# fABElous: An Attribute-Based Scheme for Industrial Internet of Things

Michele La Manna
*Information Engineering dept.*
*University of Florence*
Florence, Italy
michele.lamanna@unifi.it

Pericle Perazzo
*Information Engineering dept.*
*University of Pisa*
Pisa, Italy
pericle.perazzo@iet.unipi.it

Marco Rasori
*Information Engineering dept.*
*University of Florence*
Florence, Italy
marco.rasori@unifi.it

Gianluca Dini
*Information Engineering dept.*
*University of Pisa*
Pisa, Italy
gianluca.dini@unipi.it

***Abstract*—The Internet of Things (IoT) is a technological vision in which constrained or embedded devices connect together through the Internet. This enables common objects to be empowered with communication and cooperation capabilities. Industry can take an enormous advantage of IoT, leading to the so-called Industrial IoT. In these systems, integrity, confidentiality, and access control over data are key requirements. An emerging approach to reach confidentiality and access control is Attribute-Based Encryption (ABE), which is a technique able to enforce cryptographically an access control over data. In this paper, we propose fABElous, an ABE scheme suitable for Industrial IoT applications, which aims at minimizing the overhead of encryption on communication. fABElous ensures data integrity, confidentiality, and access control, while reducing the communication overhead of 35% with respect to using ABE techniques naively.**

***Index Terms*—Attribute-Based Encryption, ABE, Ciphertext-Policy Attribute-Based Encryption, CP-ABE, Broadcast.**

## I. Introduction

Internet of Things (IoT) technologies [1]–[3] allow us to connect constrained or embedded devices through the Internet. This has a deep impact in our everyday life, as common objects can be empowered with communication and cooperation capabilities. In particular, industry can take an enormous advantage of IoT. For example a smart factory can be monitored and controlled through the Internet, thus optimizing the industrial processes [4]. Another example is a smart warehouse, in which sensors can tell to automated guided vehicles where to find particular goods in order to load them on a given truck. In all these systems, security is a key requirement, especially for integrity, confidentiality, and access control over data. An emerging approach to reach confidentiality and access control is *Attribute-Based Encryption* (ABE) [5]–[10], a cryptographic technique that enforces an access control mechanism over encrypted data. ABE describes data and decrypting parties by means of *attributes*, and it regulates the "decryptability" of data with *access policies*, which are Boolean formulas defined over these attributes. Though ABE techniques offer a high level of security and an intrinsic fine-grained access control, they do not fit easily in the IoT world. Despite one may think, the computation power required to execute ABE operations is not the major concern, as ABE was shown to be well-suitable for IoT devices [11], [12]. Instead, the most challenging aspect is the communication overhead generated by the ABE encryption (over 1 KB overhead per message), which may be quite burdensome for wireless networks with limited bitrate like those employed in IoT [13], [14]. Indeed, modern IoT networks use low-power communication protocols like Bluetooth LE, IEEE 802.15.4, and LoRa, which provide for low bitrates (230 Kbps for BLE [15], 163 Kbps for 802.15.4 [16], 50 Kbps for LoRa [17]).

In this paper, we propose fABElous, an ABE scheme suitable for Industrial IoT applications which aims at minimizing the communication overhead introduced by ABE encryption. The fABElous scheme ensures data integrity, confidentiality, and access control, while reducing the communication overhead of 35% with respect to using ABE techniques naively.

The rest of the paper is structured as follows. Section II introduces some related work and some background on Attribute-Based Encryption. Section III describes fABElous in detail: its architecture, its reference use case, its system procedures, its reference threat model. Section IV analyzes the performances of fABElous in terms of communication overhead.

## II. Background and Related Work

### A. Background: CP-ABE

There are two ABE paradigms in the literature: *Ciphertext-Policy Attribute Based Encryption* (CP-ABE), and *Key-Policy Attribute Based Encryption* (KP-ABE). Although the names are quite self-explanatory, we give intuition into the workings of such paradigms. The access control mechanism implemented by KP-ABE answers the question: "What type of data can I (the decryptor) access?", in this case the policy is embedded inside the key belonging to the decryptor (Key-Policy). Differently, the access control mechanism implemented by CP-ABE answers the question: "Who can access the data that I (the encryptor) am encrypting?"; or better yet: "What *characteristics* must a decryptor have in order to access the data that I am encrypting?", in this case the policy is embedded inside the ciphertext itself (Ciphertext-Policy). Typically, CP-ABE is considered a more flexible solution [7] because it guarantees more control to the encryptor over the data. For this reason, we will now focus on CP-ABE, and give

