Energy-aware key management and access control for the internet of things

Mohamed Mohammedi, Mawloud Omar, Djamila Zamouche, Kahina Louiba, Saliha Ouared, Kenza Hocini

Noname manuscript No.
(will be inserted by the editor)

Energy-aware key management and access control for the Internet of things

Mohamed Mohammedi[1*] · **Mawloud Omar**[2] · **Djamila Zamouche**[1] · **Kahina Louiba**[1] · **Saliha Ouared**[1] · **Kenza Hocini**[3]

Received: 24 May 2019

Abstract The need to establish a secure communication for most applications of Internet of Things (IoT) has become increasingly crucial. Nevertheless, one of the major issues of such networks is how to establish cryptographic keys between all IoT objects to ensure secure data exchange. Several key management schemes have been suggested in the literature to achieve this goal, but they must be revised and innovated, while taking into account the limited resources of IoT objects. Likewise, the IoT presents a system where objects belonging to the physical world, are connected to the Internet, and have the capacity to measure, communicate, and act around all over the world. Nevertheless, some information available to IoT objects is private, hence the need to ensure an access control whose aim is to guarantee that the information be accessible only to those whose the access is allowed. In this paper, we propose a scheme involving two basic modules, namely key management, and access control. The key management module is designed to ensure the symmetric key generation for the IoT objects in a completely distributed manner without resorting to a central authority. In contrast, the access control module is used to impose an access control policy so as to prevent unauthorized access to services provided by a particular IoT object. The latter is ensured thanks to the existing cooperation between the TGOs servers of different domains in the IoT. Through assessments based on security analysis, we demonstrate that the proposed scheme is more secure than the existing ones. Simulations were also performed to validate the proposed solution by comparing it with some relevant concurrent schemes. The obtained results are not only encouraging, but also very favorable for the proposed scheme.

Keywords Internet of Things · Security · key management · Symmetric key · Access control

E-mail:* Mohamed Mohammedi
mohamed.mohammedi@univ-bejaia.dz
00213 698 41 69 98

[1] Laboratoire d'Informatique Médicale, Faculté des Sciences Exactes, Université de Bejaia, 06000 Bejaia, Algérie.
[2] LIGM, ESIEE Paris, Université Gustave-Eiffel, Noisy-le-Grand, France.
[3] Unité de Recherches LaMOS (Modélisation et Optimisation des Systèmes), Faculté des Sciences Exactes, Université de Bejaia, 06000 Bejaia, Algérie.

1 Introduction

Internet of Things (IoT) is a major evolution, which falls under the continuity of the recent developments of information and communication technologies and embedded systems. The World Wide Web has progressed considerably over the last few years from a calculators network to a network of personal computers, then towards a network, which integrates any communicating device [20] [37]. IoT is a set of smart objects, which are connected [42] through a wireless communication means, using an optical medium or radio frequencies. Each object has the ability not only to collect data, but also to transmit it to the sink via the Internet or by using other communication technologies so that it will be treated and visualized on dedicated dashboards [15]. These smart objects, which are regarded as the basic platform of the context of IoT, are the objects of everyday life, such as vehicles, televisions, refrigerators, washing machines, buildings, etc [6] [28]. Each IoT objects is equipped with electronic components such as radio communication supports, processors, sensors, actuators, etc. The emergence of such networks shows their important role which covers numerous application areas including smart homes, smart cities, smart farms, energy, transport, health, industry, agriculture, logistics, etc [9] [27].

Internet of Things is envisaged as an extension of a next generation of Internet [13] [26]. The rise of the IoT can be observed in many areas from the most personal to the most industrial. This has led to huge benefits like a better energy management, improved health monitoring, etc. In such a type of system, any IoT object is potentially connected to the Internet and capable to communicate with other objects [17]. Nevertheless, this generates not only conventional attack threats on networks and data, but also it generates the emergence of new threats, which affect the communications. Furthermore, given the dynamic nature of IoT environment, where objects can join or leave the system, protecting data flows from unauthorized access presents a challenge. Consequently, taking into account the nature of the information exchanged between the IoT objects, security is a very important attribute for the communications between IoT objects to guarantee the reliability and privacy of communications [41], also for the generated data to guarantee protection of the confidential data against unauthorized access. For that reason, it is compulsory to have a secure key management scheme to secure transactions between the IoT objects, and it is necessary to establish mechanisms managing and controlling the access to the data of connected devices,as well as to the device itself. In this context, numerous research is focused on the integration of the key management in the IoT [8]. Due to various resource constraints related to the IoT object such as energy, memory space, processing capabilities, etc., security in such networks poses different challenges. Some existing schemes attempt to address these constraints, but these schemes are different from one to another in terms the techniques used for sharing keys between IoT objects, their security level, their taking into account the characteristics of IoT objects, as well as the heterogeneity of these entities. Also, the integration of access control in the IoT has been the subject of several studies in recent years. However, meeting the constraints of this technological revolution in security approaches remains a challenge.

In this paper, we propose a scheme involving two modules to overcome the security issues, which we have already raised before in the IoT environment. First, a key management module

that allows IoT objects to establish their ECC key pairs, which will be used subsequently to generate their shared symmetric keys. In other words, this module permits each two IoT object to cooperate and compute their symmetric cryptographic key in a completely distributed manner without resorting to a central server. To the best of our knowledge, this contribution is the first in which the symmetric cryptographic technique is applied upon the trust graphs, which is extremely involved not only in the security of data exchanges between the system objects, but also in the reduction of the number of symmetric keys generated in the network. Second, the access control module which is an improvement of the Kerberos authentication scheme. The aim of this module is to ensure scalability by distributing this standard version to different domains, as well as adding a trust relationship between them to ensure the security of this scheme. Furthermore, we have integrated the RBAC model (Role-Based Access Control) [33] [43] [48], in the access server of the second module to allow access to users or prevent them from this access after a verification of the access rights and the role associated with the user. Through the security analysis, we have proved the robustness of the proposed scheme against some malicious known cryptographic attacks. The performance evaluation of this scheme shows a clear improvement in the results compared to the original version of the standard Kerberos scheme. Furthermore, simulation obtained results demonstrate that the proposed scheme provides very encouraging results in comparison with the concurrent ones.

The remaining of the paper is structured as follows. In Section 2, we briefly outline some relevant key management, and access control schemes in the framework of the IoT. In Section 3 and 4, we give separately the detailed description of the proposed scheme, and some security analysis. In Section 5, we provide the obtained results following the performance evaluation of the proposed scheme with its two basic modules. Finally, in Section 6, we conclude this study and we provide some future research directions.

2 Related Work

In this section, we review many recent and pertinent security schemes in the IoT, which we recapitulate in two groups, namely: (1) Key management schemes, and (2) Access control schemes.

2.1 Key Management Schemes

IoT is one of the major communications developments, which have been seen in these last few years, the latter allows the IoT objects to be connected between them and with Internet [25]. Therefore, to provide an appreciable security level of these communications, several key management schemes have been suggested in the literature to provide a shared secret symmetric key between two or more IoT objects, typically used for security purposes [11] [12]. IoT objects, which constitute the network, have strict constraints depending on the resources such as energy, computational power, memory, etc. Consequently, few key management proposed schemes are appropriate for the IoT. Most of the existing schemes [12] [25] use public

cryptography key to compute the encryption key. However, public key cryptography generates a high computational cost, which is one of the main concerns for resource constrained objects as stated in [29]. In [12] [25], the authors have proposed centralized key management schemes, where the encryption key is generated from a central server. In [18], the authors have proposed a key management scheme based on the use of a node, which is unconstrained in resource to unload the heavy cryptographic operations of a resource constrained node. However, the authors have assumed that these unconstrained resource nodes are trustworthy, but no mechanism is established to detect the case where these nodes are compromised. This aforementioned technique is also used in [12]. In [21] [22] [35] [39], the authors have proposed a group key management scheme. However, none of them allow to overcome the 1 *affects* n phenomenon [44]. In [35] [39], the authors have presented their model by using a tree topology, to improve the efficiency in terms of processing time, as well as the energy consumption. The authors in [21] [22] exploit Elliptic Curve Cryptography (ECC) [50] [49], which is a lightweight cryptographic solution suitable for securing resource constrained objects. In [10], the authors have proposed a self-certified key management scheme based on ECC to provide mutual authentication, in their scheme zero knowledge proof was applied. In the scheme proposed by Chen et al. [38], each node is elected to be a server node. After the node election to be a server node, it will generate an encryption key space for other nodes in the system. In addition, two nodes share the same cryptographic key whether they are associated with the same server node. The Elliptic Curve Diffie-Hellman algorithm (ECDH) [51] and Elliptic Curve Cryptography (ECC) based on implicit certificates are exploited in [24]. Sciancalepore et al. [24], have proposed a distributed key management scheme in the IoT where two nodes cooperate to compute a symmetric key after they have authenticated each other. Hong et al. [7], have proposed a key management scheme for outsourced access control in the cloud computing. To diminish key generation cost by merging repeated data resources, Hong et al. have proposed a key generation algorithm based upon RST (Resource Set Tree). As well, to check a user authorization to gain accesses to outsourced in cloud computing, the authors have proposed key assignment algorithm based upon hierarchical CRT (Chinese Remainder Theorem). Esposito et al. [1], have proposed a set of methods to ensure confidentiality with end-to-end assurances. This is done through group-based keys in a distributed cluster Key Management framework.

2.2 Access Control Schemes

The illegitimate access to confidential data and physical systems in IoT network can have a negative impact upon our daily lives [32]. With increasing attention to solutions, which limit access to services and resources in IoT, there has been an amplified effort in access control research in such a network. In [16], Xue et al. have proposed an access control system, which allows secure communications in IoT environment. The proposed access control model, can be used by users to access buildings using their own smartphones. Indeed, Xue et al.'s model functions using a protocol named SPCL (Smart Phone Controlled Lock), through a server as a central controller, which gives orders to the lock to open or not. In [30], Ye et al. have proposed an authentication and access control scheme. Indeed, the proposed ECC-based au-

thentication serves to ensure the communication security between users and sensor nodes. In contrast, the proposed access control model is based on attribute, which is used to manage authorization decisions and ensure fine-grained access control. In [14], Patel et al. have proposed a capacity-based authentication and access control protocol (CBAC), whose objective is to ensure secure authorization in IoT environment. In [31], Pereira et al. have proposed a mobile agent access control scheme, which is based on the RBAC model. The main purpose of Pereira et al.'s scheme is to guarantee the secure exchange of clinical information between different health institutions, which belong to the same trust circle. Indeed, this is ensured by a strong access control of mobile agents. In [32], Ramos et al. have proposed a capacity-based access control model in a distributed architecture. The access control logic is embedded in the device, which has capabilities to obtain, process, and forward information to other entities and objects without requiring for a central entity PDP (Policy Decision Point). The capacity concept was originally presented as a token, which gives the owner permission to access an entity whenever. In [19], Bairagi, and Chakroborti, have proposed an access control scheme based on trust. The devices, which want to request services from other devices must be connected through an intelligent gateway called Service Discovery and Analysis (SDA) module. When a device requests a service, the SDA finds the set of devices, which can provide that service based on the trust value of the requestor and the service provider criteria. In [36], Anggorojati et al. have proposed an access delegation mechanism with security considerations for IoT networks or federated machine-to-machine communication networks. The proposed security model is based primarily on capability-based context aware access control. Indeed, the access delegation process is accomplished by using context information integration, a capacity propagation mechanism, and secure capacity propagation in federated IoT environments. In [34], Seitz et al. have proposed an authorization system and access control procedures with different entities, which cannot have the same access rights. The authorization engine used in Seitz et al.'s scheme consists of two components, namely access control system and an assertion issuing system. The former is used to produce policy-based access control decisions to grant access to a user. While the latter is used to code the authorization decision as an assertion, performs the policy evaluation, and publishes authorization statements for the user's access to resources. In [23], Rivera et al. have proposed a unified access control scheme for intelligent agents, IoT devices, as well as hybrid elements. Rivera et al.'s scheme relies on the application of user-managed access (UMA) to allow users to authorize arbitrary tries to access their resources, while providing access control. This mechanism is provided at the authorization server level, which in its turn provides authorization data and confirmation in token format to gain access to the protected resource. In [2], Cao et al. have designed a new keyword index to withstand the keyword guessing attack from both access and search patterns. After that, Cao et al. have proposed an attribute-based encryption scheme, whose objective is to support an improved fine-grained access control search. In [3], Nasiraee and Ashouri-Talouki have proposed an anonymous and decentralized attribute-based encryption in the standard model (ABE). In order to check the user's attributes anonymously Nasiraee and Ashouri-Talouki's scheme relies on cryptographic accumulators. Afterwards, to guarantee the ABE access control against unauthorized users, the authors have comprised the accumulator in the ciphertext. Besides, to keep the

privacy of the policy against the Public Cloud Server (PCS) Nasiraee and Ashouri-Talouki have proposed a decentralized policy obfuscation method. In [4] Yan et al. have proposed an access control scheme called IoT-FBAC (Function-based Access Control scheme in IoT), which relies on Identity-based Encryption scheme. The aim of Yan et al's scheme is to avoid over-privilege access to applications from accessing unauthorized functions.

