Wireless Communications and Mobile Computing

Security, Privacy, and Trust on Internet of Things

Lead Guest Editor: Constantinos Kolias
Guest Editors: Weizhi Meng, Georgios Kambourakis, and Jiageng Chen

Security, Privacy, and Trust on Internet of Things

Wireless Communications and Mobile Computing

Security, Privacy, and Trust on Internet of Things

Lead Guest Editor: Constantinos Kolias
Guest Editors: Weizhi Meng, Georgios Kambourakis, and Jiageng Chen

Editorial Board

Contents

Hindawi
Wireless Communications and Mobile Computing
Volume 2019, Article ID 6452157, 3 pages
https://doi.org/10.1155/2019/6452157

Editorial

Security, Privacy, and Trust on Internet of Things

Constantinos Kolias,[1] **Weizhi Meng**,[2] **Georgios Kambourakis**,[3] **and Jiageng Chen**[4]

[1]*Computer Science Department, University of Idaho, USA*
[2]*Department of Applied Mathematics and Computer Science, Technical University of Denmark, Denmark*
[3]*Department of Information and Communication Systems Engineering, University of the Aegean, Greece*
[4]*School of Computer Science, Central China Normal University, China*

Correspondence should be addressed to Constantinos Kolias; kolias@uidaho.edu

Received 25 December 2018; Accepted 31 December 2018; Published 3 February 2019

1. Introduction

The ability of smart objects to stay connected to the Internet for purposes of transmitting and receiving data is referred to as the Internet of Things (IoT). As per recent estimates, the number of IoT devices will surpass 50 billion by 2020. Unsurprisingly, this mushrooming of IoT devices has drawn the attention of attackers who seek to exploit them for their own benefit, with the Mirai botnet being perhaps the most prominent example of IoT specific malware [1, 2]. Basically, IoT brings along a plethora of potential security and privacy risks to the end-users, including the unsanctioned access and abuse of private information, the enabling and strengthening of assaults against other systems, and the breeding of risks pertaining to personal safeness [3]. Especially, IoT facilitates the creation of an assortment of privacy risks to the consumer associated with the collection of personal and sensitive information, like their preferences, locations, habits, and so on. In the mid- or long-run these pieces of data can be used to, say, profile or impersonate the user or group of interest. On the other hand, such risks to security, privacy, and trust may significantly diminish end-user's confidence in IoT and therefore impede its full realization.

The feature topic at hand intends to promote the dissemination of the latest methodologies, solutions, and case studies pertaining to IoT security, privacy, and trust issues. Its objective is to publish high-quality articles presenting security algorithms, protocols, policies, frameworks, and solutions for the IoT ecosystem.

The goal of this special issue was to attract high-quality contributions from researchers working in the broad area of security, privacy, and trust for IoT ecosystems, including but not limited to (a) cloud computing-based security solutions for IoT data, (b) mobile service privacy for IoT devices, (c) standardization efforts related to IoT, (d) testbeds and case studies for IoT, (e) Intrusion detection for IoT, (f) trust management for IoT, and (g) virtualization solutions to IoT security

2. Submissions

This special issue presents high-quality articles describing security and privacy issues, attacks as well as their remedies for the IoT ecosystems. We received a total of 29 submissions and, after a rigorous review process, we selected 10 articles covering the subject from different perspectives, i.e., about 30% of all the submitted papers.

In "On the RCCA Security of Hybrid Signcryption for Internet of Things" by H. Dai et al., hybrid signcryption schemes are lucrative for protecting communications in IoT environments. Such schemes achieve multiple cryptographic services simultaneously but with much lower overhead than separate traditional cryptographic schemes. This attribute makes them ideal for resource-constrained environments. Unlike most approaches that verify such security schemes primarily against Chosen Ciphertext Attacks, this paper proposes verification against Repayable Chosen Ciphertext Attacks. Despite being a theoretically weaker security notion, it is "secure enough" for IoT applications and at the same time much more efficient.

In "A Hierarchical Matrix Decomposition-Based Signcryption without Key-Recovery in Large-Scale WSN" by C.

Yuan et al., identity-based encryption schemes present a great potential for wireless, low resources networks due to their lower resource requirements. However, such schemes assume that a central entity, namely, the Private Key Generator (PKG), maintains all private keys; therefore, it can easily impersonate any user. The paper proposes a novel signcryption technique based on hierarchical matrix decomposition to generate the keys for cluster head nodes. By limiting the control of central authorities on the private keys it becomes possible to solve the key escrow issue associated with such schemes.

In "A Blockchain-Based Contractual Routing Protocol for the Internet of Things Using Smart Contracts" by G. Ramezan and C. Leung, conventional secure routing protocols assume a central authority (CA) to facilitate the identification and authentication for each device in the network. Particularly, in the highly heterogeneous IoT environments the lack of a standardized central management system introduces the problem of trust. The paper proposes a blockchain based contractual routing protocol which operates in a fully distributed manner without requiring any trusted CA. The introduced protocol makes use of the smart contracts concept to discover a route to a destination or data gateway within heterogeneous IoT networks. The protocol is proven resistant to both Blackhole and Greyhole attacks.

While in "Shielding IoT against cyber-attacks: An event-based approach using SIEM" by D. D. Lopez et al., due to the high level of heterogeneity in IoT environments traditional security solutions cannot perform ideally. Security Information and Event Management systems seem to be an appealing solution; however, current practices known from conventional computer networks fail to take into account the possible correlations between IoT layers and the peculiarities of corresponding security events and attack surfaces. The paper proposes a custom-tailored security architecture and explores possible mappings between events, vulnerabilities, and attack surfaces for typical IoT ecosystems.