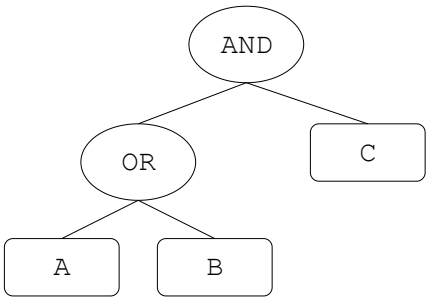


Fig. 1. Example of access policy. Logical operators are nodes represented as ellipses, while attributes are leaves depicted in rounded boxes. This policy reads as "At least one between $B$ OR $A$ must be present, AND $C$ must be present as well".

some background notions that will help the reader to better understand the rest of the paper.

In the following, we will call *data producers* whoever produces and encrypts data. Similarly, we will call *data consumers* whoever decrypts and consumes data. Each data producer encrypts data by means of an *encryption key* ($EK$). Each data consumer decrypts data by means of a *decryption key* ($DK$). The encryption key is public and unique for all the data producers, whereas the decryption key is private and specific to a single data consumer. A single piece of encrypted data is called *ciphertext* ($CP$). Each data consumer is described by an *attribute set* ($\gamma$), which is embedded into its decryption key. Two examples of attribute sets can be $\gamma_1 = \{A, B, D\}$, and $\gamma_2 = \{A, C\}$. The access rights on a ciphertext are described by an access policy ($\mathcal{P}$). An access policy can be seen as a tree, where the inner nodes are the logical operators "AND" and/or "OR" and the leaves are attributes. As an example, consider the policy $\mathcal{P} = (A \,\mathrm{OR}\, B) \,\mathrm{AND}\, C$ in Fig. 1. It appears clear from the figure that the attribute set $\gamma_2$ *satisfies* the access policy $\mathcal{P}$, while $\gamma_1$ does not.

The following set of *cryptographic primitives* (from now on, simply primitives) are commonly found in Attribute-Based Encryption schemes, and we will make use of them as black boxes.

*1)* $(MK, EK) = \mathrm{Setup}()$*:* This primitive initializes the cryptographic scheme. It outputs a *master key* $MK$ and an associated *encryption key* $EK$.

*2)* $CP = \mathrm{Encrypt}(M, \mathcal{P}, EK)$*:* This primitive encrypts a plaintext $M$ under the policy $\mathcal{P}$. It takes as input the message $M$, the policy $\mathcal{P}$, and the encryption key $EK$. It outputs the ciphertext $CP$.

*3)* $DK = \mathrm{KeyGen}(\gamma, MK)$*:* This primitive generates a decryption key. It takes as input a set of attributes $\gamma$ which describes the data consumer, and the master key $MK$. It outputs a decryption key $DK$.

*4)* $M = \mathrm{Decrypt}(CP, DK)$*:* This primitive decrypts a ciphertext $CP$. It takes the ciphertext $CP$ and the data consumer's decryption key $DK$ as input, and outputs the message $M$ if decryption is successful, $\perp$ otherwise.

### B. Related Work

Attribute-Based Encryption has been applied to protect confidentiality and ensure fine-grained access control in many different application scenarios like cloud computing [8], [18]–[20], e-health [21], wireless sensor networks [10], Internet of Things [22], [23], smart cities [9], online social networks [24]. In this section we will call *users* humans that take part in the services/systems described. We will also call *nodes* the independent devices (mainly sensors and actuators) that take part in the services/systems described.

Yu et al. [10] realized the first distributed fine-grained data access control for Wireless Sensor Networks (WSNs) using ABE. In their work, the system is composed by one controller, many users and many nodes. The network controller assigns a secret key to each user, according to a policy that describes the type of data such user should decrypt. Every node possesses a set of attributes, which are generated from the controller and loaded on the node before its installation. Their system is able to revoke a key with a single broadcast message. However, their system does not take into account the possibility of actuators which receive and use ABE-encrypted data. This precludes the possibility of having actuators which receive and use ABE-encrypted data in complex IoT scenarios like a smart factory. In addition, FDAC uses *Key-Policy Attribute-Based Encryption* (KP-ABE) [6], which is less flexible than CP-ABE, as already underlined in the previous section [7], [25], [26].