After a thorough review of the literature, we revealed that numerous schemes have been proposed to address the key management issues in the IoT environment [5]. Most of these schemes aim to integrate; (1) the heterogeneity of the objects that the IoT can interconnect, while taking into account the computing resources of these objects (e.g., memory rate, computing power, storage space, etc.); and (2) cryptographic mechanisms to ensure both privacy, security requirements and usefulness of the service provided for the IoT environment. Consequently, the reviewed schemes differ from the techniques used for cryptographic key sharing between the IoT objects. Also, by their degree of security, their taking into account of the characteristics and the heterogeneity of the IoT objects. To sum up, each of the reviewed key management schemes has interesting advantages in some aspects, but also some weaknesses. These latter may prove to be assets or difficulties depending on the cryptographic techniques that will be used to secure communications between the network entities. As IoT objects have a resource constraint, so in most cases they cannot perform the various cryptographic operations for secret key generation. The schemes proposed in [18] [38] [12] [24], this has been taken into account, unlike those, which are proposed in [39] [22], where the authors did not take this constraint into consideration. As regards to the examined schemes relating to access control, we have noticed that some of them are more efficient than others. We revealed that the centralized trust based scheme proposed in [16], suffers from security pitfalls, does not provide scalability, and has a high execution time. However, the energy consumption, storage, and communication costs are reduced. The distributed trust based schemes proposed in [19] [36] [32] are vulnerable to cryptographic attacks. As well, the energy consumption, execution time, and storage cost are high. Nevertheless, these schemes provide the functionality of scalability, and offer a reduced communication cost. Two other schemes based on distributed trust proposed in [14] [31] guarantee the scalability and the resistance against cryptographic attacks, their energy consumption is reduced in [14]. Nevertheless, that which is proposed in [31] suffers from a high execution time and energy consumption, nonetheless its communication and storage costs are reduced. As regards the authorization-based schemes proposed in [23] [34] they are not scalable, and present some performance limits in terms of energy consumption, execution time, and storage cost. Besides, the scheme proposed in [34] provides resistance against some malicious known cryptographic attacks, contrary to the scheme proposed in [23]. To sum up, through this critical study, we were able to apprehend the major issues surrounding both the key management and access control in the IoT environment. To mend the shortcomings of the aforementioned work it appeared to us that a scheme which involves two modules (key management and access control), which would be based on ECC and would not be based on a central server could be a good solution to the aforesaid issues.

3 The Proposed Scheme

In this section, we give the novelty and our main contributions to the proposed scheme. Likewise, we introduce the considered network model, as well as we give a detailed description of each module, which is involved in the proposed scheme.

Internet of Things is envisaged like a next-generation extension of the Internet, the objects of such an environment are connected to each other. A large number of applications require a highly secure network to avoid holes in communications. Therefore, the establishment and sharing secret keys between the communicating objects is necessary. In consequence, the execution of a robust key management algorithm proves to be essential to secure communications between the IoT objects. An extensive review of the existing literature has helped us to better understand all the key management schemes in the IoT, which have been suggested in the last six years. Nevertheless, the prior proposed schemes are in fact not effective because they all rely on complex cryptographic algorithms, resulting not only in high computational and storage loads, but also in a high communications load. Additionally, in terms of security the majority of them are vulnerable to different types of known cryptographic attacks, which directly affect the privacy of the exchanged data. Besides, most of them don't ensure the scalability criterion. In the proposed scheme, we have taken into account all shortcomings of the reviewed schemes and we propose a new and scalable key management algorithm based upon trust graphs in the context of IoT. The key management module provides the following important properties: (1) it is completely distributed where the key management is established through a trust graph; (2) it allows IoT objects to have a key pair (private and public key), which will be used to generate symmetric keys in a distributed manner without resorting to a central server; (3) it guarantees the scalability criterion; (4) it overcomes the 1 *affects* n phenomenon [44], where the IoT object leaving the network will never be able to decrypt future messages exchanged in the network and the IoT object joining the network will not be able to decrypt the old messages; (5) it achieves better security against various known attacks, with lower storage, energy consumption, communication, and computation costs.

The main idea of key management module is to integrate symmetric cryptography technique upon trust graphs to satisfy the requirements of the IoT system in terms of security. Based on trust graphs, the proposed scheme becomes scalable and considerably reduces the number of symmetric keys generated in the network. During system initialization, each object i locally computes its ECC key pair, which will be used in symmetric key generation process. Thus, the proposed scheme allows two trusted IoT objects to cooperate so as to compute their symmetric keys by using their ECC key pairs after they mutually authenticate each other. We consider the communication network of the IoT as an unoriented graph denoted by $G = \langle X, U \rangle$, where X and U represent the set of vertices and edges, respectively. The graph vertices represent the IoT objects and the existence of an edge, which connects a to b means that the object a trusts b and it can share a symmetric key with it. We illustrate in Fig. 1, the trust graph model of the key management module. The symmetric key generation between the objects are based on the trust transitive relationship, i.e., if a trusts b

and b trusts c, then a can trust c (see Fig. 2) [46] [45]. Consequently, the object a and c can cooperate to compute their symmetric keys in order to secure their future communications.

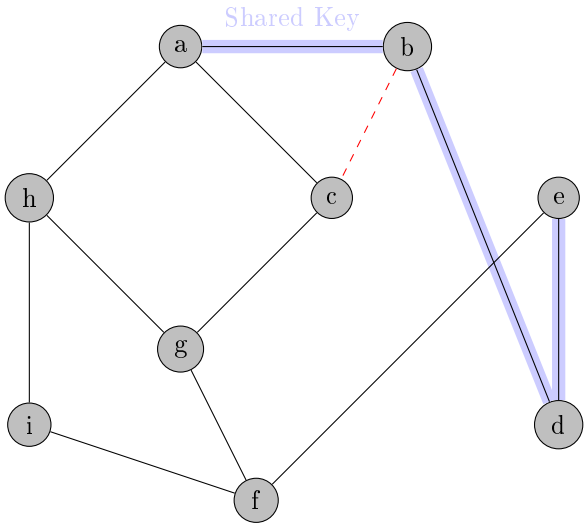

Fig. 1: Trust graph model.

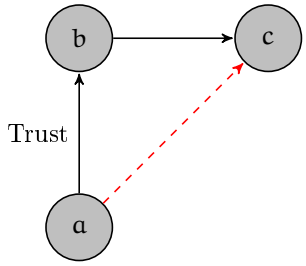

Fig. 2: Transitive trust relationship.

3.1 Key Management Module Algorithm

In this subsection, we describe in detail the key management module based on trust graphs, so as to mend the security and performance shortcomings of the previous schemes described above. Accordingly, the key management module is extensible or retractile in terms of the number of IoT objects, which integrate the networks. This module operates in five basic phases, namely: system initialization, mutual authentication with the session key agreement, communication, joining the system, and removal from the system. Before describing in detail the different phases, which constitute this module, the meanings of all parameters used by the algorithm relating to this module are summarized in Table 1.

Table 1: Notations used in key management module

Symbol	Meaning
ID_i	The object i's identity
T_i	The object i's timestamp
ΔT	The valid time interval of the transmission delay
F_p	A prime field
$E_p(a, b)$	An elliptic curve equation with order n
n	A large prime number
G	The base point with the order n over $E_p(a, b)$
r_a/r_b	A random number chosen by the object a/b from $[1, n-1]$, respectively
$\langle d_i, Q_i \rangle$	The object i's private and public keys
K	Shared secret symmetric key between object a, and b
$\langle \cdot \rangle_K$	Encryption operation with symmetric key K
H	A collision free one-way secure hash function
KDF	The key derivation function
$\oplus$	Bit-wise exclusive-or (XOR) operation
$\|$	Concatenation operation
$\langle +, -, \cdot \rangle$	Elliptic curve point addition, subtraction and multiplication

3.1.1 System Initialization Phase

In this phase, all the IoT objects agree on different parameters to be taken into account during the execution of the computation process of the shared symmetric key between the two communication parties. The detailed steps of this phase are described as follows:

- **Step-1:** all objects publicly agree in a public manner on an elliptic curve equation $E_p(a, b)$ with order n defined on a finite field F_p. Furthermore, the IoT objects also agree on a base point G over the elliptic curve.
- **Step-2:** each object chooses a random number $d_i \in [1, n-1]$ as a secret key. Afterwards, each of these objects derives its public key by computing $Q_i = d_i \cdot G$. Consequently, each network object has a key pair $\langle d_i, Q_i \rangle$, which will be used in the symmetric key generation process.

3.1.2 Mutual Authentication with Key Agreement Phase

In this phase, we explain how two IoT objects, namely I_a and I_b, can authenticate each other and generate a symmetric cryptographic key K by using Elliptic Curve Diffie-Hellman (ECDH) algorithm. Each system object has a list of identifiers of all the objects to which it trusts in order to share with them a symmetric key to secure the future transactions which will be done between them. This list of identifiers of each object is denoted by $U_i = \{Id_1, Id_2, \cdots, Id_m\}$, where m is the number of objects with which the object i shares a symmetric key. The detailed steps of the mutual authentication process with the session key agreement are described in the following:

- **Step-1:** initially, I_a selects a random number $r_a \in [1, n-1]$, and computes some parameters as follows: $\eta = r_a \cdot G$, $\xi = r_a \cdot d_a$, $\alpha = \xi \cdot G$, $\nu = r_a \cdot Q_b$, $\gamma = \alpha \oplus \nu$, $\vartheta = H(ID_a \| \alpha \| \eta \| T_a)$. Afterwards, I_a sends the message request $\langle ID_a, \eta, \gamma, \vartheta, T_a \rangle$ to I_b.

- **Step-2:** upon receiving the request message $\langle ID_a, \eta, \gamma, \vartheta, T_a \rangle$ from I_a, I_b firstly checks the validity and the freshness of the request message by computing the difference $S = |T_b - T_a|$, where T_b is the current timestamp of I_b, in the case where $S > \Delta T$, a replay attack could be suspected. Therefore, I_b rejects I_a's login request. Otherwise, if $S \leqslant \Delta T$, I_b performs the following operations, so it computes $\alpha = \gamma \oplus d_b \cdot \eta$, after the α deduction, I_b computes $\vartheta' = H(ID_a \| \alpha \| \eta \| T_a)$, and it compares the result to ϑ already received. If both of them are not equal I_b rejects I_a's login request and the authentication process in its turn will be canceled. Otherwise, I_b selects a random number $r_b \in [1, n-1]$, and keeps it temporarily in its locally, then it computes $\psi = r_b \cdot Q_b \oplus \alpha$, $Z = H(ID_a \| ID_b \| \alpha \| \psi \| T_b)$. Finally, I_b sends the request message $\langle ID_b, \psi, Z, T_b \rangle$ to I_a.
- **Step-3:** upon receiving the request message $\langle ID_b, \psi, Z, T_b \rangle$ from I_b, I_a checks the validity and the freshness of the request message by computing the difference $S = |T_a - T_b|$, if $S > \Delta T$, I_a computes $r_b \cdot Q_b = \psi \oplus \alpha$, $Z' = H(ID_a \| ID_b \| \alpha \| \psi \| T_b)$, and compares the computed value Z' and the already received one Z. If both of them are equal, then I_b is authenticated by I_a. Therefore, the latter computes the symmetric key K by using the KDF function such that $K = KDF(ID_a \| ID_b \| \rho \| T_a \| T_b)$, where $\rho = r_b \cdot Q_b \cdot \xi = r_a \cdot r_b \cdot d_a \cdot d_b \cdot G$. Afterwards, I_a computes $\phi = r_a \cdot Q_a \oplus r_b \cdot Q_b$, and sends the request message $\langle ID_a, \phi, Auth = H(K, \phi \| r_a \cdot Q_a) \rangle$ to I_b.
- **Step-4:** upon receiving the request message $\langle ID_a, \phi, Auth = H(K, \phi \| r_a \cdot Q_a) \rangle$ from I_a, I_b computes $r_a \cdot Q_a = \phi \oplus r_b \cdot Q_b$, and checks $Auth$ to confirm not only the accuracy of the received request message, but also the authentication of I_a. If the above check passes correctly, then I_a is authenticated by I_b. Thus, the latter computes the same symmetric key K such that $K = KDF(ID_a \| ID_b \| \rho \| T_a \| T_b)$, where $\rho = r_a \cdot Q_a \cdot r_b \cdot d_b = r_a \cdot r_b \cdot d_a \cdot d_b \cdot G$. Fig. 3 summarizes the different phases of the key management algorithm.