In "BaDS: Blockchain-Based Architecture for Data Sharing with ABS and CP-ABE in IoT" by Y. Zhang et al., cloud infrastructures are an indispensable component of IoT applications, yet they may not always be considered as fully trusted entities. This paper proposes a privacy-preserving and user-controlled data sharing architecture which permits detailed access control. The proposed approach is based on the Blockchain model and smart contracts to ensure the scalability of access control tables.

In "Towards Secure Network Computing Services for Lightweight Clients using Blockchain" by Y. Xu et al., the network-based service sharing paradigm may indirectly extend the abilities of the resource-constrained IoT devices; nevertheless it introduces additional risks since untrusted/unverified code can be loaded from the network and then be executed even natively. This paper proposes a novel blockchain-based secure service provisioning mechanism for protecting lightweight IoT devices from malicious or insecure services in network computing scenarios. The power of blockchain is primarily leveraged towards identifying and verifying the corresponding provider and service.

In "Security Vulnerabilities and Countermeasures for Time Synchronization in TSCH Networks" by W. Yang et al., numerous IoT applications require that all nodes must maintain high-precision time synchronization. Such communication systems suffer from time-synchronization attacks, primarily in single-hop pair-wise synchronization situations. The paper examines the security vulnerabilities of TSCH technology to identify the potential vulnerabilities and attacks. The corresponding security enhancements are also outlined and an authentication-based mechanism along with a clock-offset filter is proposed.

In "Towards Smart Healthcare: Patient Data Privacy and Security in Sensor-Cloud Infrastructure" by I. Masood et al., Modern Wireless Body Area Networks (WBANs) systems extensively rely on cloud computing (CC) technologies to overcome their inherent computational constraints. Such hybrid infrastructures have been applied in the healthcare domain with great success, but at the same time, new threats against patient data privacy and security were surfaced. This paper surveys the techniques for patient data privacy and security in sensor-based cloud infrastructures. The paper also provides a framework for patient physiological parameters (PPPs) privacy and security particularly appropriate for such ecosystems.

In "Towards Privacy Preserving IoT Environments: A Survey" by M. Seliem et al., privacy is one pivotal requirement of IoT applications. One of the most essential concerns of IoT applications is the lack of control over raw personal data communicated from the sensors to the cloud application counterparts. This paper conducts a thorough survey of existing research and proposed solutions regarding privacy in IoT ecosystems, from a multipoint of view to outline the numerous associated risks and potential mitigations.

In "FAPRP: A Machine Learning Approach to Flooding Attacks Prevention Routing Protocol in Mobile Ad Hoc Networks" by N. T. Luong et al., IoT communications may sometimes be deprived of a centralized infrastructure thus completely relying on number of self-organizing nodes to form Mobile Ad hoc Networks (MANETs). Such types of networks are prone to request route flooding attack, a devastating attack which is trivial to initiate and challenging to remedy. The authors introduce the Flooding Attack Detection Algorithm (FADA) which is based on historical network traces and the k-NN algorithm to detect and isolate the malicious nodes in the network. Then a new routing protocol for such settings is introduced which incorporates FADA algorithm as part of its route request phase, minimizing the risk.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this special issue.

Acknowledgments

The guest editors would like to express their gratitude to SPTT editorial board for giving the opportunity to edit this special issue. Also, they wish to thank the authors for

submitting their work as well as the tireless reviewers who have constructively evaluated the papers within the short-stipulated time. Finally, they sincerely hope the reader will share their view and find this special issue very useful.

Constantinos Kolias
Weizhi Meng
Georgios Kambourakis
Jiageng Chen

References

[1] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: mirai and other botnets," *IEEE Computer Society*, vol. 50, no. 7, pp. 80–84, 2017.

[2] M. Antonakakis, T. April, M. Bailey et al., "Understanding the mirai botnet," in *Proceedings of the USENIX Security Symposium*, pp. 1092–1110, August, 2017.

[3] J. Voas, R. Kuhn, C. Kolias, A. Stavrou, and G. Kambourakis, "Cybertrust in the IoT Age," *The Computer Journal*, vol. 51, no. 7, pp. 12–15, 2018.

Hindawi
Wireless Communications and Mobile Computing
Volume 2019, Article ID 6869307, 17 pages
https://doi.org/10.1155/2019/6869307

Research Article

FAPRP: A Machine Learning Approach to Flooding Attacks Prevention Routing Protocol in Mobile Ad Hoc Networks

Ngoc T. Luong[1,2] **Tu T. Vo**[1] **and Doan Hoang**[3]

[1]*Faculty of Information Technology, Hue University of Sciences, Hue University, Hue 530000, Vietnam*
[2]*Faculty of Mathematics and Informatics Teacher Education, Dong Thap University, Dong Thap 870000, Vietnam*
[3]*Faculty of Engineering and Information Technology, the University of Technology Sydney, Sydney 2007, Australia*

Correspondence should be addressed to Doan Hoang; doan.hoang@uts.edu.au

Received 6 July 2018; Revised 9 November 2018; Accepted 29 November 2018; Published 10 January 2019