Picazo-Sanchez et al. [21] proposed a secure publish-subscribe protocol for medical Wireless Body Area Networks (WBANs) using ABE. In their work, the system is composed by a star-topology network where a smartphone (or a similar device) communicates with various nodes placed over the user's body area, monitoring the user's health conditions. The system allows any node to publish data and to subscribe to data generated from other nodes.

Singh et al. [23] proposed a secure MQTT for IoT, allowing the usage of ABE. In their work, the architecture is composed by one Key Authority (KA), one broker, and several nodes. Every node can be a subscriber, a publisher or both. Every node knows the public key, and a secret key associated with some attributes which describes its characteristics. Every node subscribes to the data it needs for its proper functioning. Both Picazo-Sanchez et al.'s and Singh et al.'s schemes follow the CP-ABE paradigm like ours, but they use a publish-subscribe method, which is unsuitable for our objectives since it introduces too much latency.

Rasori et al. [9] proposed a system for smart cities using ABE. The application streams in real-time data from roads within a smart city. In this work, the architecture is composed by many sensors, a cloud server and users. The sensors capture the data and store it on a cloud server. Paying users can retrieve data from the cloud server for the roads they are interested in with his/her ABE key they have paid for.

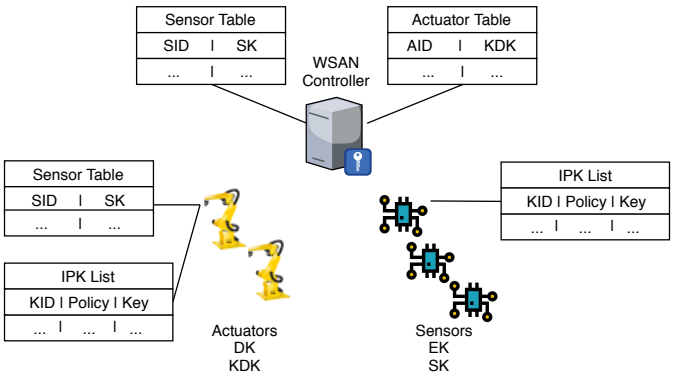


Fig. 2. An overview of fABElous architecture.

## III. Architecture

We assume a low-bitrate *Wireless Sensor and Actuator Network* (WSAN), composed by a set of *sensors* and *actuators*, which exchange encrypted data with each other (Fig. 2). As an use-case example, consider a smart factory with many sensors and actuators which must communicate in a delay-bounded way to implement a real-time application [27]. Given the strict requirements, sensors and actuators must communicate directly through the WSAN. The WSAN inside the smart factory uses IEEE 802.15.4 as a link-layer protocol, which is low-energy and low-bitrate. As a consequence, communications and encrypt/decrypt operations must be as lightweight as possible.

A sensor is a data producer that measures and encrypts some quantity, and then it sends the encrypted data to a set of actuators over the WSAN. An actuator is a data consumer that receives encrypted data from a set of sensors over the WSAN, and then uses it to control some mechanism. The encrypted data received by an actuator could be a command which the actuator executes, as well as a measured data from a sensor which the actuator uses to take a decision. Sensors and actuators are regulated by a *WSAN controller* node belonging to the WSAN. For the sake of simplicity, we keep the "sensor" role and the "actuator" role separated, however a single device may act as both. We assume that the WSAN controller has its own pair of asymmetric keys (e.g., RSA, ECC, etc.) used for digital signature and encryption. In addition, each sensor and each actuator has a unique identifier called, respectively, *Sensor ID* (*SID*) and *Actuator ID* (*AID*), which are assigned by the controller.

### A. Key Distribution Mechanism

In order to satisfy the strict requirements of our model regarding security and messages size, we diminish the use of CP-ABE heavier ciphertext and primitives. Indeed, in fABElous each sensor executes the Encrypt primitive only once for securing multiple data described the same policy. Similarly, each actuator executes the Decrypt primitive only once for extracting data generated by the same sensor and described by the same policy. The basic idea is to distribute symmetric keys using the CP-ABE scheme as a reliable tool to achieve fine-grained multicast. Each sensor encrypts a symmetric key with the CP-ABE Encrypt primitive under a certain policy and broadcasts it to all the actuators. All the actuators will receive the ciphertext, but only few will be able to successfully executes the Decrypt primitive and retrieve the symmetric key. In this way, when the sensor wants to transmit data and apply the aforementioned policy on it, the sensor encrypts such data with the symmetric key instead. In the following, we explain more in detail the procedures that the WSAN controller, the sensors and actuators may execute inside our system.