3.1.3 Communication Phase

For that an IoT object i could communicate with a particular IoT object j, these two communication parties must share a symmetric key. To compute the latter, these two IoT objects must have mutual confidence from one to another. This mutual trust can be direct or indirect, we summarize in the following, the communication principle by using these two trusty types.

- **Communication using a direct trust:** each two system objects which have a direct mutual trust (e.g., a and e as illustrated in Fig. 4) directly computes their symmetric key following the key computation process defined in section 3.1.2 without using the transitive relation of trust. Afterwards, the two objects could use their shared symmetric key to secure the communications, which take place between them. Consequently, in the proposed scheme, each two IoT objects can compute a symmetric key, which will be shared between them.
- **Communication using indirect trust:** when the object a wants to communicate with b (see Fig. 4), while a and b do not share a symmetric key between them, so this message will borrow one path from a to b, this path represents a trust chain between a and b. The object a randomly chooses one of the objects with which it shares a symmetric key

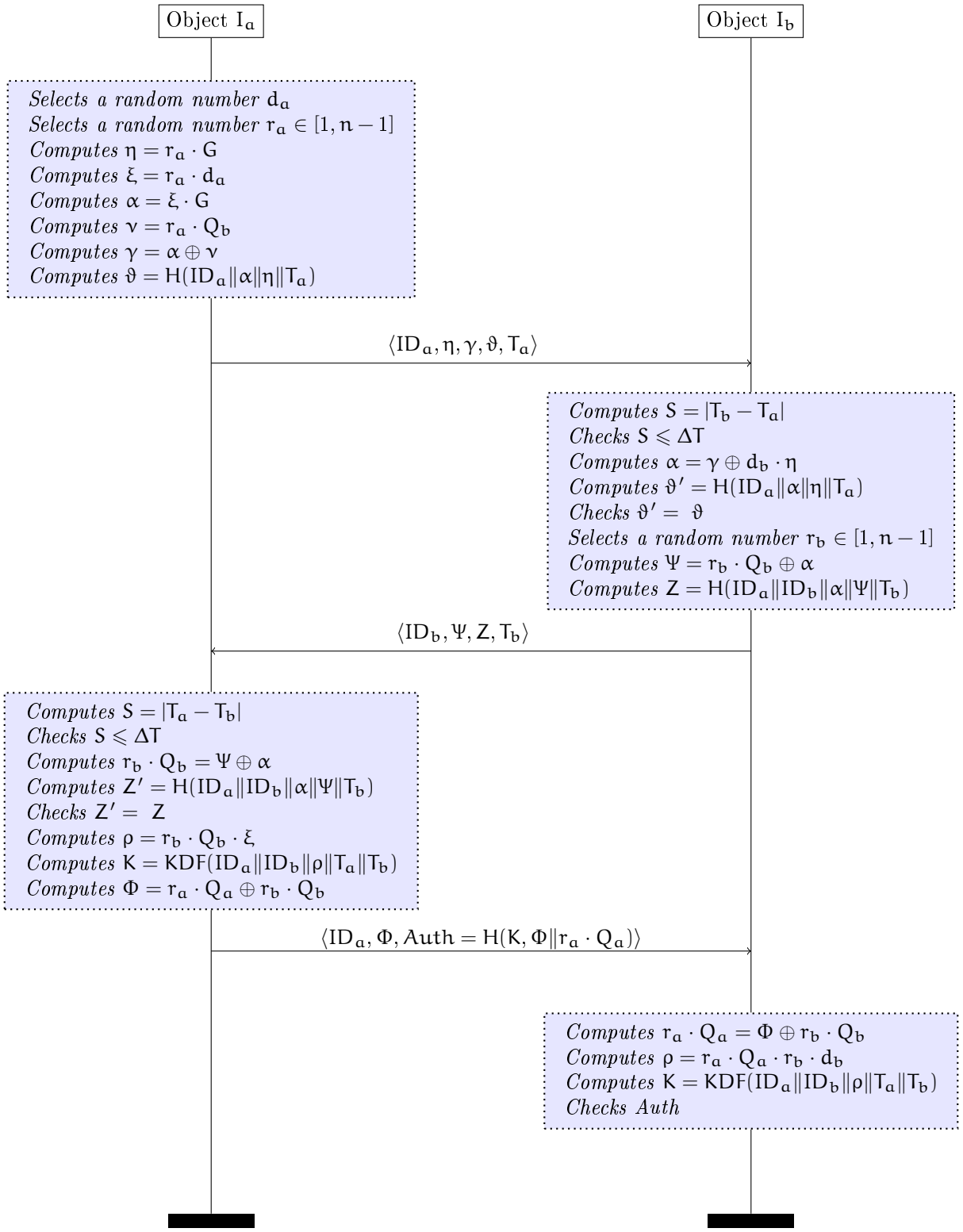

Fig. 3: The key management module algorithm.

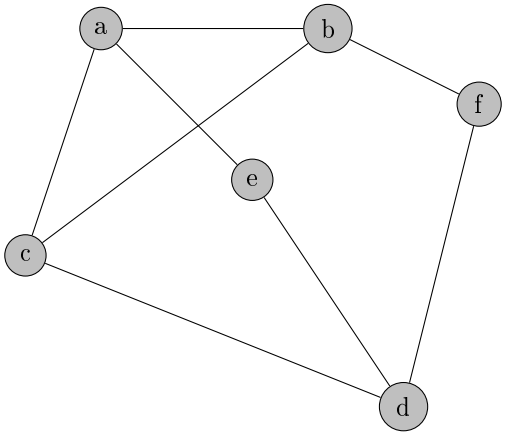

Fig. 4: Communication phase.

and trust it. Assume that $\mathfrak{a}$ chooses $\mathfrak{e}$, so $\mathfrak{a}$ encrypts its message with the symmetric key it shares with $\mathfrak{e}$, then sends it to $\mathfrak{e}$, which in its turn decrypts it with the same key. Then, $\mathfrak{e}$ sends this message to $\mathfrak{b}$ after having encrypted it with the key it shares with it. Upon receiving this message, the object $\mathfrak{b}$ decrypts it with the key which it shares with $\mathfrak{e}$. By the transitive trust principle, the object $\mathfrak{a}$ trusts $\mathfrak{e}$ and the object $\mathfrak{e}$ trusts $\mathfrak{b}$, then $\mathfrak{a}$ can trust $\mathfrak{b}$ (see Fig. 2). Consequently, for future communications between $\mathfrak{a}$ and $\mathfrak{b}$, both of them will compute a shared symmetric key among them.

3.1.4 Joining the System Phase

This operation occurs when a new object integrates the system. When a new object wants to join the system, it must send an integration request to the system to one of the objects, which have already integrated into the system. Upon receiving the request, if the member object believes that the new object is trustworthy, therefore, the new object and the member object compute a secret key after they have authenticated each other. After then, this new object can communicate with the other system objects based on the transitive trust principle. From that, the new object becomes a member of the system. Obviously, the object which joins the system will not be able to decrypt the old exchanged messages with the key that it shares with one of the system member objects.

3.1.5 Removal From the System Phase

When a network object considers that one of the objects with which it shares a secret key is no longer trustworthy, the shared symmetric key between them will be revoked and the malicious object will leave the system. Therefore, this object will not be able to decipher the future messages exchanged in the network. In the case where this malicious object belongs to the trust chain which connects several member objects, in this case the latter explores another trust chain.

In the ongoing, we explain the novelty and our main contributions for the access control module, which is involved in the proposed scheme. As well, we introduce the network model of the second module, and we describe in detail the access control module algorithm.

IoT is a mixture of countless objects and technologies, which make up an heterogeneous network. Objects in such an environment are often empowered to perform different services and access resources and data from other objects using secure communication. The latter plays a very important role in realizing some of these services. This requires supporting both a good authentication and access control mechanism to prevent unauthorized access to the services of IoT objects. The standard Kerberos scheme is effective for authenticating and controlling user access to resources and services of IoT objects. Nevertheless, since Kerberos is only a centralized scheme, and whenever a user wants to use a service of a particular object, it must involve the authentication server. The latter suffers from processing power limitation, so that Kerberos scheme has poor expandability, and this demonstrates that Kerberos is not suitable for largescale networks. It is for these reasons that we have thought to extend the standard Kerberos by the distributivity of this scheme. To achieve this goal, we propose a Kerberos enhancement, which consists of integration of trust principle, control of access rights, and a user's role. The standard Kerberos scheme distributivity allows each user in a certain domain to access the data of each object using the means of delegating access rights and a role-based verification (RBAC). Consequently, it is on these principles that we based to develop a proposal to address the issues raised in the standard Kerberos scheme.

The main purpose of the access control module is to provide the authentication and access control services for users who wish to access object services in IoT environments. To put it another way, this module allows users who have the right to access the services of a particular object, as well as prevent others who do not have the access right. Fig. 5, shows the network model considered in the access control module. When a user i wants to access a particular service of an IoT object j, he asks the authentication server AS_i of his domain for a TGT_i ticket in order to access the TGO_i server. Afterwards, using the received TGT_i ticket, the user i, requests the TGO_i server, an access ticket TGO_i to use the service of a particular object j. The TGO_i server checks the access rights, the role associated with this user, as well as the validity of the TGO_i ticket, so as to answer him with a request, which confirms or denies the access to the requested object. In the case where the user i has the access right, the TGO_i server delivers to him the requested ticket (TGO_i ticket) to access the services of a particular IoT object requested by the user i. Finally, when the access to this object is successfully performed, the object launches a protocol for executing a particular service in the object j. This scenario is executed when the object belongs to the same domain as that of the user i. Otherwise, if the object belongs to a different domain, a trust relationship must be established between the TGO_i servers of different domain.