Guest Editor: Jiageng Chen

Request route flooding attack is one of the main challenges in the security of Mobile Ad Hoc Networks (MANETs) as it is easy to initiate and difficult to prevent. A malicious node can launch an attack simply by sending an excessively high number of route request (RREQ) packets or useless data packets to nonexistent destinations. As a result, the network is rendered useless as all its resources are used up to serve this storm of RREQ packets and hence unable to perform its normal routing duty. Most existing research efforts on detecting such a flooding attack use the number of RREQs originated by a node per unit time as the threshold to classify an attacker. These algorithms work to some extent; however, they suffer high misdetection rate and reduce network performance. This paper proposes a new flooding attacks detection algorithm (FADA) for MANETs based on a machine learning approach. The algorithm relies on the route discovery history information of each node to capture similar characteristics and behaviors of nodes belonging to the same class to decide if a node is malicious. The paper also proposes a new flooding attacks prevention routing protocol (FAPRP) by extending the original AODV protocol and integrating FADA algorithm. The performance of the proposed solution is evaluated in terms of successful attack detection ratio, packet delivery ratio, and routing load both in normal and under RREQ attack scenarios using NS2 simulation. The simulation results show that the proposed FAPRP can detect over 99% of RREQ flooding attacks for all scenarios using route discovery frequency vector of sizes larger than 35 and performs better in terms of packet delivery ratio and routing load compared to existing solutions for RREQ flooding attacks.

1. Introduction

A Mobile Ad Hoc Network (MANET) [1] is a collection of wireless mobile devices (called nodes) that dynamically form an ad hoc network in situations such as disaster rescue, urgent conference, or military mission, without the support of a network infrastructure. The topology of the network may change frequently because nodes can join or leave the network at will. In a MANET, nodes coordinate among themselves to maintain the connections among them. Data transfer from a source node to a non-neighbor destination node is routed through intermediate nodes. A node can act as a host and a router at the same time. A network routing protocol in a MANET specifies how nodes in the network communicate with each other. It enables the nodes to discover and maintain the routes between any two of them. Many routing protocols have been developed for MANETs such as ad hoc on-demand distance vector (AODV) [2], dynamic destination sequenced distance vector (DSDV) [3], and zone routing protocol (ZRP) [4]. They are classified into three groups: proactive, reactive, and hybrid routing protocols. With proactive routing protocols, the routes between nodes need to be established before data packets can be sent. These protocols are suitable for fixed topology networks. In contrary, reactive routing protocols are suitable for dynamic topology networks as nodes only try to discover routes on demand. In complex network topologies, hybrid routing protocols are often used [5]. MANETs are thus essential in infrastructureless situations for communication; however, they suffer from various types of Denial of Service (DoS) attacks that deny user services or resources he/she would normally expect to receive. Disrupting routing services at the

network layer is an example of DoS [6, 7] where a malicious node (MN) tries to deplete resources of other nodes. Other types of DoS include Blackhole [8], Sinkhole [9], Grayhole [10], Whirlwind [11], Wormhole [12], and flooding attacks [13]. Flooding attack is a particular form of DoS attacks in MANETs where malicious nodes mimic legitimate nodes in all aspects except that they do route discoveries much more frequently with the purpose of exhausting the processing resources of other nodes. This type of attacks is simple to perform with on-demand routing protocols, typically as AODV [14]. Among HELLO, RREQ, and DATA flooding attacks, route request (RREQ) flooding attack is the most hazardous because it is easy to create a storm of request route packets and cause widespread damage. This paper focuses on the request route flooding attack.

Previous researches on RREQ flooding attacks mainly focus on detection algorithms that rely on the sending frequency of RREQ packets [13, 15–20]. Every node uses a fixed (or dynamic) threshold value to detect an attack. The threshold is calculated based on the number of RREQs originated by node per unit time. A node labels a neighbor node malicious if it receives more RREQs than the allowed threshold from its neighbors. These algorithms, however, have many weaknesses in dealing with the dynamics of MANETs. These include the following: (1) An algorithm with a fixed threshold is not flexible and is not able to cope with dynamic environments where optimal threshold values vary. (2) Even with dynamic threshold algorithms, where the threshold takes into account other factors such as network traffic, mobility speed, and frequency of malicious node attacks, misclassifications rates are still high. In high mobility environments, the connection state of network nodes changes very frequently; a node may not be able to capture accurate and adequate information to distill it to a single threshold. (3) A normal node may be mistaken for a malicious node even if it legitimately sends out a high number of route requests in response to a high priority event. Or (4) a malicious node may avoid the threshold detection mechanism simply by sending RREQ packets at a frequency just lower than the threshold value.

In this paper, we propose and investigate a different approach for detecting flooding attacks. Our solution relies on the route discovery history information of each node to classify a node as malicious or normal. The route discovery history of each node is represented by a route discovery frequency vector (RDFV). The route discovery histories reveal similar characteristics and behaviors of nodes belonging to the same class. This feature is exploited to differentiate abnormal behavior from a normal one. RDFV is defined as the feature vector for detecting malicious nodes in MANET environment. We propose a flooding attack detection algorithm to detect malicious node based on RDFV. We propose a novel flooding attacks prevention routing protocol by incorporating the FADA algorithm and extending the AODV protocol. We evaluate the performance of our solution in terms of successful detection ratio, packet delivery ratio, and routing load both in normal and under RREQ attack scenarios using NS2 simulation. The simulation results showed that our approach can detect over 99% of RREQ flooding attacks, had better packet delivery ratio and routing load compared to existing solutions for RREQ flooding attacks, and introduced negligible overhead relative to AODV for normal scenarios. The main contributions of the paper are as follows:

(1) It introduced a new route discovery history measure, the vector of route discovery frequency, to capture the behavior of MANET nodes.

(2) It proposed a flooding attack detection algorithm, a k-nearest neighbors-based machine learning algorithm, using RDFV dataset to detect malicious nodes.

(3) It proposed a flooding attack prevention routing protocol by integrating FADA into the original AODV protocol.

(4) It evaluated the effectiveness and the performance of the proposed solution for high-speed mobility MANETs under RREQ flooding attacks.