### B. System Procedures

*1) System Initialization:* The system initialization procedure is executed only once, to start the system. The controller runs the Setup primitive, thus obtaining the master key and the encryption key.

*2) Sensor Join:* The sensor join procedure (Fig. 3) is executed whenever a new sensor joins the WSAN.

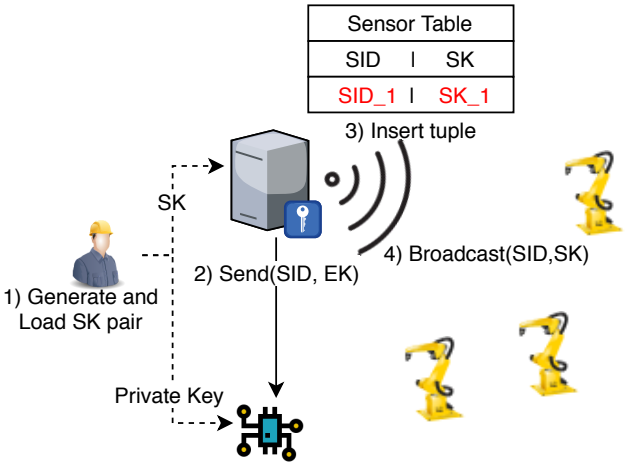


Fig. 3. Sensor join procedure. Dashed lines represent human-device communication.

First, the human operator who is physically deploying the sensor generates a pair of asymmetric keys which the sensor will use for digital signatures. The operator loads the private key on the sensor, and the public key on the controller (step 1). We call *signature key* (*SK*) such a public key. After that, the controller assigns an identifier *SID* to the sensor, and it sends the identifier and the encryption key to the sensor with a signed message (step 2). The controller adds a tuple $\langle SID, SK \rangle$ to a locally maintained *Sensor Table* (step 3). Each tuple in the Sensor Table represents a sensor in the system. Finally, the controller signs and broadcasts the tuple to all the WSAN actuators (step 4). The WSAN actuators add such a tuple to their locally maintained copy of the WSAN Sensor Table.

*3) Actuator Join:* The actuator join procedure is executed whenever a new actuator joins the WSAN.

First, the human operator who is physically deploying the actuator generates a pair of asymmetric keys which the actuator will use for receiving encrypted keys. The operator loads the private key on the actuator, and the public key on the

controller (step 1). We call *key-distribution key* ($KDK$) such a public key. After that, the controller assigns an identifier $AID$ to the actuator, and it generates a decryption key with the $\mathrm{KeyGen}$ primitive, according to the actuator's attribute set. The controller signs the identifier and the decryption key, it encrypts such signed message with the actuator's key-distribution key, then sends the obtained ciphertext to the actuator. The controller adds a tuple $\langle AID, KDK \rangle$ to a locally maintained *Actuator Table* (step 3). Each tuple in the Actuator Table represents an actuator in the system. Finally, the WSAN controller sends the WSAN Sensor Table to the actuator with a signed message (step 4).

*4) New Policy installation:* The new policy installation procedure (Fig. 4) is executed by a sensor to share a symmetric key with some actuators belonging to the WSAN. When a

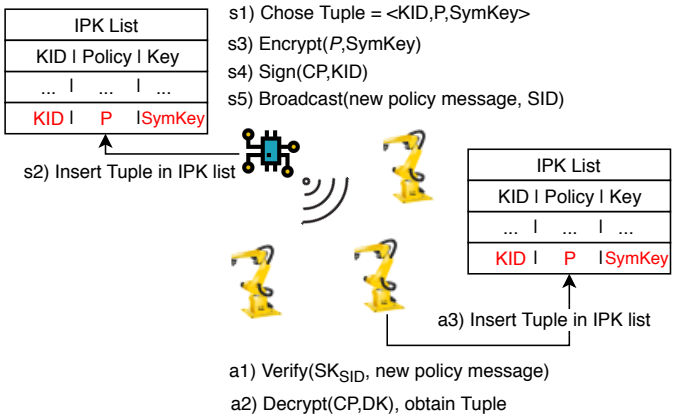


Fig. 4. New policy installation procedure.