3.2 Access Control Module Algorithm

Our contribution is based on Kerberos, which is a centralized scheme, it does not provide the scalability criterion. Indeed, it causes a large load of key sharing and an authentication

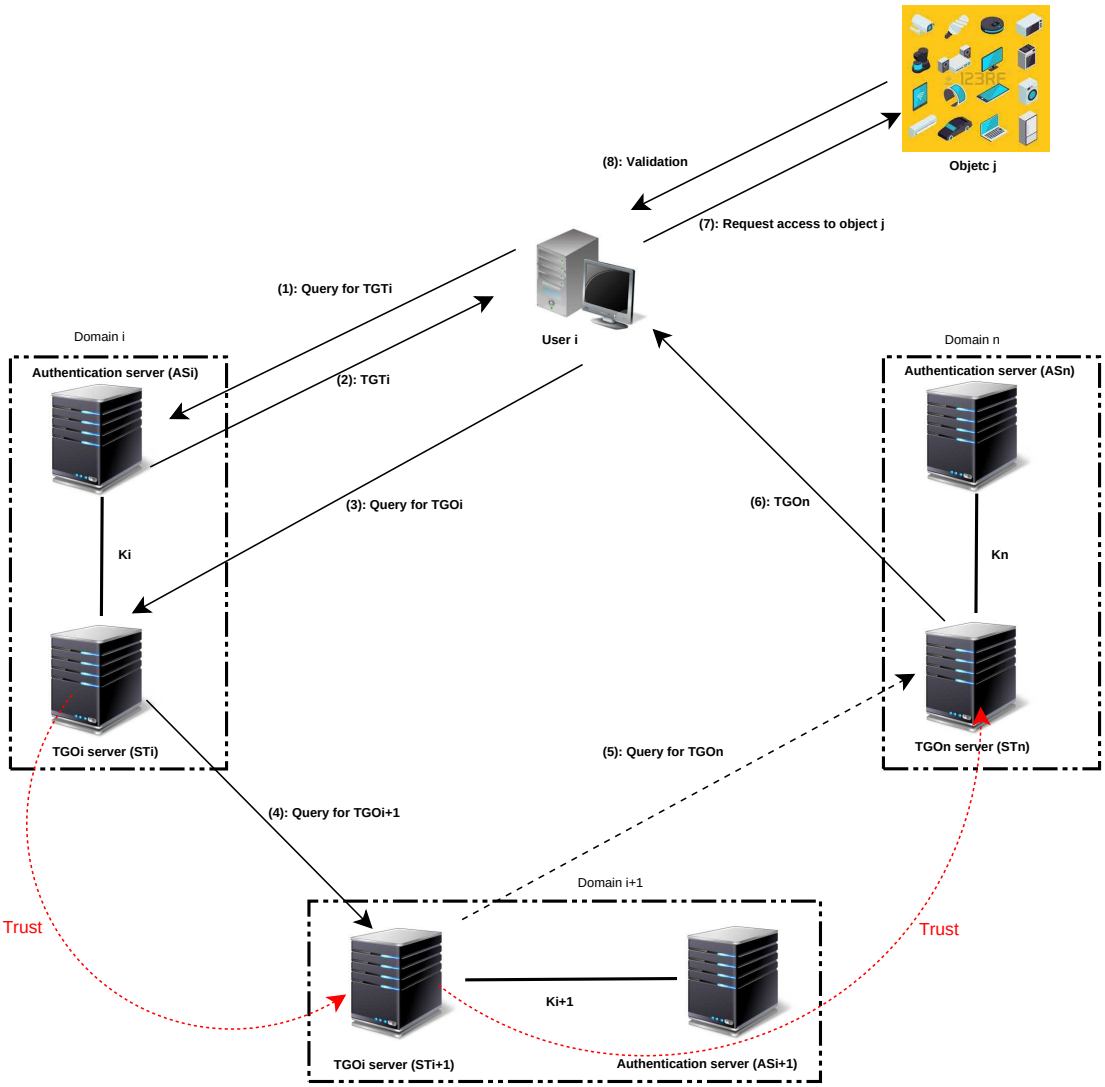

Fig. 5: Network model of the access control module.

server saturation. In addition, standard Kerberos does not provide the reliability criterion because if the authentication/TGO server fails, the entire system will fail. On account of these weaknesses, we thought about implementing a distributed solution, which inherits the goodness of Kerberos standard. In the upgraded version, we have assigned to each domain, which can be a hospital, company, university, governmental organization, etc., a Kerberos scheme. This is necessary to provide both authentication and access control service, which is ensured by RBAC model (based on the user's role). Therefore, the major interest of the RBAC access control model is to allow the authorization or prohibition of access to the requested object at the local level (i.e., within its domain).

The access control module can be divided into three phases, namely: user authentication, obtaining access ticket to the object j, and access to the object j. Furthermore, the connection phase between different domains is a special case, which only executes when a user requests access to an object located in another domain different from its original domain. The following subsections detail the different phases, which constitute the access

control module. In Table 2, we summarize the notation and terminology used through the algorithm of the second module.

Table 2: Notations used in access control module

Symbol	Meaning
ID_i	The user i's identity
ID_j	The object j's identity
ID_{AS_i}	Authentication server identity of domain i
ID_{ST_i}	TGO Server identity of domain i
TGT_i	Ticket Granting Ticket of domain i
TGO_i	Ticket Granting Object of domain i
AS_i	Authentication server of domain i
S_i	Shared symmetric key between TGO_i server and user i
SK_{ij}	Shared symmetric key between user i and object j
K_i	Shared symmetric key between authentication server AS_i and TGO_i server
PW_i	The user i's access password
PW_j	The object j's access password
ED	The expiry date of a session key
ΔT	The expected valid time interval of the transmission delay
T_i, T_i'	The user i's current timestamps
T_{TGO}	The TGO_i server current timestamp
T_j	The object j's current timestamp
R_i	The user i's role

3.2.1 User Authentication Phase

In this phase, the user i identify himself near the authentication server AS_i to obtain the TGT_i ticket, which authorizes him to make requests to TGO_i server that interests him.

- **Step-1:** before accessing the services of a particular IoT object, the user i must authenticate with the authentication server AS_i of his domain. For this fact, the user i, firstly, sends to the authentication server AS_i an authentication request which include its identity, authentication server's identity, as well as that of the TGO_i server, which interests him, such that $\langle ID_i, ID_{AS_i}, ID_{ST_i} \rangle$.
- **Step-2:** upon receiving the authentication request $\langle ID_i, ID_{AS_i}, ID_{ST_i} \rangle$ from the user i, the authentication server AS_i checks the validity of user i's identity and checks also if this user has the right to access to TGO_i server. If both are correct, the authentication server AS_i generates a session key S_i, which will be shared between the user i and TGO_i server to establish secure communication between them. Then, it fixes the expiration date ED_{S_i} of this session key. Subsequently, it computes the TGT_i ticket, which allows this user to make requests to TGO_i server such that $TGT_i = ID_i + S_i + ED_{S_i}$. The latter will be encrypted by using the symmetric key K_i, which is shared between the authentication server AS_i, and TGO_i server. Finally, the authentication server AS_i responds to the user i with the request $\langle S_i, TGT_i \rangle_{PW_i}$. The user i's authentication phase is depicted in Fig. 6.

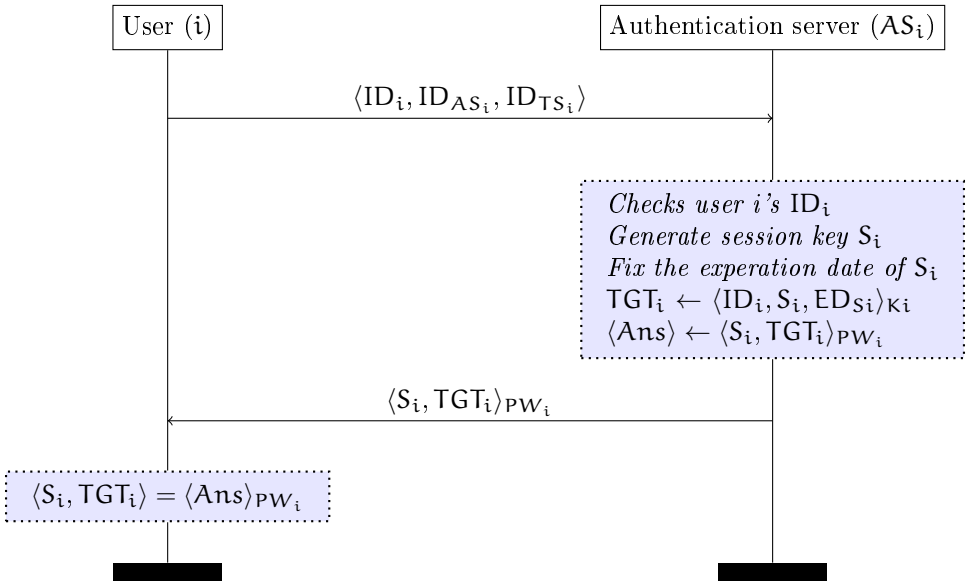

Fig. 6: User authentication phase.

3.2.2 Phase of Obtaining the Access Ticket to Object j

Once the user i acquires its TGT_i ticket, and wants to use the service of object j, then he sends a demand of TGO_i ticket to the TGO_i server.

- **Step-1:** the user i sends to server TGO_i of his domain an access message request $\langle ID_i, ID_{ST_i}, ID_j, R_i, TGT_i, \langle T_i \rangle_{Si} \rangle$, where T_i indicates, the current timestamp of the user i.
- **Step-2:** upon receiving the access request $\langle ID_i, ID_{ST_i}, ID_j, R_i, TGT_i, \langle T_i \rangle_{Si} \rangle$ from the user i, the server TGO_i checks whether the object j belongs to its domain. If yes, it decrypts the TGT_i ticket by using K_i to get the session key S_i to decrypt T_i, and checks its freshness to see if it's close to the current timestamp T_{TGO}. So, the TGO_i server computes $T_{TGO} - T_i \leqslant \Delta T$, where ΔT indicate, the expected valid time interval of the transmission delay. If $T_{TGO} - T_i > \Delta T$, the TGO_i server rejects the request. Otherwise, it checks the expiry date of the TGT_i ticket and evaluates the access rights and the role associated with the user i. If the user i has access rights, the TGO_i server generates an access ticket TGO_i. The latter is composed of a session key SK_{ij}, which will be shared between the user i and the object j and an expiry date $ED_{SK_{ij}}$ of this session key, user i's identity, object j's identity, as well as user i's role. Afterwards, the obtained ticket $TGO_i = ID_i + ID_j + SK_{ij} + R_i + ED_{SK_{ij}}$ is encrypted with the object j's password PW_j. Finally, the TGO_i server responds to the user i with a request $\langle SK_{ij}, TGO_i \rangle_{S_i}$. The phase of obtaining the access ticket to the object j is depicted in Fig. 7.

3.2.3 Access Phase to Object j

This last phase represents the communication phase between the user i and the object j, which represents the service provider. The steps which constitute this phase are the following:

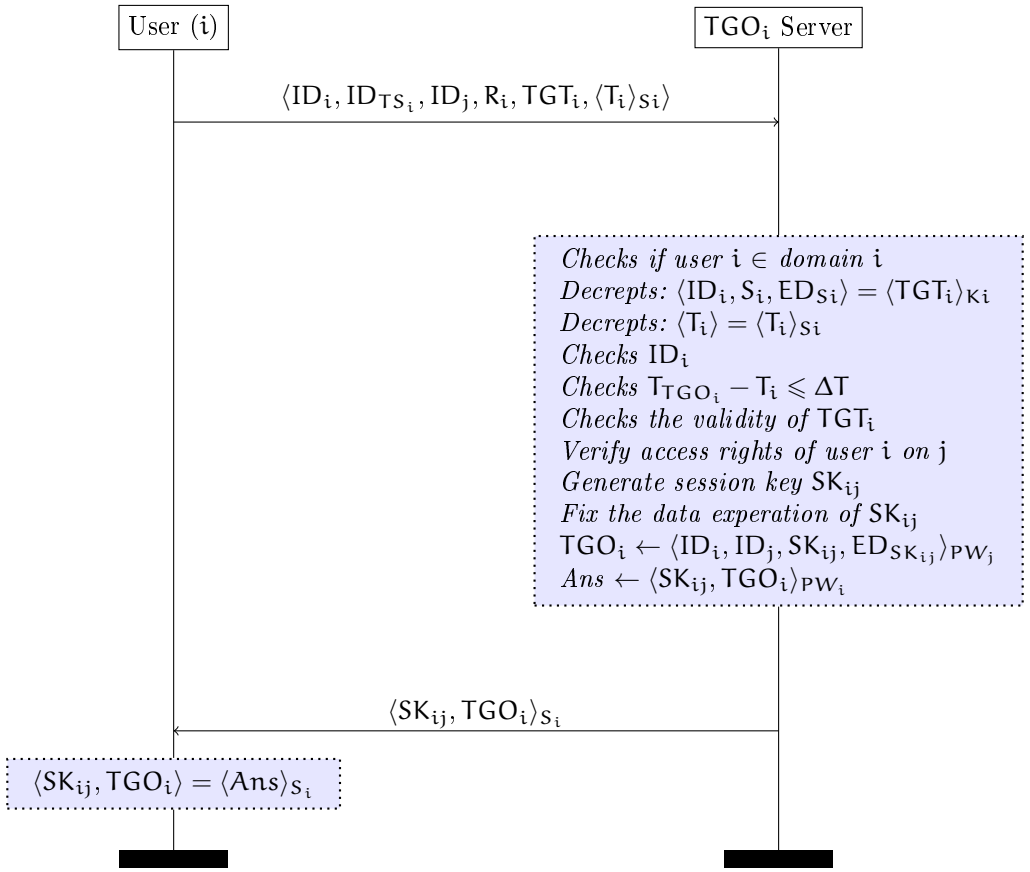

Fig. 7: Phase of obtaining the access ticket to object j.