The remainder of this paper is structured as follows: Section 2 presents a review of the related work on detection of flooding attacks. Section 3 presents our solution and a novel flooding attacks prevention routing protocol by improving AODV protocol using FADA. Section 4 presents the results of evaluating the performance of the proposed solution relative to existing solutions. Section 5 concludes the paper.

2. Related Works

2.1. Overview of AODV. AODV is a popular reactive routing protocol in which a node only initiates the process for finding a path to the destination if it wants to send data. Basically, when the source node (N_S) wants to communicate with the destination node (N_D), without an already discovered route to the destination, N_S starts a route discovery process by broadcasting a route request (RREQ) packet containing the destination address. The nodes that receive the packet will in turn broadcast it. When N_D receives the packet, it will send a route reply (RREP) packet back to source node. Once a route has been discovered, HELLO and RERR packets can be used to maintain the status of the route.

Figure 1 describes the route discovery process of AODV; source node (N_7) discovers route to destination node (N_{11}) by broadcasting an RREQ to its neighbor nodes. When a node receives the RREQ packet for the first time, it broadcasts the packet and sets up a reverse path to the source. If the node receives the same RREQ subsequently, it simply drops the packet. When N_{11} gets a RREQ, it unicasts a RREP packet to the source node through the established reverse $\{N_{11} \longrightarrow N_{10} \longrightarrow N_9 \longrightarrow N_7\}$. When N_7 gets a RREP, it establishes successfully a new path to N_{11} with 3 hops routing cost and adds the new entry to its routing table.

2.2. Flooding Attacks on AODV. Flooding attack is a form of DoS attacks in which malicious nodes broadcast false packets in the network to exhaust the resources and disrupt the network operation. Depending on the type of packet used to flood the network, flooding attack can be categorized into three categories, RREQ, DATA, and HELLO flooding attack.

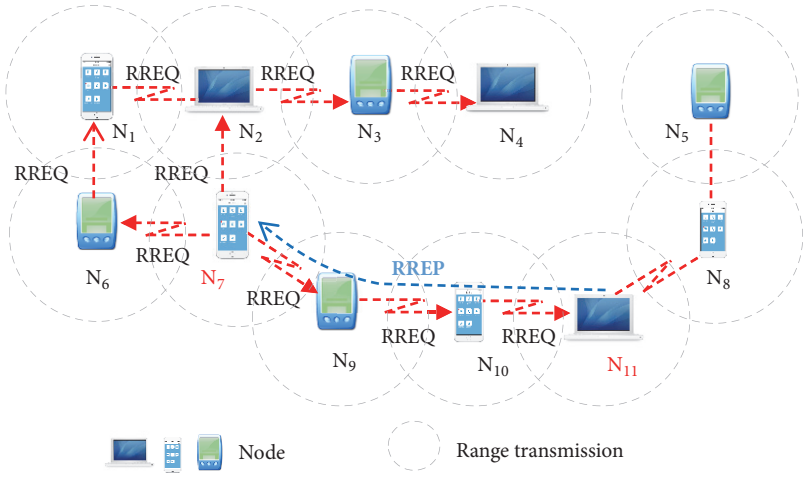


Figure 1: Description of route discovery process of AODV in the MANET.

In RREQ flooding attack, a malicious node continuously and excessively broadcasts fake RREQ packets, which causes a broadcast storm and floods. The RREQ flooding attack is considered most harmful in MANET because it can ruin the route discovery process by exhausting the channel bandwidths and the processing resources of affected nodes. In DATA flooding attack, a malicious node can excessively broadcast data packets to any nodes in the network. This type of attacks has more impact on the nodes participating in the data routing to the destinations. In HELLO flooding attack, nodes periodically broadcast HELLO packets to announce their existence to their neighbors. A malicious node abuses this feature to broadcast HELLO packets excessively and forces its neighbors to spend their resources on processing unnecessary packets. This type is only detrimental to the neighbors of a malicious node. Figure 2 shows the behavior of malicious nodes (M) in a MANET for these types of attacks.

2.3. Review on Related Research. This section summarizes related work on threshold-based, machine learning-based, hash function-based, and digital-signature-based approaches in detecting and preventing flooding attacks in MANETs. Table 1 summarizes these methods and their drawbacks.

2.3.1. On Fixed Threshold-Based Approach. Solutions are simple with a fixed threshold for mitigating the impact of RREQ flooding attacks. However, with a static threshold, these methods are not suitable for dynamic environments where nodes are highly mobile and frequently broadcast route request packets. In [15], Gada used three fixed thresholds: RREQ_ACCEPT_LIMIT, RREQ_BLACKLIST_LIMIT, and RATE_RATELIMIT. The default value of RATE-RATELIMIT is 10. If the rate of receiving request packets is greater than RREQ_ACCEPT_LIMIT but less than RREQ_BLACKLIST_LIMIT, packets are simply dropped and not processed. If it is greater than RREQ_BLACKLIST_LIMIT, the source is declared as a malicious node. The weakness of this solution is that it may lead to blacklisting of normal nodes false positive [16] and cause excessive end-to-end delay by dropping legitimate request packets once the RREQ_ACCEPT_LIMIT threshold is crossed.

In [16], Song et al. proposed a simple technique using an Effective Filtering Scheme (EFS) to detect malicious nodes. This solution uses two limit values: RATE_LIMIT and BLACKLIST_LIMIT. If the detected RREQ rate is higher than the RATE_LIMIT and the BLACKLIST_LIMIT, the malicious node is declared and it will be put into the black list. If the rate of RREQs originated by a node is between the RATE_LIMIT and the BLACKLIST_LIMIT, the RREQ packet is added to a "delay queue" waiting to be processed. Here the authors set the RATE_LIMIT threshold to 5 and set the BLACKLIST_LIMIT up to 10.