sensor performs this procedure for the first time, it creates an *Identifier Policy Key* (IPK) list. This list links a policy $\mathcal{P}$ to a symmetric key $\mathrm{SymKey}$ through a *symmetric key identifier* ($KID$), and this allows a sensor to encrypt data with the advantages of symmetric encryption (faster and with smaller ciphertexts than asymmetric encryption), plus the capability of enforcing an access policy on said data. In other words, the IPK list is composed by one or more tuples in the form $\langle KID, \mathcal{P}, \mathrm{SymKey} \rangle$. The IPK list is a structure owned by each sensor that belongs to the WSAN, and thus, two different sensors will have two different IPK lists. Actuators store a similar IPK list, containing the tuples that the sensors shared with them, by means of this procedure.

The following steps allow a sensor to create a tuple for its IPK list, and to share said tuple with some actuators over the WSAN. The sensor generates an IPK tuple by choosing a policy $\mathcal{P}$, a random symmetric key $\mathrm{SymKey}$ and a random $KID$ (step s1). The sensor adds to its IPK list the tuple generated in step s1 (step s2). The sensor encrypts the symmetric key under $\mathcal{P}$ using the $\mathrm{Encrypt}$ primitive (step s3). Then the sensor signs the concatenation of its $SID$, the ciphertext and the $KID$ (step s4). Throughout the paper we will refer to a signed concatenation of a $SID$, an ABE ciphertext and a $KID$ as a *new policy message*. The sensor transmits the new policy message over the WSAN (step s5). Each actuator inside the WSAN verifies the sensor signature on the new policy message, using the signature key associated to the received $SID$ inside the sensor table (step a1). If the signature is not correct, the message is discarded. Otherwise, the actuator checks if its attribute set $\gamma$ satisfies $\mathcal{P}$. If the attribute set $\gamma$ does not satisfy the policy $\mathcal{P}$, the message is discarded. Otherwise, if the attribute set $\gamma$ satisfies the policy $\mathcal{P}$, the actuator decrypts the ciphertext executing the primitive $\mathrm{Decrypt}$, obtaining the symmetric key (step a2). Finally, the actuator inserts the tuple in its IPK list with the quantities retrieved in step a2 (step a3).

*5) Data Exchange:* The data exchange procedure (Fig. 5) is executed by a sensor to transmit data to one or more actuators in a low-latency fashion inside the WSAN.

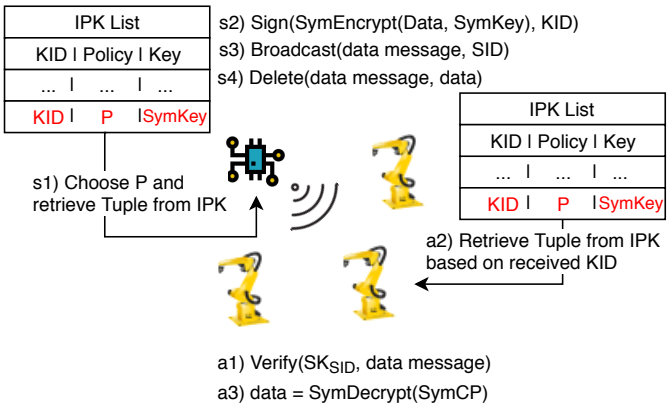


Fig. 5. Data exchange procedure.

The sensor chooses a policy $\mathcal{P}$ and retrieves the associated symmetric key and $KID$ from its IPK list (step s1). If there is no matching tuple in its IPK list, the sensor performs a new policy installation procedure. The sensor encrypts the data using the symmetric key (obtaining $SymCP$), and it signs the concatenation of its $SID$, the ciphertext and the $KID$ (step s2). Throughout the paper we will refer to a signed concatenation of a $SID$, a symmetric ciphertext and a $KID$ as a *data message*. Then, the sensor broadcasts the data message over the WSAN (step s3). Each actuator inside the WSAN, which is interested in the transmitted data, verifies the sensor signature on the data message, by retrieving from the sensor table the signature key associated to the received $SID$ (step a1). If the signature is not correct, the data message is discarded. Otherwise, the actuator retrieves from its IPK list the tuple associated with the received $KID$ (step a2). The actuator uses the latest symmetric key retrieved to decrypt the data message and consumes its content (step a3). Finally, the sensor securely deletes the sensed data (step s4).