- **Step-1:** after receiving the request $\langle SK_{ij}, TGO_i \rangle_{S_i}$ from the TGO_i server, the user i decrypts the bloc by using the symmetric key S_i to get the session key SK_{ij}, which will be shared between him and the object i. After the user i extracts his access ticket TGO_i to access to the particular object j in his domain, he sends an access request $\langle ID_i, ID_j, TGO_i, \langle T_i' \rangle_{SK_{ij}} \rangle$ to the object j.
- **Step-2:** after receiving the access request $\langle ID_i, ID_j, TGO_i, \langle T_i' \rangle_{SK_{ij}} \rangle$ from the user i, the object j decrypts the TGO_i ticket by using PW_j to have the session key Sk_{ij}. Afterwards, it checks if the TGO_i ticket is still valid. If it true, it checks whether the user i's timestamp T_i' sent by the user i is close to the object j's current timestamp T_j. If $T_j - T_i' \leqslant \Delta T$, the object j allows the user i to access to its service. The access phase to the object j performed by the user i is depicted in Fig. 8.

3.2.4 Connection Phase Between Different Domains

In this phase, a trust relationship is defined between TGO_i servers of different domains, allows a user i to access a particular service of an object j in another domain different from its original domain.

- **Step-1:** initially, the user i authenticates himself near the authentication server AS_i in his domain to obtain a TGT_i ticket to access to a TGO_i server. When the user i gets his

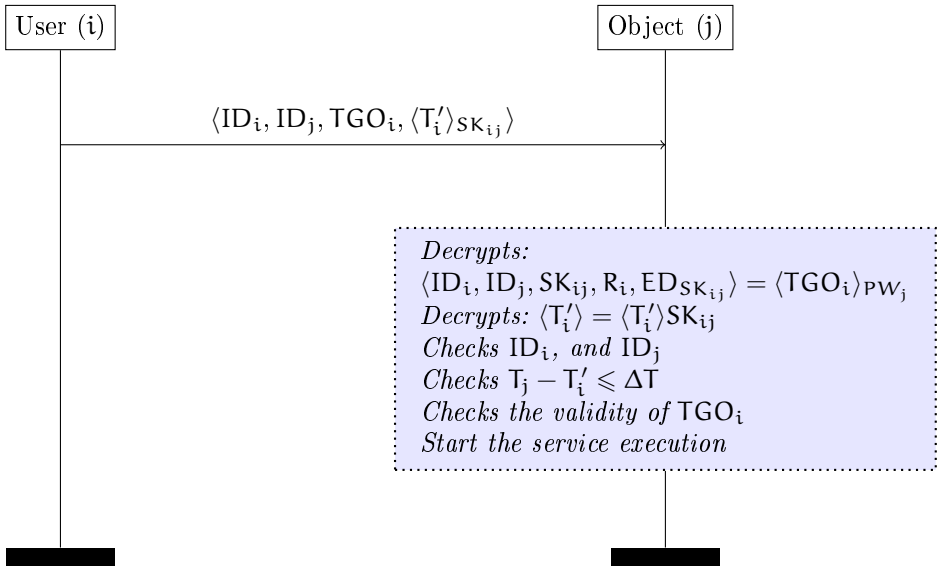

Fig. 8: Access phase to the object j.

ticket TGT_i for the server TGO_i in his domain, the server TGO_i check whether the object j belongs to his domain or not. In the positive case, the access control will be ensured by the three phases that we have already seen before without having recourse to the phase of connection between the different domains. Otherwise, the TGO_i server redirects the request to another TGO_{i+1} neighbor server from another domain, which it trusts. The TGO_{i+1} server of external domain check the access rights, and the components of TGT_i ticket, if they are adequate, then the TGO_{i+1} server of external domain sends a ticket TGO_{i+1} to the user i to confirm his/her access right to the object j. Otherwise, the TGO_{i+1} server redirects the request to another neighbor TGO_{i+2} server from another domain, to whom he trusts, and so on. The steps of this phase are depicted in Fig. 9.

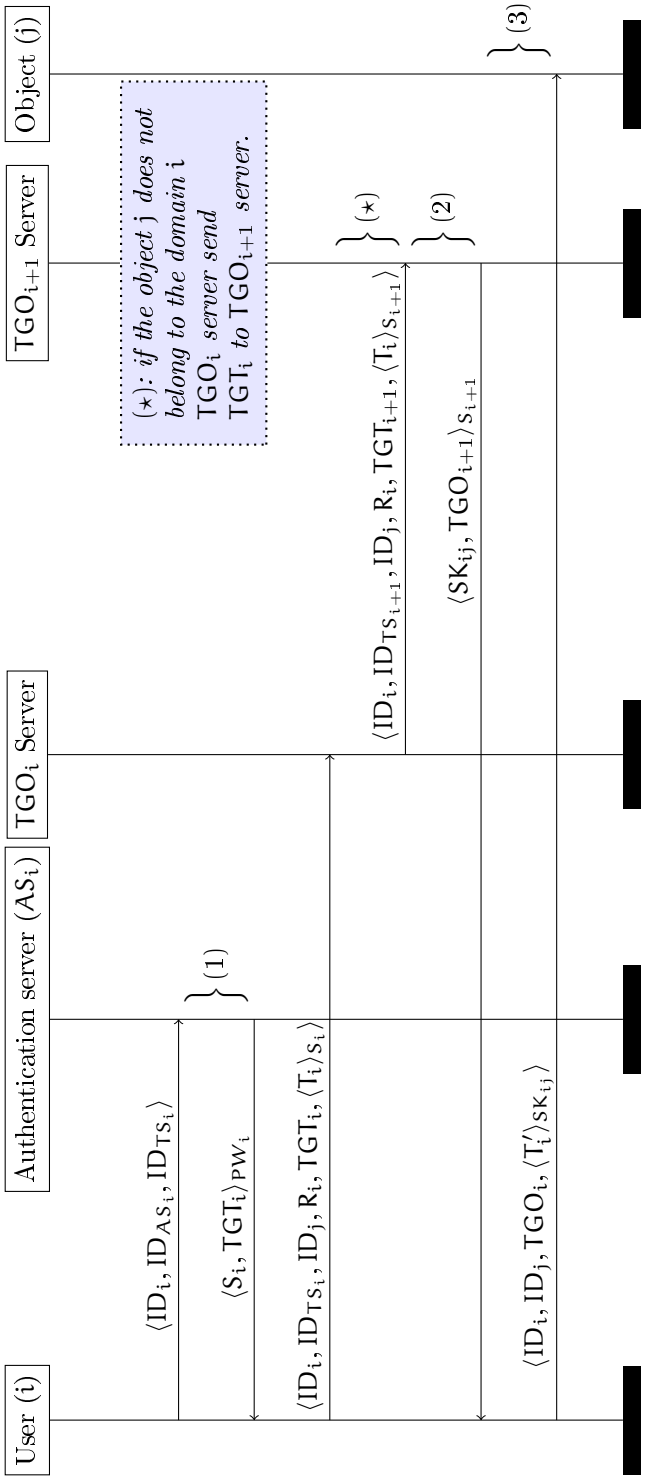

Fig. 9: Connection phase between different domains.

Note that the operations (1), (2), and (3) depicted in Fig. 9 are the same operations of the three phases which we have already explained before in Sections 3.2.1, 3.2.2, and 3.2.3, respectively.

4 Security Analysis

In this section, we analyze the security of the proposed scheme to demonstrate the level of its resistance against well-known malicious cryptographic attacks.

4.1 Key Management Module

Proposition 1 *The key management module can prevent against replay attack.*

Proof Since the exchanged messages have a valid time interval ΔT, an adversary will not be able to reuse messages of a previous session run of this module in an another future session. Assume that an intruder intercepts the request message $\langle ID_a, \eta, \gamma, \vartheta, T_a \rangle$ sent from I_a to I_b and tries to reuse it for another future session. When I_b receives this request message, it will check its freshness by computing the difference $S = |T_b - T_a|$, it will find that it is greater than the valid time interval for the transmission delay ΔT, since the message was not sent at the time T_a. For that reason, replay attack is impossible to be performed in the proposed scheme.

Proposition 2 *The key management module can prevent against denial of service attack.*

Proof Assume that an adversary intercepts the message request $\langle ID_b, \psi, Z, T_b \rangle$ sent from I_b to I_a and replaces it with $\langle ID_b, X, Z, T_b \rangle$, after then sends it to I_a, where X is any random point on the elliptic curve. Upon receiving this request message, I_a computes $Z' = H(ID_a \| ID_b \| \alpha \| X \| T_b)$, and compares the result (Z') to the already received value of Z, but they are different. Therefore, an intruder cannot overwhelm I_a with unnecessary requests for the purpose of conducting an identity or session theft attack. As a result, the proposed scheme can withstand denial of service attack.

Proposition 3 *The key management module can prevent against man-in-the-middle attack.*

Proof Assume that an adversary intercepts the message request $\langle ID_a, \eta, \gamma, \vartheta, T_a \rangle$ sent from I_a to I_b, it cannot get η, since computing r_a is equivalent to solve Elliptic Curve Discrete Logarithm Problem (ECDLP) [40]. What is more, he will not be able to get γ, because he will not be able to compute α since r_a and d_a are secret and are only known by I_a. In addition, the adversary cannot get ν, because the computation of the secret value r_a is equivalent to solve the ECDLP problem. When the intruder intercepts the request $\langle ID_b, \psi, Z, T_b \rangle$, it replaces ψ with X, where X is a random value chosen by the intruder, then sends the request $\langle ID_b, X, Z, T_b \rangle$ to I_a. Upon receiving this request message, I_a computes $Z' = H(ID_a \| ID_b \| \alpha \| X \| T_b)$ and compares it with Z and finds not the same result. Consequently, the authentication fails and the computed key will be revoked.

Proposition 4 *The key management module can prevent against impersonation attack.*

Proof An adversary cannot authenticate itself as I_a to share a symmetric session key with I_b. The adversary intercepts the request message $\langle ID_a, \eta, \gamma, \vartheta, T_a \rangle$ forwarded by I_a, it will not be able to obtain η, because the computation of the secret value r_a is equivalent to solve the Elliptic Curve Discrete Logarithm Problem (ECDLP). As well, γ cannot be obtained because it will not be able to compute α, since r_a and d_a are secret and are only known by I_a, and cannot also get ν, as the computation of the secret value r_a is equivalent to solve the problem of ECDLP.

Proposition 5 *The key management module can prevent against modification attack.*

Proof Assume that a malicious object wants to alter the exchanged data between the two objects I_a and I_b, it will have no means to do it. Indeed, it is difficult to carry out a modification attack since I_a and I_b use the one-way secure hash function to guarantee the integrity of the exchanged data during all the transactions done between the two communication parties.

4.2 Access Control Module

Proposition 6 *The access control module can withstand man-in-the-middle attack.*

Proof Assume that an intruder wants to stand between two communication entities to pretend to be near to each other. However, the intruder will not have the opportunity to pretend to be one near the other under penalty of being spotted by the two communication entities. Because the intruder will not be able to retrieve sensitive information or modify intercepted messages from the TGT_i and TGO_i tickets, which are encrypted by using session keys, which are known only by the two communication entities. Consequently, such attack is impossible to implement in the proposed scheme.

Proposition 7 *The access control module can withstand Sybil attack.*

Proof Assume that an intruder wants to impersonate multiple valid users to send wrong requests to TGO_i server or object j at a given time. To do this, the intruder must first identify himself near the authentication server AS_i to obtain a TGT_i ticket to request a TGO_i ticket from the TGO_i server. However, the intruder cannot pass this step because the authentication server AS_i detects it. We assume that the intruder passes this step, so, he must generate several tickets to hope to deceive the TGO_i server or the object j. However, by receiving these tickets, the server TGO_i and object j checks the validation of these tickets through the decryption operations. Thus, the request messages of TGO_i ticket or service request will be rejected and the attack is detected.

