 AND gates, requiring approximately 1.941 seconds for encryption, whereas encryption with $A < 768$ (2^{15}) generates a simpler access tree with only three leaves and one AND gate, requiring approximately 0.547 seconds. We can also extend these considerations on the usage of numerical attributes to the KP-ABE scheme from Goyal and colleagues[9] because it uses a similar access tree construction as that of Bethencourt and colleagues.[10]

Use Case: IoT in Healthcare

To demonstrate the feasibility of using ABE in real-world IoT scenarios, we consider a

Medical parameter	Heart rate	Respiration rate	Blood oxygen saturation	Body temperature	ECG
Sensor	SPO_2 finger grip	e-Health airflow sensor	SPO_2 finger grip	TMP36	e-Health ECG sensor
Sampling rate[13]	Every 5 seconds	Every 10 seconds	Every second	Every minute	500 samples per second
Sample size	1 byte	1 byte	3 bytes	3 bytes	3 bytes

(a)



(b)

Figure 5. Healthcare use case parameters and latency evaluation on an Intel Edison board, using an 80-bit security level. (a) Sensor properties and application parameters. (b) Latency on Intel Edison.

simple yet realistic use case: smart healthcare. We implemented a prototype wireless healthcare data reader system for remote monitoring, data collection, and processing. In our system, measurements from medical sensors are collected, encrypted with CP-ABE, and sent to a data collection server (via Wi-Fi) by an Intel Edison board equipped with an e-Health Sensor Shield version 2.0. The whole process is carried out by two services running on the board: the first reads the data from sensors and writes it into files (one per data type), and the second encrypts the files with CP-ABE and sends them to the server, which could represent an untrusted gateway, cloud service, or another IoT device. Figure 5a summarizes our application parameters. The specific system sampling rate requirements give us clear latency constraints based on which one should choose the acceptable range for the number of attributes and security level.

In general, the reading and sending rates should be roughly the same to guarantee the expected quality of service. Furthermore, because most of the traffic in our scenario is ECG data, approximately 1,500 bytes/second (500 reads of 3 bytes every second), we focus on ECG data. Given the approximately 80 ms needed for data transmission (per UDP packet) and the average 45 ms needed to encrypt the measurements file with AES, the most expensive operations are related to CP-ABE. To find a reasonable balance between the assured security level and expressiveness (in terms of the number of attributes), we conducted tests using up to 10 attributes and an 80-bit security level, measuring the overall latency. In Figure 5b, latency remains smaller, or close to 1 second (our upper bound for latency) with a maximum of five attributes. We can conclude that CP-ABE can be used in such a scenario to support up to five attributes with 80 bits of security. Note that the encryption time is a bit longer compared to the results given earlier because "time" includes AES encryption and per-file key generation, and the background reading service is always busy recording data.

We have shown the feasibility of adopting ABE in representative IoT systems. Our results can be a reference for researchers and designers of novel ABE-based

security solutions. We believe future research should focus on improving ABE efficiency, via both a careful selection of attributes and software and hardware optimizations for the cryptographic library. Our analysis shows that the utilized library can be significantly optimized via proper memory management, customized data structure deployment, and simplification of cryptographic arithmetic operations considering input attributes. Moreover, considering the fact that the complexity of CP-ABE and KP-ABE depends on the number of exponentiations and pairing operations performed by each of their algorithms, future work could address the migration of complex arithmetic operations, such as exponentiation, to hardware accelerators (for example, custom logic on field-programmable gate arrays) in order to enhance energy efficiency and total execution time. MICRO

Acknowledgments

This research was partially supported by the EU Marie Curie Fellowship PCIG11-GA-2012-321980 and EU projects ReCRED (ref. 653417), EU TagItSmart! (H2020-ICT30-2015-688061), and EU-India REACH (ICI+/2014/342-896).