### C. Threat Model

The fABElous scheme provides data integrity, confidentiality, and access control. In the following we analyze possible threats and explain how fABElous addresses them.

*1) Eavesdropper:* Eavesdroppers are surely a threat to confidentiality. An eavesdropper can try to gain information by examining the traffic between sensors, actuators and the controller. However, every exchange of information is protected. If an eavesdropper is able to obtain a data message, he cannot access the data since he does not have the symmetric key. Even if he have access to the ABE ciphertext containing that symmetric key, he cannot decrypt it either, because he lacks an ABE decryption key. Even more so, if the eavesdropper intercepts the message exchange between an actuator and the WSAN controller during the actuator join procedure, he cannot retrieve the decryption key since it is safely encrypted with the key-distribution key of said actuator.

*2) Compromised Sensor:* Suppose that an attacker gains complete access over a sensor. Said attacker obtains: (i) the data generated by the sensor from the moment of the compromise on; (ii) the private signature key of said sensor; (iii) the IPK list used by the sensor including the symmetric keys used for data encryption. Note that this attacker cannot, in any way, obtain data generated by other sensors. Each sensor deletes past data securely, so the attacker cannot retrieve it from the sensor. However if an attacker intercepted and stored past transmissions, he would be able to retrieve past data by using the stolen symmetric keys. In order to mitigate this, sensors could periodically refresh the symmetric keys by deleting some tuples from their IPK list and executing again the new policy installation procedure on the same policies. In this way, the attacker cannot retrieve data produced before the last refresh of the symmetric key.

Note that the attacker could also disseminate malicious data authenticated with the signature key of the compromised sensor. In this way, malicious data is accepted by the actuators that receive it. This attack can be thwarted by revoking the signature key of the compromised sensor. We plan to add this functionality to a future version of fABElous.

*3) Compromised Actuator:* Suppose that an attacker gains complete access over an actuator. Said attacker obtains: (i) the private ABE decryption key of said actuator; (ii) the private key-distribution key of said actuator; (iii) the IPK list used by the actuator, including the symmetric keys used for data decryption. Each actuator may delete past data securely after consumption, so the attacker cannot retrieve it from the actuator. However, if an attacker intercepted and stored past transmissions, he would be able to retrieve past data by using the stolen symmetric keys. With this set of information, the attacker can decrypt every past and future data message that the compromised actuator has access to. Note that this does not imply that the attacker has access to all the data generated by the sensors. Indeed, his decryption capabilities are limited by the access privileges of the compromised actuator. If the compromised actuator cannot decrypt some data because its attribute set does not satisfy the policy of such data, then the attacker will not be able to decrypt it as well. This is achieved thanks to ABE technology, which enforces a fine-grained access control even in case of device compromise. The actuator compromise can be completely worked out by revoking its decryption key. We plan to add this functionality to a future version of fABElous.

## IV. Performance Evaluation

In this section we show more in detail the parameters we considered for evaluating fABElous. Data is composed by the combination of $120$ bytes of raw data, plus $4$ byte of KID, plus $4$ bytes of timestamp (to avoid replay attacks). The used digital signature algorithm is ECDSA, which has the benefit of adding a constant size signature of $40$ byte (considering $80$-bit security). For the symmetric key encryption we used AES with $128$-bit keys in CBC mode. The policy used to evaluate CP-ABE communication overhead is a simple, yet effective $\mathcal{P}_1 = (A \,\text{AND}\, B \,\text{AND}\, C)$. We used only AND operators without losing in generality since the specific Boolean operator does not influence the communication overhead. To measure the communication overhead we used the CP-ABE toolkit of Bethencourt et al. [7]. Table I shows the communication overhead of fABElous compared to other schemes.