Proposition 8 *The access control module can detect a DoS attack.*

Proof Assume that an intruder attempts to overwhelm the TGO_i server or the object i with useless queries for the purpose of leading an impersonation or session theft attacks. The intruder will never be able to authenticate himself or obtain any confidential information. Because if the number of the attempts to access TGO_i server or object i with a wrong

password (or session key) exceeds the predefined number, the TGO_i server or object i block the operation of a login session. Consequently, this allows us to justify that the proposed scheme resists against such attack.

Proposition 9 *The access control module can withstand impersonation attack.*

Proof If an intruder wants to impersonate a user i he must first identify himself near the authentication server AS_i. However, the latter detects message invalidity and rejects the login request. Consequently, when an intruder wants to impersonate a user i, he will not have the opportunity to do it, based on what was previously demonstrated in sybile attack.

Proposition 10 *The access control module can withstand replay attack.*

Proof Assume that an intruder has intercepted a valid ticket request from the TGO_i server already forwarded by the user i. If the intruder reuses the request $\langle \mathrm{ID}_i, \mathrm{ID}_{\mathrm{ST}_i}, \mathrm{ID}_j, \mathsf{R}_i, \mathrm{TGT}_i, \langle \mathsf{T}_i \rangle_{\mathrm{S}i} \rangle$, the TGO_i server detects this attack by checking the user i's timestamp T_i of the received request, which will be rejected if $\mathsf{T}_{\mathsf{TGO}} - \mathsf{T}_i > \Delta\mathsf{T}$. For the same reason, the intruder cannot reuse the exchanged messages between the user i and the object j. Because, the latter detects such attack by checking the freshness of the received request by the validity of the condition $\mathsf{T}_j - \mathsf{T}_i' \leqslant \Delta\mathsf{T}$. So, an intruder will not be able to reuse messages from one session in another session.

Proposition 11 *The access control module can bypass brute force attack.*

Proof In the access control module passwords used by the user i, the object j, as well as the different servers are for single use. In addition, the expiry date of the different tickets issued during the execution of the proposed scheme is quite short, which usually dates eight hours. Besides, the stolen ticket can be canceled prematurely, if necessary. Moreover, the stolen ticket is valid only if it is used from the ticket owner machine IP address. Consequently, the periodic renewal of passwords and tickets does not allow an intruder to easily and quickly find these secrets, so the brute force attack is in this case ineffective.

5 Experimental Validation

5.1 Simulation Environment and Parameters

In this subsection, we expose and discuss the simulation parameters which we have considered. In order to evaluate the performance of the proposed scheme, we have developed simulations under Java programming language. For the performance evaluation of the proposed scheme, the simulations were performed on a machine, which operate upon Windows $64-\mathsf{bit}$ system, running over an Intel(R) Core (TM) $\mathsf{i3}-2350\mathrm{M}$ CPU @ 2.30 GHz 2.30 GHz processor and 4 GB RAM memory. The simulations were done with 210 users. Finally, as for the obtained results are the average of 25 simulated independent iterations. In what follows, we present and discuss the obtained simulation results of both the key management and access control modules according to some major metrics which we will discuss later.

5.2 Key Management Module

During the validation of the key management module, we have focused on four main performance metrics, namely: communication cost, processing time, storage cost, and energy consumption. Through the communication cost, we compute the number of bits sent by each object $\mathtt{i}$ during the key establishment process to see the complexity degree of the key management module in a number of messages broadcasted for the establishment of this process. By means of the processing time, we evaluate the execution time of this module during the establishment key process to see at which stages this module is reactive to improve the real time criteria. Moreover, via the storage cost we evaluate the ability of the object $\mathtt{i}$ to store the different parameters, which it will need in the computing process of the symmetric key. Finally, through consumed energy, we analyze the amount of energy consumed by each object in the system to measure its lifetime, as well as that of the entire network. As mentioned above, simulations of the key management module have been done under Java programming language, with objects which have the same initial energy capacity $\mathtt{E}_0 = 3$ Joule. So as to compute the energy consumed by each object during the key establishment process, we have opted for the radio energy consumption model proposed by Heinzelman et al. [47]. The latter is used to compute the energy consumed during the transmission and reception of message ($\mathtt{pk}$) from object $\mathtt{i}$ to $\mathtt{j}$, located at a distance $\mathtt{d}$.

In order to evaluate the performance of the key management module, we have chosen to compare the obtained results with those of the concurrent schemes, namely that of Porambage et al. [22] with its two versions, Haripriya and Kulothungan [10], and Sciancalepore et al. [24] by adopting the same energy model. We have measured the performance metrics that we have considered according to the ECC key size: P-128, P-160, P-192, and P-224 bits. The increase in the ECC key size is essential for seeing the ECC keys influence on the four metrics listed above, because ECC keys are needed to achieve the desired security and functionality. In what follows, we present the interpretation of the obtained results of the first module.

5.2.1 Communication Cost

In this subsection, we compare the performances of the key management module, as well as those of the concurrent schemes in terms of communication cost. Fig. 10 illustrates the communication cost variation in function of ECC key size.

As we can see from the simulated graph, the size of messages exchanged between objects increases with increasing of the ECC key size, this is with all schemes, including ours. From the results, it was inferred that the proposed scheme gives better results compared to the concurrent ones. This is because it performs in three communication round and the amount of the exchanged data is less than those of the concurrent schemes. Indeed, to ensure the authentication process between objects, the proposed scheme uses only the hash functions. However, the concurrent schemes use not only much hash functions and ECDSA signature algorithm, but also all of them performs in four communication round, which in part increases the communication cost.

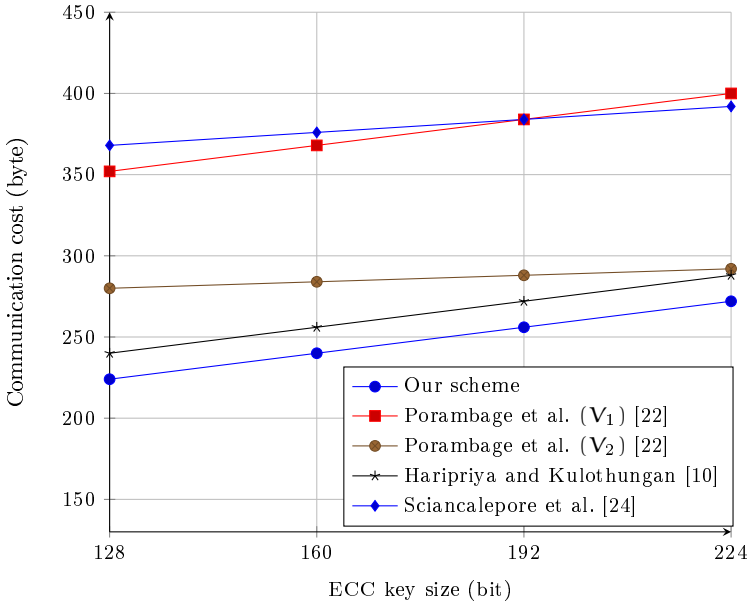

Fig. 10: Overall communication cost in function of ECC key size.

5.2.2 Processing Time

In this subsection, we were interested in comparing the processing time of the first module, and the concurrent schemes. Fig. 11 illustrates the variation of the overall processing time in function of the ECC key size.

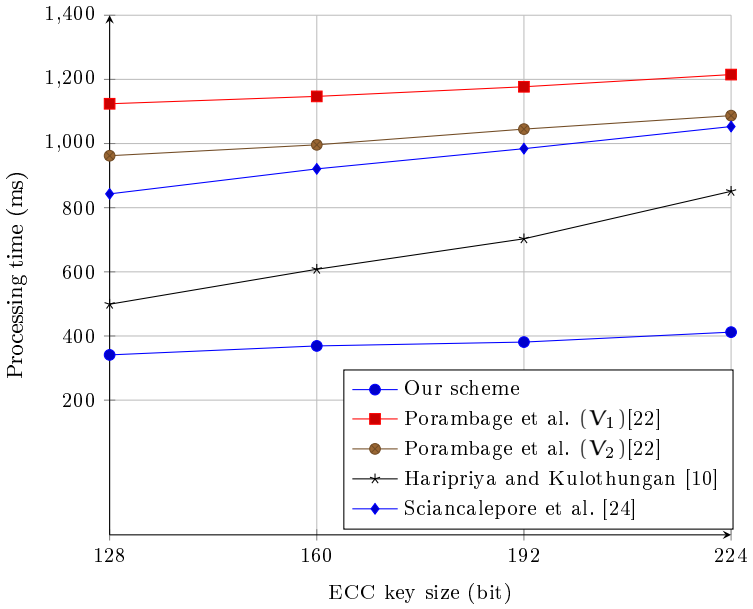

Fig. 11: Overall processing time in function of ECC key size.

From Fig. 11, it can be seen that the processing time increases by increasing ECC key size for both the proposed scheme and the concurrent ones. The result relating to the proposed

scheme is very satisfactory and better than those obtained by the concurrent ones. Besides, the performances of the proposed scheme are higher compared to other schemes with a basic processing time of 412 (ms). Because of the fact that the key management module uses non-complex security mechanisms, on the other hand, the concurrent schemes rely on complex processing operations, especially, Elliptic Curve Digital Signature Algorithm (ECDSA), certificates, etc., resulting in a high computational, and communication load.

5.2.3 Storage Cost

In this subsection, we discuss the storage cost variation in function of the ECC key size for this module, as well as for those of the concurrent schemes. Fig. 12, illustrates, the obtained results in terms of storage cost of each object.

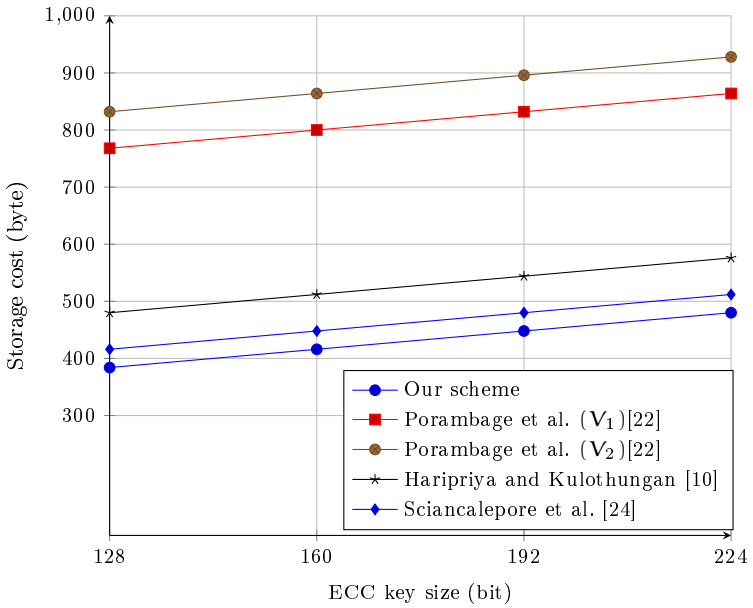

Fig. 12: Storage cost in each object in function of ECC key size.

As expected, the storage cost on each object in the proposed scheme and the concurrent ones increases when the ECC key size increases. It is shown that the proposed scheme indicates acceptable results compared to those presented by the concurrent ones. Indeed, in the proposed scheme, the IoT objects store only two parameters, but, the scheme proposed by Porambage et al. [22], Haripriya and Kulothungan [10], and Sciancalepore et al. [24] require several parameters to be stored in each IoT object. Consequently, the results are clearly favorable for the proposed scheme.

5.2.4 Consumed Energy

In this subsection, we discuss the energy consumed variation in function of the ECC key size for the first module, as well as for those of the concurrent schemes. Fig. 13 illustrates the obtained results in terms of energy consumption on each IoT object.