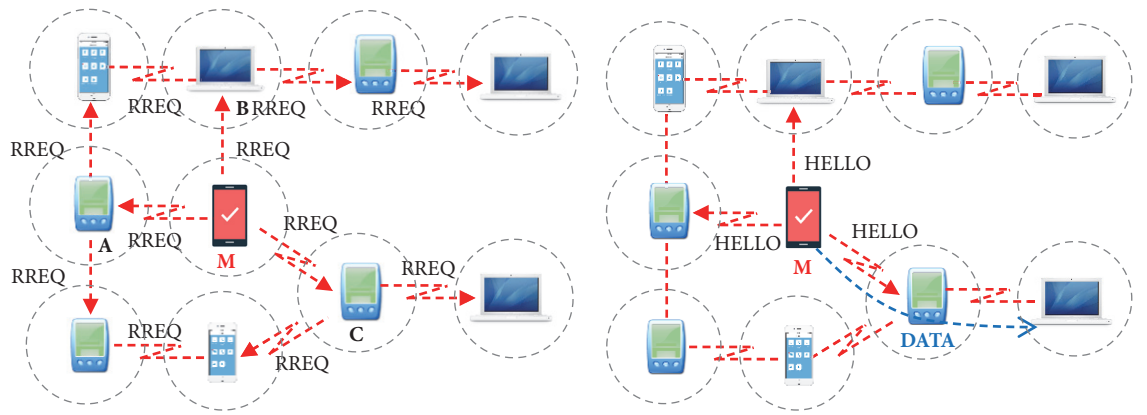
In [13, 17], the authors developed flooding attack prevention (FAP) that prevents RREQ and DATA flooding attacks in MANETs. They argued that the priority of a node is adversely proportional to its broadcast frequency of RREQ. Hence, nodes that generate a high frequency of route requests will have a low priority and may be removed out of the routing process. It is suggested that a node should not originate more than 10 RREQ packets per second and, hence, the threshold of FAP is set at 15 for a good margin.

2.3.2. On Dynamic Threshold-Based Approach. Solutions with dynamic thresholds are more flexible as they can cope with the dynamic environment of MANETs. In [18], Mohammad proposed an improved protocol called B-AODV. In this method, each node employs a balance index (BI) for acceptance or rejection of RREQ packets. If the RREQ rate is higher than the BI value, a malicious node is defined and the RREQ packet is dropped. The results showed that B-AODV is resilience against RREQ flooding attacks. The main drawback of B-AODV is that it may drop legitimate request packets of the node moving at high speed as the number of request packets may be higher than the balance index value [19]. Also, the method does not have a confirmation mechanism which can identify the node properly as a malicious node.

In [19], Gurung proposed a new mechanism called Mitigating Flooding Attack Mechanism. The mechanism is

Table 1: Summary of drawbacks of related works for detecting flooding attacks.

Ref	Name	Year	Method	Drawback
[15]	Proposed-AODV	2004	Fixed threshold	It uses static threshold value which is not suitable for high mobility environment. Malicious node can pass the security mechanism by transmitting RREQ packets at a frequency lower than the threshold.
[13]	FAP	2005		
[16]	EFS	2006		
[18]	B-AODV	2016	Dynamic threshold	It can drop valid request packets of the node moving with high mobility speed if the number of request packets is greater than BI value. Malicious node can pass the security mechanism by transmitting RREQ packets at a frequency lower than the threshold.
[19]	F-IDS	2017	Dynamic threshold	Performance varies. Using new control packets (ALERT) will increase communication overhead and limit the performance when operating in network environment without attacks. Malicious node can pass the security mechanism by transmitting RREQ packets at a frequency lower than the threshold.
[20]	SMA_2AODV	2017	Dynamic threshold	Malicious node can pass the security mechanism by transmitting the RREQ packets at a frequency lower than the threshold.
[21]	SVMT	2013	SVM	The proposed algorithm uses fixed threshold to detect malicious nodes.
[22]	kNN-AODV	2014	kNN	The algorithm for building training data sets was not presented or justified.



(a) RREQ flooding (b) HELLO and DATA flooding

Figure 2: Description of flooding attacks in the MANET.

based on a dynamic threshold and consists of three phases. It deploys special Flooding Intrusion Detection System (F-IDS) nodes to detect and prevent flooding attack. The F-IDS nodes are set in the promiscuous mode to monitor the behavior of nodes in the network. The proposed mechanism has several features: (1) it uses a dynamic threshold; (2) it has a confirmation mechanism in which the special F-IDS node confirms the node as a malicious node by sending a dummy reply packet and waits for the data packets; and (3) it has a recovery mechanism that allows the node to participate in the network after the expiry of the blocking time period. However, the use of several F-IDS nodes to monitor their neighbors and to communicate among them limits the performance of the overall network, especially when the network is not under attack.

In [20], Tu introduced security mobile agents (SMA) to detect flooding attacks. An improved protocol, SMA_2AODV, is proposed by integrating these SMAs into the discovery route process of the AODV protocol. During the training period, SMA agents are used to collect information for determining the minimal time-slot (the minimum time-slot for successfully discovering a path from a source node to a destination node) of the system (TS_{min}). After the training phase, node N_i checks the security of the RREQ packet received from source node N_j before broadcasting it to the neighbors. If route discovery time-slot is smaller than the minimal time-slot of the system ($T < TS_{min}$), a flooding attack is said to have occurred with N_j as the attacker. N_i then adds N_j to its black list. All RREQ packets of nodes in the black list will be dropped. The drawback of this method is

Table 2: Description of symbols.