TABLE I
Transmission Size

| Scheme | Size (bytes) | Overhead (%) |
|---|---|---|
| No security | 120 | 0% |
| Authentication only | 160 | 25% |
| "Naive" CP-ABE | 1.250 | 90% |
| fABElous | 192(+1.122*) | 100%−37.5% |

*Once per policy installation

*No security* scheme refers to sensors transmitting raw data without any kind of protection. *Authentication only* scheme refers to sensors transmitting signed data using ECDSA. *"Naive" CP-ABE* scheme refers to sensors constantly transmitting data encrypted with CP-ABE. The reduction of the fABElous communication overhead over number of transmissions is calculated as:

$$\text{Overhead}(\%) = \frac{1122 + N \cdot 72}{1.122 + N \cdot 192} \cdot 100,$$

where $N$ is the number of data messages sent, 72 is the amount of overhead bytes in each data message, $1.122$ byte is the size of the CP-ABE ciphertext containing an AES key, and $192$ byte is the total size of each data message. As it can be seen from the table, the overhead of fABElous encryption is as big as $100\%$ in the worst case (no data exchange procedure executed after a new policy installation procedure).

Compared to no security, fABElous has an incredible amount of communication overhead, even in its best-case scenario. However, fABElous grants data integrity, data confidentiality, and fine-grained access control, which are three features required by our use case. Compared to authentication only, fABElous has more than twice the amount of communication overhead, even in its best-case scenario. However, in addition to data integrity, fABElous also grants data confidentiality and fine-grained access control, which are two features required by our use case. Compared to "Naive" CP-ABE, fABElous

has less communication overhead since the second data exchange execution ($N \geq 2$). Indeed, in its best-case scenario, fABElous communication overhead is $49\%$ less than "Naive" CP-ABE communication overhead ($N \geq 99$). Furthermore, in addition to data confidentiality and fine-grained access control, fABElous also grants data integrity, which is a feature required by our use case. Fig. 6 shows how fABElous communication

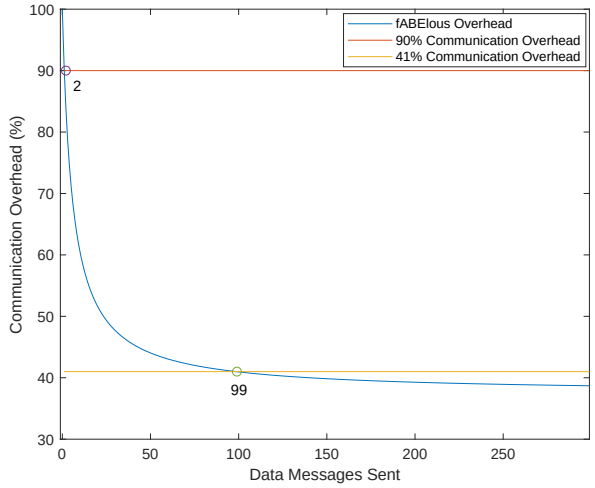


Fig. 6. fABElous communication overhead.

overhead drops with each execution of the data exchange procedure. The communication overhead is lower than $90\%$ after two executions of the data exchange procedure. Furthermore, the communication overhead drops below $55\%$ after 106 executions of the data exchange procedure.

## V. Conclusion

In this paper, we proposed fABElous, an ABE scheme suitable for Industrial IoT applications which aims at minimizing the communication overhead introduced by ABE encryption. We described its architecture, system procedures, and provided an use case example. We analyzed its vulnerability and showed our next research objectives. Finally, we proved how fABElous ensures data integrity, confidentiality, and access control, while reducing the communication overhead of 35% with respect to using ABE techniques naively.

## Acknowledgments

This work was supported by the Italian Ministry of Education and Research (MIUR) in the framework of the CrossLab project (Departments of Excellence), and by the project PRA_2018_81 "Wearable sensor systems: personalized analysis and data security in healthcare" funded by the University of Pisa.

## References

[1] L. Mainetti, L. Patrono, and A. Vilei, "Evolution of wireless sensor networks towards the internet of things: A survey," in *Software, Telecommunications and Computer Networks (SoftCOM), 2011 19th International Conference on*. IEEE, 2011, pp. 1–6.

[2] K. Ashton *et al.*, "That 'internet of things' thing," *RFID journal*, vol. 22, no. 7, pp. 97–114, 2009.

[3] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[4] A. Gilchrist, *Industry 4.0: the industrial internet of things*. Apress, 2016.

[5] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2005, pp. 457–473.

[6] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*. Acm, 2006, pp. 89–98.

[7] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Security and Privacy, 2007. SP'07. IEEE Symposium on*. IEEE, 2007, pp. 321–334.