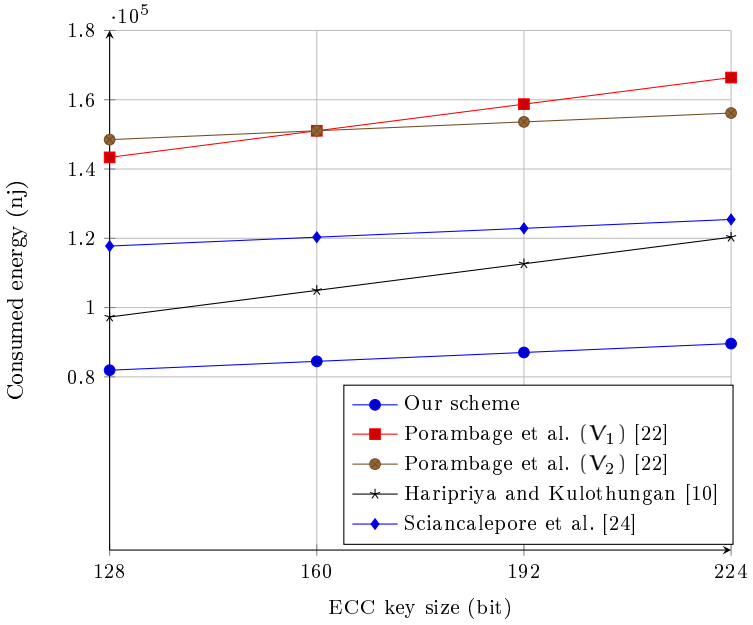

Fig. 13: Energy consumption on each IoT object in function of key size ECC.

From the simulated graph, the energy consumption with the proposed scheme is lower and acceptable than those of the concurrent ones. When the ECC key size increases, the consumed energy in the proposed scheme is more significant than the comparable ones. This is because the key management module has smaller exchanged message size, less communication load and needs less hash operations. However, the concurrent schemes based upon complex algorithms, especially, cryptographic algorithms, which are expensive in CPU resources.

5.3 Access Control Module

In order to validate the access control module experimentally, we chose to perform simulations by using Java programming language, and the IDE Eclipse platform. To evaluate the performance of the access control module, we have chosen to compare it with Ramos et al.'s scheme [32], and Nasiraee and Ashouri-Talouki's scheme [3]. For the performed simulations, the variable parameter used is the number of users trying to access a service of a particular object in the IoT environment. During the simulation, we are interested in the following performance metrics: (1) processing time: represents the execution time required for a user, each time the scheme is run. This metric is important for testing the efficiency of a protocol across communicating users in the network; (2) communication cost: this metric represents the size of the messages transmitted by one communication part during the scheme run. In the Internet of Things, communications between the IoT communication parties are the most expensive operations, especially, with the increase of communicating objects; (3) encryption time: represents the time needed to cipher a data block by using a well-defined encryption algorithm. Security plays a very important role in IoT systems, mostly, when exchanging messages between two communication entities, so the use of encryption algorithms

is extremely important. However, the use of such algorithm influences the encryption time of a scheme. The reason of that is, increasing the size of the exchanged messages between communicating objects directly influences the encryption time; and (4) decryption time: represents the time needed to decipher a data block encrypted with a well-defined algorithm. As already mentioned, a use of an encryption algorithm is important to ensure the security of the transmitted messages. A use of the decryption algorithm completes this function, which influences the decryption time with the increase of the size of the exchanged messages and the number of communicating objects.

In what follows, we provide and discuss the obtained simulation results of the access control module. In order to evaluate this module, we compare its performances with those of Ramos et al.'s scheme [32], and Nasiraee and Ashouri-Talouki's scheme [3].

5.3.1 Processing Time

Fig. 14, depicts the variation of processing time in function of the number of users. As shown in Fig. 14, the processing time of the proposed scheme, that of Ramos et al. [32], as well as that of Nasiraee and Ashouri-Talouki [3], increases with the increase in the number of users. The results relating to the proposed scheme are very satisfactory and better than those presented by Nasiraee and Ashouri-Talouki [3] and Ramos et al.'s scheme [32]. This is explained by the lightweight cryptographic operations used in the access control module, and the transmission load, which is lower, while Ramos et al.'s scheme uses complex algorithms, and particularly Elliptic Curve Digital Signature Algorithm (ECDSA), which is very expensive in CPU resources.

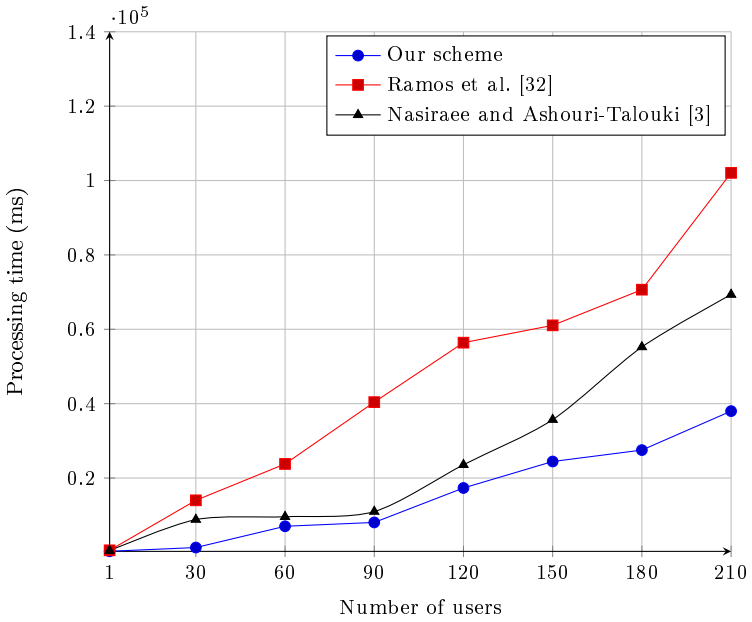

Fig. 14: Overall processing time in function of the number of users.

5.3.2 Communication Cost

Fig. 15, depicts the variation of communication cost in function of the number of users. We note from Fig. 15, that the communication cost is always increasing with increasing the number of users, which is quite obvious. From Fig. 15, it is inferred that the proposed scheme shows better performance compared to that of Nasiraee and Ashouri-Talouki [3] and Ramos et al. [32]. The proposed scheme reduces the communication load between two entities because it avoids adding certificates and signatures during data transmission, thus providing a minimum communication cost.

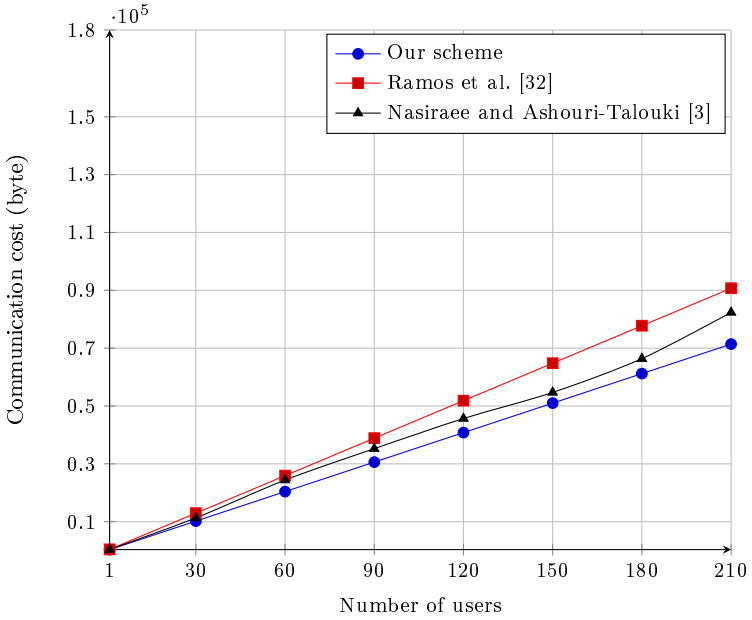

Fig. 15: Overall communication cost in function of the number of users.

5.3.3 Encryption Time

Fig. 16, depicts the variation of encryption time in function of the number of users. As observed in Fig. 16, more the number of users increases more the encryption time increases too. Because of the fact that, the increase in the number of users implies an increase in the amount of the encrypted and signed exchanged messages between the communicating entities. Moreover, the results relating to the proposed scheme are very satisfactory and better than those obtained by Ramos et al.'s scheme [32], and Nasiraee and Ashouri-Talouki [3]. Because in the access control module the user only has to encrypt a few amount of messages by using a lightweight encryption algorithm. However, this amount in Ramos et al.'s scheme is much larger, and it takes more time to be encrypted, and signed by using ECDSA algorithm.

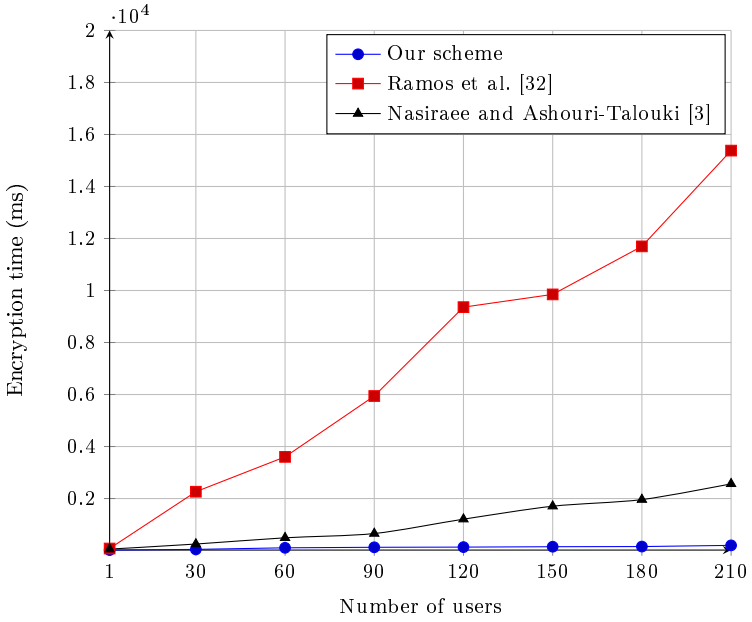

Fig. 16: Overall encryption time in function of the number of users.

5.3.4 Decryption Time

Fig. 17, depicts the variation of decryption time in function of the number of users. As shown in Fig. 17, we notice that the decryption time increases with the increase of the number of users. Likewise, this graph show that the decryption time of the proposed scheme obviously outperforms Ramos et al.'s scheme [32], and Nasiraee and Ashouri-Talouki [3]. This is due to the complex cryptographic algorithm used in the concurrent schemes, which increases both the encryption, and decryption times of the exchanged data in the IoT environment.

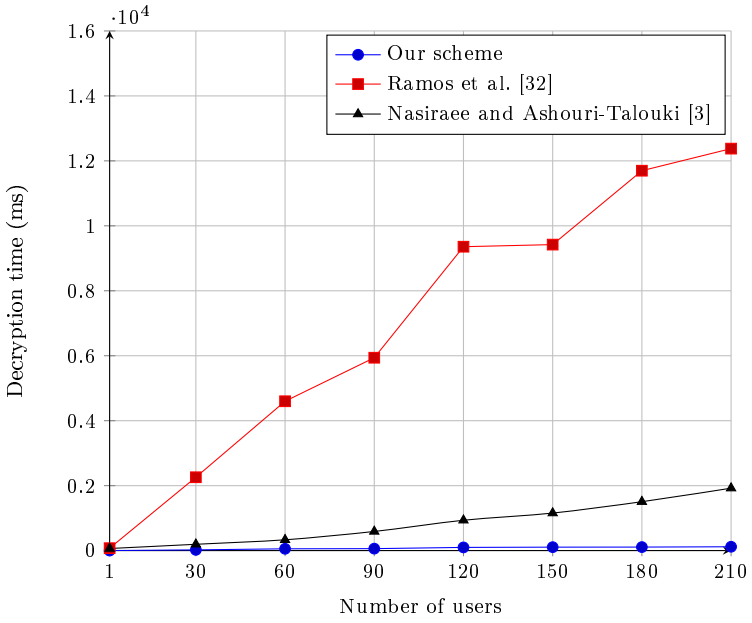

Fig. 17: Overall decryption time in function of the number of users.