References

1. R. Roman et al., "On the Features and Challenges of Security and Privacy in Distributed Internet of Things," *Computer Networks*, July 2013, pp. 2266–2279.
2. S.R. Moosavi et al., "Session Resumption-Based End-to-End Security for Healthcare Internet-of-Things," *Proc. IEEE Int'l Conf. Computer and Information Technology*, 2015, pp. 581–588.
3. S.S.M. Chow, "A Framework of Multi-Authority Attribute-Based Encryption with Outsourcing and Revocation," *Proc. 21st ACM Symp. Access Control Models and Technologies*, 2016, pp. 215–226.
4. X. Li et al., "Smart Community: An Internet of Things Application," *IEEE Comm.*, Dec. 2011, pp. 68–75.
5. H. Ma et al., "Verifiable and Exculpable Outsourced Attribute-Based Encryption for Access Control in Cloud Computing," *IEEE Trans. Dependable and Secure Computing*, 2015; doi:10.1109/TDSC.2015.2499755.
6. L. Ming et al., "Data Security and Privacy in Wireless Body Area Networks," *IEEE Wireless Comm.*, Feb. 2010; doi:10.1109/MWC.2010.5416350.
7. M. Ambrosin et al., "On the Feasibility of Attribute-Based Encryption on Smartphone Devices," *Proc. Workshop IoT Challenges in Mobile and Industrial Systems*, 2015, pp. 49–54.
8. X. Wang et al., "Performance Evaluation of Attribute-Based Encryption: Toward Data Privacy in the IoT," *Proc. IEEE Int'l Conf. Comm.*, 2014; doi:10.1109/ICC.2014.6883405.
9. V. Goyal et al., "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data," *Proc. 13th ACM Conf. Computer and Comm. Security*, 2006, pp. 89–98.
10. J. Bethencourt et al., "Ciphertext-Policy Attribute-Based Encryption," *Proc. IEEE Symp. Security and Privacy*, 2007, pp. 321–334.
11. D. Warren and C. Dewar, *Understanding 5G: Perspectives on Future Technological Advancements in Mobile*, tech. report, GSMA Intelligence, 2014.
12. K. Spilker, *From the MVPs: Introduction to the Internet of Things from the Device to Microsoft Azure Cloud*, Microsoft Press, 2015.
13. J. Ming-Zhe et al., "IoT-Based Remote Facial Expression Monitoring System with sEMG Signal," *Proc. IEEE Sensors Applications Symp.*, 2016; doi:10.1109/SAS.2016.7479847.

Moreno Ambrosin is a PhD student in computer science at the University of Padua. His research interests include distributed systems security. Ambrosin received an MSc in computer science from the University of Padua. He is a student member of IEEE. Contact him at ambrosin@math.unipd.it.

Arman Anzanpour is a PhD student in the IoT for Health group at the University of Turku. His research interests include the Internet of Things and smart health monitoring frameworks. Anzanpour received a master's degree in biomedical engineering from Amirkabir University of Technology.

He is a student member of IEEE. Contact him at armanz@utu.fi.

Mauro Conti is an associate professor in the Department of Mathematics at the University of Padua. His research interests include security and privacy. Conti received a PhD in computer science from Sapienza University of Rome. His awards include a Marie Curie Fellowship and a fellowship by the German DAAD. He is an associate editor of *IEEE Communications Surveys & Tutorials* and *IEEE Transactions on Information Forensics and Security*. He is a senior member of IEEE. Contact him at conti@math.unipd.it.

Tooska Dargahi is a postdoctoral researcher at CNIT (Consorzio Nazionale Interuniversitario per le Telecomunicazioni) and the University of Rome Tor Vergata. Her research interests include security and privacy. Dargahi received a PhD in computer engineering from Islamic Azad University, Science and Research Branch. She is a professional member of ACM. Contact her at tooska.dargahi@cnit.it.

Sanaz Rahimi Moosavi is a PhD student in the Department of Information Technology at the University of Turku. Her research interests include security and privacy, healthcare technology, the Internet of Things, and lightweight cryptography. Rahimi Moosavi received an MSc (Tech.) in IT networked systems security from the University of Turku. She is a student member of IEEE. Contact her at saramo@utu.fi.

Amir M. Rahmani is the Marie Curie Global Fellow at the University of California, Irvine, and TU Wien, Austria. He is also an adjunct professor (docent) in embedded parallel and distributed computing at the University of Turku. His research interests include self-aware parallel and distributed computing, healthcare Internet of Things, and embedded systems. Rahmani received a PhD in ICT from the University of Turku. He is a member of IEEE. Contact him at amirr1@uci.edu.

Pasi Liljeberg is an adjunct professor of embedded computing architectures at the University of Turku. His research interests include the Internet of Things, healthcare technology, embedded systems, and multicore processor architectures. Liljeberg received a PhD in communication systems from the University of Turku. He is a member of IEEE. Contact him at pasi.liljeberg@utu.fi.