Variable	Description
t_i	Route discovery time i^{th}
T_i	Inter-route discovery time i^{th}
V_{Ns}	Vector of route discovery frequency of N_S node
m	Size of vector of route discovery frequency
k	Cutoff value for kNN algorithm

that TS_{min} is only valid if no malicious node exists during the training period.

2.3.3. On Machine Learning Approach. In [21], Patel proposed the use of support vector machine (SVM) algorithm for detecting and preventing flooding attacks. The behavior of every node is collected and passes to the support vector machine to decide if a node is malicious based on a threshold limit.

In [22], Wenchao proposed a new intrusion detection system based on k-nearest neighbors (kNN) classification algorithm in wireless sensor network to separate abnormal nodes from normal nodes by observing their behaviors. An m-dimensional vector is used to represent nodes and their behaviors such as the number of routing messages that can be sent over a period of time, the number of nodes with different destinations in the sending routing packets, and the number of nodes with the same source node in the receiving routing packets. The paper shows that the system achieves high detection accuracy, but it does not provide justifications or the algorithm for building training datasets.

3. The Proposed FAPRP Solution

This section we present our algorithms and routing protocol for detecting flooding attacks in MANETs. First, we define a feature vector that represents the behavior of a node based on its history of route discovery: the route discovery frequency vector. Second, we describe an algorithm for obtaining the training dataset which describes the normal behavior and the abnormal behavior of nodes for normal/malicious classification. Third, we present our flooding attack detection algorithm, and finally we present our proposed AODV-based flooding attacks prevention routing protocol. Table 2 defines symbols used in the paper.

3.1. Route Discovery Frequency Vector. In order to detect RREQ flooding attacks with kNN, the crucial problem is the selection of a feature vector that maximizes the separation of the normal and the malicious data classes and produces highly reliable classification. The selected features should be able to succinctly capture the inherent behavior of a node performing RREQ requests and the time-related network activities through their historical data records in order to differentiate "normal" from "malicious" behavior. We propose a route discovery frequency vector as the feature vector for this purpose. To quantify this vector, we define the following terms.

Definition 1. Route discovery time (t_i) is the duration from the time a node first broadcasts a route discovery packet to the time it receives the corresponding route response. Assuming that node N_i receives the i^{th} RREQ packet from the source node *Ns* at time s_i and N_i receives the route response packet at time e_i, the route discovery time (t_i) is defined by

$$t_i = e_i - s_i. \tag{1}$$

Definition 2. Inter-route discovery time (T_i) is the duration from the end of a route discovery to the beginning of the next route discovery. Assuming that the node N_i receives the $i+1^{th}$ RREQ packet from the source node *Ns* at time s_{i+1}, the inter-route discovery time (T_i) is defined by (2).

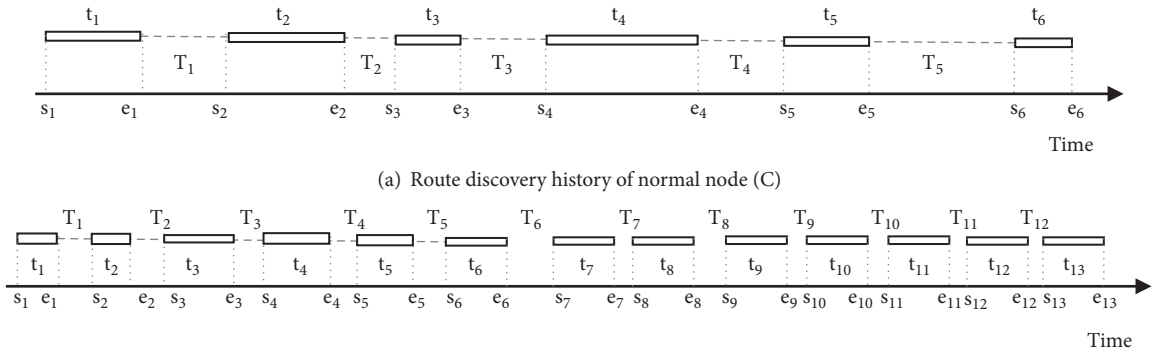
$$T_i = s_{i+1} - e_i \tag{2}$$

In AODV routing protocol, route discovery frequency of a node depends on how frequent the node has to find a path to the required destination. All normal nodes have route discovery frequencies within a range, but malicious nodes have higher route discovery frequencies as their aim is to flood the network. Consider Figure 2(a); it shows three normal nodes, A, B, C, and one malicious node, M. Figure 3(a) shows the route discovery history of the normal node (C) as recorded by the normal node (A). Figure 3(b) shows route discovery history of the malicious node (M) that is also recorded by the normal node (A). The figures show that node C sent 6 RREQ packets and node M sent 13 RREQ packets over roughly the same duration.

We use a m-dimensional vector V_{Ni} (a_1, a_2, a_3, …, a_m) to represent route discovery history of node N_i, where m is the size of the vector and a_i is the i^{th} inter-route discovery time.

Example 1. Route discovery history of the malicious node shown in Figure 3(b) is represented by the route discovery frequency vector V_M ($T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8, T_9, T_{10}, T_{11}, T_{12}$) of size 12.

Figure 4 shows typical vectors of size 40 of the route discovery frequency of normal and malicious nodes, by NS2 simulation. It can be seen that the inter-route discovery time values for all normal nodes (N_1 to N_5) are generally larger (> 1 sec) than those for malicious nodes (M_1 to M_5) as they have low route discovery frequencies. However, there are cases where the malicious inter-route discovery times (T_i) are indistinguishable from the normal ones. One reason for this is the mobility of nodes in the environment; a recording node may not receive RREQ packets from a malicious node until some later time. Other reason for the overlapping



(a) Route discovery history of normal node (C)

(b) Route discovery history of malicious node (M)

Figure 3: Route discovery history recorded at normal node (A).