6 Conclusion

In this paper, we have proposed a scheme consisting of key management and access control modules to enhance security in IoT environment. The key management module is based on trust graphs for Internet of things. The idea of integrating the cryptography technique into the trust graphs contributes enormously to the security of the data exchange and key management in such networks. This first module allows the IoT objects to have a key pair which will be used for the generation of their symmetric key. Likewise, it allows these IoT objects to generate their secret keys in a distributed manner without resorting to a central authority. Moreover, The proposed scheme minimizes the number of keys, which are generated in the IoT, since each IoT object only shares a single symmetric cryptographic key with the objects it trusts in the network. The access control module is an improvement of a network authentication scheme "Kerberos". This improvement is intended to ensure scalability in this scheme, because each time a user wants to use the service of an IoT object, he must involve the authentication server. The latter suffers from processing power limitation, so that the Kerberos authentication scheme has poor expandability, and this demonstrates that Kerberos standard is not suitable for large-scale networks. The main merit of this improvement is that: (i) it provides distributivity through a trust relationship which connects $\mathsf{TGO_i}$ servers from different domains; (ii) it checks for permissions, access rights, and roles associated with each user based on the RBAC model; (iii) it ensures a speed of execution; (iv) the system can continue to operate even if an authentication server or $\mathsf{TGO_i}$ server break down; and (v) it ensures the sharing and distributivity of data. The security analysis of the proposed scheme confirms that it can withstand various malicious cryptographic attacks. Finally, the proposed scheme has been validated by simulation. The obtained results indicate that the proposed scheme offers a reduced processing, storage and communication costs, encryption and decryption time, as well as provides a minimal energy consumption compared with the concurrent schemes.

In our future work, we plan to implement the proposed scheme under a real platform. We also intend to deepen the performance analysis of the proposed scheme by considering other performance metrics to concretize the results for possible comparisons with other existing schemes. Similarly, to measure the weaknesses of the two modules involved in the proposed scheme.

Acknowledgment

This work was carried out in the framework of the research activities of the LIMED (laboratory of Medical Computing) laboratory, which is affiliated to the faculty of exact sciences of the university of Bejaia and the LIGM laboratory of the University of Gustave Eiffel, France. It has been sponsored by the General Directorate for Scientific Research and Technological Development, Ministry of Higher Education and Scientific Research (DGRSDT), Algeria.

References

1. Esposito, C., Ficco, M., Castiglione, A., Palmieri, F., De Santis, A. Distributed group key management for event notification confidentiality among sensors. IEEE Transactions on Dependable and Secure Computing. **17**(3), 566–580 (2020).
2. Cao, Q., Li, Y., Wu, Z., Miao, Y., Liu, J.: Privacy-preserving conjunctive keyword search on encrypted data with enhanced fine-grained access control. World Wide Web. **23**, 959–989 (2020).
3. Nasiraee, H., Ashouri-Talouki, M.: Anonymous decentralized attribute-based access control for cloud-assisted IoT. Future Generation Computer Systems. **110** 45–56 (2020).
4. Yan, H., Wang, Y., Jia, C., Li, J., Xiang, Y., Pedrycz, W.: IoT-FBAC: Function-based access control scheme using identity-based encryption in IoT. Future Generation Computer Systems. **95** 344–353 (2019).
5. Miloslavskaya, N., Tolstoy, A.: Internet of Things: information security challenges and solutions, Cluster Computing, 1–17 (2018).
6. Conti, F., Palossi, D., Andri, R., Magno, M., Benini, L.: Accelerated visual context classification on a low-power smartwatch. IEEE Transactions on Human-Machine Systems. **47**(1), 19–30 (2017).
7. Hong, S., Kim, H., Chang, J. An efficient key management scheme for user access control in outsourced databases. World Wide Web. **20**, 467–490 (2017).
8. Jisha, C.T., Mamatha, B., Derroll, D.: Survey on Internet of Things (IoT): security issues and countermeasures. International Journal of Engineering Trends and Technology. **46**(5), 273–277 (2017).
9. Borgia, E. Gomes, D.G., Legesse, B., Lead, R., Puccinelli, D.: Special issue on internet of things: Research challenges and solutions. Computer Communications. **89-90**, 1–4 (2016).
10. Haripriya, A.P., Kulothungan, K.: ECC based self-certified key management scheme for mutual authentication in Internet of Things. In: Proceedings of the International Conference on Emerging Technological Trends. 1–6, (2016).
11. Hong, S., Kim, H.I., Chang, J.W.: An efficient key management scheme for user access control in outsourced databases. World Wide Web. **20**(3), 467–490 (2016).
12. Iqbal, A., Bayoumi, M.: Secure end-to-end key establishment protocol for resourceconstrained healthcare sensors in the context of IoT. In: Proceedings of the Internationl Conference on High Performance Computing and Simulation. **85**, 523–530 (2016).
13. Li, S., Tryfonas, T., Li, H.: The internet of things: A security point of view. Internet Research. **26**(2), 337–359 (2016).
14. Patel, S., Patel, D.R., Navik, A.P.: Energy efficient integrated authentication and access control mechanisms for Internet of Things, In: Proceedings of the International Conference on Internet of Things and Applications. 304–309 (2016).
15. Singh, S., Malik, M.: IoT with advance uses and limitations of smart object. International Journal of Engineering Applied Sciences and Technology. **1**(10), 48–50 (2016).
16. Xue, N., Liang, L., Zhang, J., Huang, X.: An access control system for intelligent buildings. In: Proceedings of the 9^{th} EAI International Conference on Mobile Multimedia Communications. 11–17 (2016).
17. Yu, J., Kim, M., Bang, H.C., Bae, S.H., Kim, S.J.: IoT as a applications: cloud-based building management systems for the internet of things. Multimedia Tools and Applications **75**(22), 14583–14596 (2016).
18. Abdmeziem, R., Tandjaoui, D.: A cooperative end to end key management scheme for e-health applications in the context of Internet of Things. Computers and Electrical Engineering. **44**, 184–197 (2015).
19. Bairagi, A.K., Chakroborti, D.: Trust based d2d communications for accessing services in Internet of Things. In: Proceedings of the 18^{th} International Conference on Computer and Information Technology. 50–54 (2015).
20. Nguyen, K.T., Laurent, M., Oualha, N.: Survey on secure communication protocols for the Internet of Things. Ad Hoc Networks. **32**, 17–31 (2015).
21. Porambage, P., Braeken, A., Kumar, P., Gurtov, A., Ylianttila, M.: Efficient key establishment for constrained IoT devices with collaborative HIPbased approach. In: Proceedings of IEEE Global Communications Conference (GLOBECOM). 1–6 (2015).
22. Porambage, P., Braeken, A., Schmitt, C., Gurtov, A., Ylianttila, M., Stiller, B.: Group key establishment for enabling secure multicast communication in wireless sensor networks deployed for IoT applications. IEEE Journals and Magazines. **24**, 1–8 (2015).

23. Rivera, D., Piris, L.C., Civera, G.L., Hoz, E.d.l., Maestre, I.M.: Applying an unified access control for IoT-based intelligent agent systems. In: Proceedings of the 8th International Conference on Service-Oriented Computing and Applications. 247–251 (2015).
24. Sciancalepore, S., Capossele, A., Piro, G., Boggia, G., Bianchi, G.: Key management protocol with implicit certificates for IoT systems. In: Proceedings of the 2015 Workshop on IoT challenges in Mobile and Industrial Systems, ACM. **2**(3), 37–42 (2015).
25. Abdmeziem, R., Tandjaoui, D.: A lightweight key management scheme for e-health applications in the context of Internet of Things. In: Proceedings of the International Conference on Next Generation Computing and Communication Technologies. 47–52 (2014).
26. Billet, B., Issarny, V.: From task graphs to concrete actions: A new task mapping algorithm for the future Internet of Things. In: Proceedings of the 11th International Conference on Mobile Ad Hoc and Sensor Systems. 470–478 (2014).
27. Chen, S., Xu, H., Liu, D., Hu, B., Wang, H.: A vision of IoT: applications, challenges, and opportunities with china perspective. IEEE Internet of Things journal. **1**(4), 349–359 (2014).
28. Palma, D., Agudo, J.E., Sánchez, H., Macías, M.M.: An Internet of Things example: classrooms access control over near field communication. Sensors. **14**(4), 6998–7012 (2014).
29. Shafagh, H., Hithnawi, A.: Poster Abstract: Security comes first, a public-key cryptography framework for the Internet of Things. In: Proceedings of the IEEE International Conference on Distributed Computing in Sensor Systems. 135–136 (2014).
30. Ye, N., Zhu, Y., WANG, R.c., Reza, M., Lin, Q.M.: An efficient authentication and access control scheme for perception layer of Internet of Things. Applied Mathematics & Information Sciences. **8**(4), 1617–1624 (2014).
31. Pereira, C.S., Augusto, A.B., Correia, R.C., Correia, M.E.: A secure RBAC mobile agent access control model for healthcare institutions. In: Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems. 349–354 (2013).
32. Ramos, J.L.H., Jara, A.J., Marín, L., Skarmeta, A.F.: Distributed capability-based access control for the Internet of Things. Journal of Internet Services and Information Security. **3**(3/4), 1–16 (2013).
33. Salunke, D., Upadhyay, A., Sarwade, A, Marde, V., Kandekar, S.: A survey paper on role based access control. International Journal of Advanced Research in Computer and Communication Engineering. **2**(3), 1340–1342 (2013).
34. Seitz, L., Selander, G., Gehrmann, C.: Authorization framework for the Internet of Things. In: Proceedings of the 14th International Symposium on a World of Wireless, Mobile and Multimedia Networks. 1–6 (2013).
35. Veltri, L., Cirani, S., Busanelli, S., Ferrari, G.: A novel batch-based group key management protocol applied to the Internet of Things. Ad Hoc Networks. 2724–2737 (2013).
36. Anggorojati, B., Mahalle, P.N., Prasad, N.R., Prasad, R.: Capability-based access control delegation model on the federated IoT network. In: Proceedings of the 15th International Symposium on Wireless Personal Multimedia Communications. 604–608 (2012).
37. Challal, Y.: Sécurité de l'Internet des Objets : vers une approche cognitive et systémique. Réseaux et télécommunications [cs.NI]. Université de Technologie de Compiègne, <tel-00866052>, (2012).
38. Chen, D., Chang, G., Sun, D., Jia, J., Wang, X.: Lightweight key management scheme to enhance the security of Internet of Things. Wireless and Mobile Computing. **5**(2), 191–198 (2012).
39. Yu, H., He, J., Zhang, T., Xiao, P.: A group key distribution scheme for wireless sensor networks in the Internet of Things scenario. International Journal of Distributed Sensor Networks. Article ID 813594. 1–12 (2012).
40. Islam, S.H., Biswas, G.P.: Design of improved password authentication and update scheme based on elliptic curve cryptography, Mathematical and Computer Modelling, 57(11-12), 2703–2717 (2011).
41. Li, M., Sun, X., Wang, H., Zhang, Y., Zhang, J. Privacy-aware access control with trust management in web service. World Wide Web. **14**, 407–430 (2011).
42. Bin, S., Yuan, L., Xiaoyi, W.: Research on data mining models for the internet of things. In: Proceedings of the International Conference on Image Analysis and Signal Processing. 127–132 (2010).
43. Zhang, G., Tian, J.: An extended role based access control model for the Internet of Things, In: Proceedings of the International Conference on Information. Networking and Automation. **1**, 319–323 (2010).
44. Tseng, Y.M., Yang, C.C., Liao, D.R.: A Secure group communication protocol for Ad Hoc wireless networks. In: Advances in Wireless Ad Hoc and Sensor Networks. Signals and Communication Technology. Springer. Boston. MA. 103–131 (2008).

45. Ren, K., Li, T., Wan, Z., Bao, F., Deng, R.H., Kim, K.: Highly reliable trust establishment scheme in ad hoc networks. Computer Networks. **45**(6), 687–699 (2004).
46. Capkun, S., Buttyán, L., Hubaux, J.P.: Small worlds in security systems: an analysis of the PGP certificate graph. In: Proceedings of the 2002 workshop on New security paradigms. 28–35 (2002).
47. Heinzelman, W., Chandrakasan, A., Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In: Proceedings of the 33^{rd} Annual Hawaii International Conference on System Sciences. 3005–3014 (2000).
48. Ferraiolo, D., Cugini, J., Kuhn, R.: Role-based access control (RBAC): Features and motivations. In: Proceedings of the 11^{th} Annual Computer Security Application Conference. 241–248 (1995).
49. Koblitz, N.: Elliptic curve cryptosystems. Mathematics of computation. **48**, 203–209 (1987).
50. Miller, V.: Uses of elliptic curves in cryptography. In Advances in Cryptology, Crypto 85, Springer Verlag LNCS 218. 417–426 (1986).
51. Diffie, W., Hellman, M. : New directions in cryptography. IEEE Transactions in Information Theory. **22**(6), 644–654 (1976).