[8] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *Infocom, 2010 proceedings IEEE*. Ieee, 2010, pp. 1–9.

[9] M. Rasori, P. Perazzo, and G. Dini, "ABE-cities: An attribute-based encryption system for smart cities," in *2018 IEEE International Conference on Smart Computing (SMARTCOMP)*, June 2018, pp. 65–72.

[10] S. Yu, K. Ren, and W. Lou, "FDAC: Toward fine-grained distributed data access control in wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 4, pp. 673–686, 2011.

[11] M. Ambrosin, A. Anzanpour, M. Conti, T. Dargahi, S. R. Moosavi, A. M. Rahmani, and P. Liljeberg, "On the feasibility of attribute-based encryption on internet of things devices," *IEEE Micro*, vol. 36, no. 6, pp. 25–35, 2016.

[12] B. Girgenti, P. Perazzo, C. Vallati, F. Righetti, G. Dini, and G. Anastasi, "On the feasibility of attribute-based encryption on constrained IoT devices," in *Smart Computing (SMARTCOMP), 2019 International Conference on (to appear)*. IEEE, 2019.

[13] S. Farrell, "Low-power wide area network (lpwan) overview," Tech. Rep., 2018.

[14] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Rfc 4944," *Transmission of IPv6 packets over IEEE*, vol. 802, no. 4, 2007.

[15] J. Tosi, F. Taffoni, M. Santacatterina, R. Sannino, and D. Formica, "Performance evaluation of bluetooth low energy: a systematic review," *Sensors*, vol. 17, no. 12, p. 2898, 2017.

[16] B. Latré, P. De Mil, I. Moerman, N. Van Dierdonck, B. Dhoedt, and P. Demeester, "Maximum throughput and minimum delay in IEEE 802.15.4," in *International Conference on Mobile Ad-Hoc and Sensor Networks*. Springer, 2005, pp. 866–876.

[17] O. Georgiou and U. Raza, "Low power wide area network analysis: Can LoRa scale?" *IEEE Wireless Communications Letters*, vol. 6, no. 2, pp. 162–165, 2017.

[18] Y. Ming, L. Fan, H. Jing-Li, and W. Zhao-Li, "An efficient attribute based encryption scheme with revocation for outsourced data sharing control," in *Instrumentation, Measurement, Computer, Communication and Control, 2011 First International Conference on*. IEEE, 2011, pp. 516–520.

[19] Z. Xu and K. M. Martin, "Dynamic user revocation and key refreshing for attribute-based encryption in cloud storage," in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*. IEEE, 2012, pp. 844–849.

[20] J. Hur, "Improving security and efficiency in attribute-based data sharing," *IEEE transactions on knowledge and data engineering*, vol. 25, no. 10, pp. 2271–2282, 2013.

[21] P. Picazo-Sanchez, J. E. Tapiador, P. Peris-Lopez, and G. Suarez-Tangil, "Secure publish-subscribe protocols for heterogeneous medical wireless body area networks," *Sensors*, vol. 14, no. 12, pp. 22 619–22 642, 2014.

[22] L. Touati and Y. Challal, "Batch-based CP-ABE with attribute revocation mechanism for the internet of things," in *Computing, Networking and Communications (ICNC), 2015 International Conference on*. IEEE, 2015, pp. 1044–1049.

[23] M. Singh, M. Rajan, V. Shivraj, and P. Balamuralidhar, "Secure MQTT for Internet of Things (IoT)," in *Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on*. IEEE, 2015, pp. 746–751.

[24] S. Jahid, P. Mittal, and N. Borisov, "EASiER: Encryption-based access control in social networks with efficient revocation," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*. ACM, 2011, pp. 411–415.

[25] M. Ambrosin, M. Conti, and T. Dargahi, "On the feasibility of attribute-based encryption on smartphone devices," in *Proceedings of the 2015*

*Workshop on IoT challenges in Mobile and Industrial Systems*. ACM, 2015, pp. 49–54.

[26] Z. Liu, Z. Cao, and D. S. Wong, "White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 76–88, 2013.

[27] F. Chen, T. Talanis, R. German, and F. Dressler, "Real-time enabled ieee 802.15. 4 sensor networks in industrial automation," in *Industrial Embedded Systems, 2009. SIES'09. IEEE International Symposium on*. IEEE, 2009, pp. 136–139.