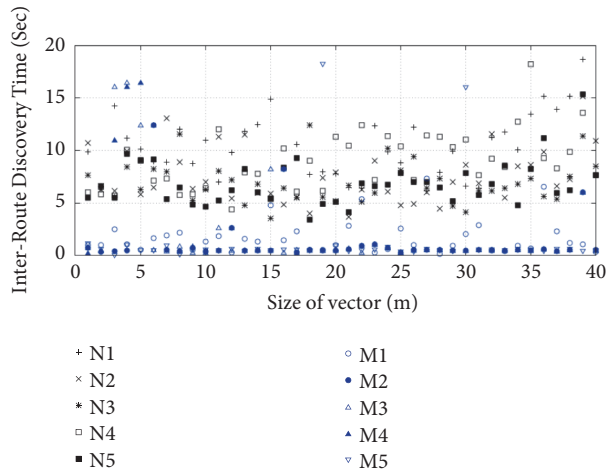


Figure 4: An example describes vectors of route discovery frequency of 5 normal nodes (N_1 to N_5) and 5 malicious nodes (M_1 to M_5).

region is when a malicious node floods the network at a frequency close to the rate at which a normal node can generate RREQs. As demonstrated in Section 4, our proposed algorithm successfully recognizes these abnormal cases based on route discovery frequency feature.

3.2. Algorithm for Obtaining a Training Dataset. We use NS2 [23] version 2.35 to build a training dataset of NVC (normal) and MVC (malicious) vector classes. The simulation scenario is set up with 100 normal nodes and 1 malicious node, operated in the area of 2000m x 2000m. Normal nodes move under random waypoint model with maximum speeds 0m/s, 10m/s, 20m/s, 30m/s, and 40m/s scenarios; a malicious node is positioned at the center (1000m x 1000m) as shown in Figure 5. Other simulation parameters include AODV routing protocol, 50 UDP connections, and constant bit rate (CBR) traffic type; the first data source commences at time 0, other data sources commence at 5 seconds apart after the first, and the malicious node, respectively, floods f packets every second (f may take on different values: 2, 5, 10, 50, and 100).

The training process proceeds as follows.

Step 1. Select the dimension or size (m) of the feature vectors.

Step 2. Set the frequency of flooding to 2 initially ($f = 2$ per second).

Step 3. For each of the mobile scenarios (0m/s, 10m/s, 20m/s, 30m/s, and 40m/s), simulate the MANET as follows. Each node records the inter-route time of a source node (T_i) on receiving a RREQ from the source node. Add T_i to the malicious history frequency vector if the source is malicious; otherwise it is added to the normal history frequency vector. At the end of this step for each scenario, two sets of vectors are established:

(i) 100 malicious vectors: $V_M^j(T_1^M, T_2^M, T_3^M, ..., T_m^M); \forall j = \overline{1..100}$;

(ii) 100 normal vectors: $\overline{V^j}$ $(\sum_{i=1}^{50} T_1^i/50, \sum_{i=1}^{50} T_2^i/50, \sum_{i=1}^{50} T_3^i/50, ..., \sum_{i=1}^{50} T_m^i/50)$; $\forall j = \overline{1..100}$.

Step 4. At the end of Step 3 for all 5 scenarios, 100 average vectors for MVC and 100 vectors for NVC are obtained for this particular flooding frequency (f=2).

Figure 5: Static network topology simulation for training, 50 UDPs connections and malicious node positioned at the square in the center.

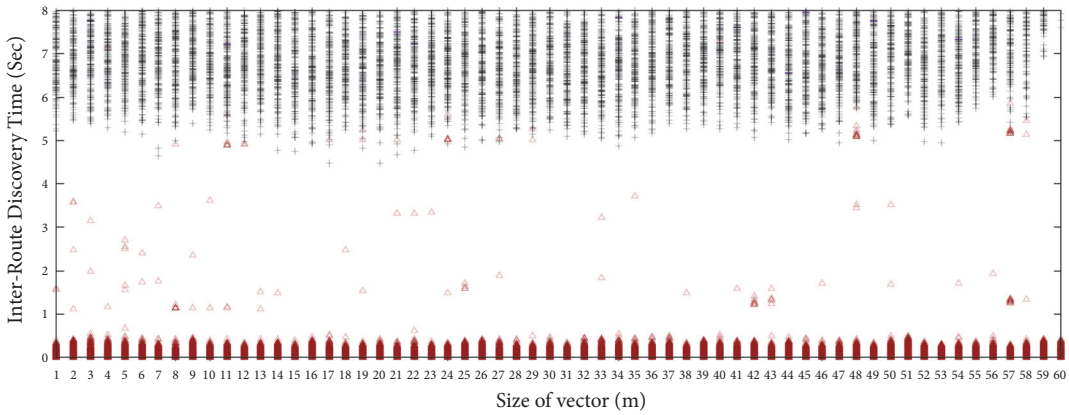


Figure 6: Two vectors class, black (+) for NVC and red (Δ) for MVC.

Step 5. The algorithm continues to establish MVC vectors and NVC vectors for other flooding frequencies (f = 5, 10, 50 and 100).

As a result of the training process, a training dataset with MVC and NVC vectors is shown in Figure 6. The training dataset is used to classify an unknown sample vector V (in the next section). In Figure 6, each vector is of size 60. It can be seen that there is an overlap between the two classes due to node mobility as well as the closeness of the rate of generation of RREQ packets of malicious and normal nodes.

3.3. Flooding Attack Detection Algorithm (FADA). All normal nodes collect route discovery information of source nodes in the network. On receiving a RREQ packet, a node employs the route discovery frequency vector (V_{Ns}) and uses a machine learning algorithm to determine if the source node is normal or malicious. The kNN-Classifier based on kNN [24] algorithm is utilized to classify the two classes based on the route discovery frequency vectors for NVC or MVC. The kNN algorithm is theoretically mature with low complexity that is widely used for data mining. The main idea is that if most of its k-nearest neighbors belong to a class, the sample belongs to the same class. In kNN, the nearest neighbor refers to the distance between two samples, and various distance metrics can be used based on the feature vector that represents the samples. One of the most popular choices is the Euclidean in (3) to calculate the distance between V_1 and V_2. Algorithm 1 describes our algorithm for recognizing malicious nodes.

$$d(V_1, V_2) = \sqrt{\sum_{i=1}^{m} (V_1[i] - V_2[i])^2} \quad (3)$$

3.4. FAPRP: A Novel Flooding Attacks Prevention Routing Protocol. In the original AODV protocol, as intermediate nodes accept all RREQ route discovery packets from any source nodes, hackers may exploit this vulnerability to perform RREQ flooding attacks. We propose the flooding attacks prevention routing protocol by introducing the flooding attacks detection algorithm into the route request phase of the AODV protocol as described in Figure 7. Similar to AODV, path discovery is entirely on-demand for FAPRP. When a

```
Input: Two class NVC and MVC, vector of route discovery frequency (V_Ns)
Output: True if V_Ns in NVC, else return False
Begin
    MAX_VECTOR = 500;
    Double Array disMVC [MAX_VECTOR], disNVC [MAX_VECTOR];
    For int vt = 1 to MAX_VECTOR do {
       disMVC[vt] = Euclidean (V_Ns, MVC.Vectors[vt]);
       disNVC[vt] = Euclidean (V_Ns, NVC.Vectors[vt]);
    }
    Sort (disMVC and disNVC, ASC); // ascending sort
    int k1 = k2 = 0;
    While (k1 + k2 < k) {
       if (disNVC[k1] < disMVC[k2]) k1++;
       else k2++;
    }
    Return (k1 > k2);
End
```

Algorithm 1: Flooding attack detection algorithm using kNN.

source node needs to send data packets to a destination node to which it has no available route, N_S broadcasts a RREQ packet to its neighbors. The intermediate node (N_i) receiving a RREQ packet from a preceding node (N_j) checks security as follows.

First, duplicate RREQ packets received by a node are dropped, similar to the AODV protocol. N_i may receive multiple RREQ packets coming from its neighboring nodes, but it only handles the first RREQ packet using the two parameters broadcast_id and src_add (source address) in the RREQ packet.

Second, unlike AODV routing protocol, N_i adds the information (s_i and e_i) to the route discovery history (RDH) of the source node. Each intermediate node stores the route discovery counter of all source nodes. If the value of the Counters[N_S] equals x, the source node N_S has initiated route discovery x times to this point. If the route history is full, N_i shifts all elements of RDH one position to the left and adds the new element (s_i, e_i) to the rightmost position.

In MANET, a source node sends and receives packets through its neighbor nodes. If all neighbor nodes of the source node reject packets, it will be isolated and cannot communicate with the other nodes in its network [13]. For this reason, in FAPRP routing protocol, only the source node's neighbor nodes deploy FADA algorithm to detect RREQ flooding attack. N_i uses the source node address and the preceding node address to determine if it is a neighbor of the source N_S. On receiving RREQ packets, the protocol works as follows.

Step A. If N_i is a neighbor of the source node N_S:

(i) N_i measures all T_i values in V_{Ns} using RDH of the source node.

(ii) If the route discovery frequency vector of source node (V_{Ns}) is not full, N_i ignores the security check and go to **Step B**.

(iii) Else, N_i uses FADA to classify N_S using its feature vector V_{Ns}.

(a) If V_{Ns} is in MVC, the source node is classified malicious, the RREQ packet is dropped, and the algorithm terminates.

(b) Else, go to **Step B**.

Step B. If N_i is not a neighbor of N_S, it executes other commands similar to AODV as follows:

(i) N_i saves broadcast_id and src_add values into its cache and adds a reverse route to source node into its routing table.

(ii) If N_i is destination or has a route toward the destination, it unicasts a RREP packet back to its neighbor from which it received the RREQ packet (N_j); otherwise, it rebroadcasts the RREQ packet.

When the destination node gets a RREQ, it updates the time instance e_i in the RDH of source node and unicasts a RREP packet to the source node through the reverse route. In the AODV protocol, there is no order information for the route response in the RREP packet. Therefore, N_i assumes that the RREP packet received is the response to the last route discovery. Thus, once the intermediate node receives an RREP packet, it updates e_i in the RDH of source node; that is, it sets i=Counters[N_S]. It increases the hop count field by 1 before forwarding the RREP packet back to the source node.

Example 2. Figure 8 describes how an intermediate node (N_i) handles the RREQ and RREP packets. First, on receiving RREQ packet at time p_1, N_i increases Counters[N_S] to 1 (Counters[N_S]=1) and records $s_1=e_1=p_1$. Second, on receiving the RREP packet at time p_2, N_i updates $e_1=p_2$. Next, at time p_3, N_i receives the RREQ packet, increases the Counters[N_S] by 1 (Counters[N_S]=2), and records $s_2=e_2=p_3$. Similarly, at time

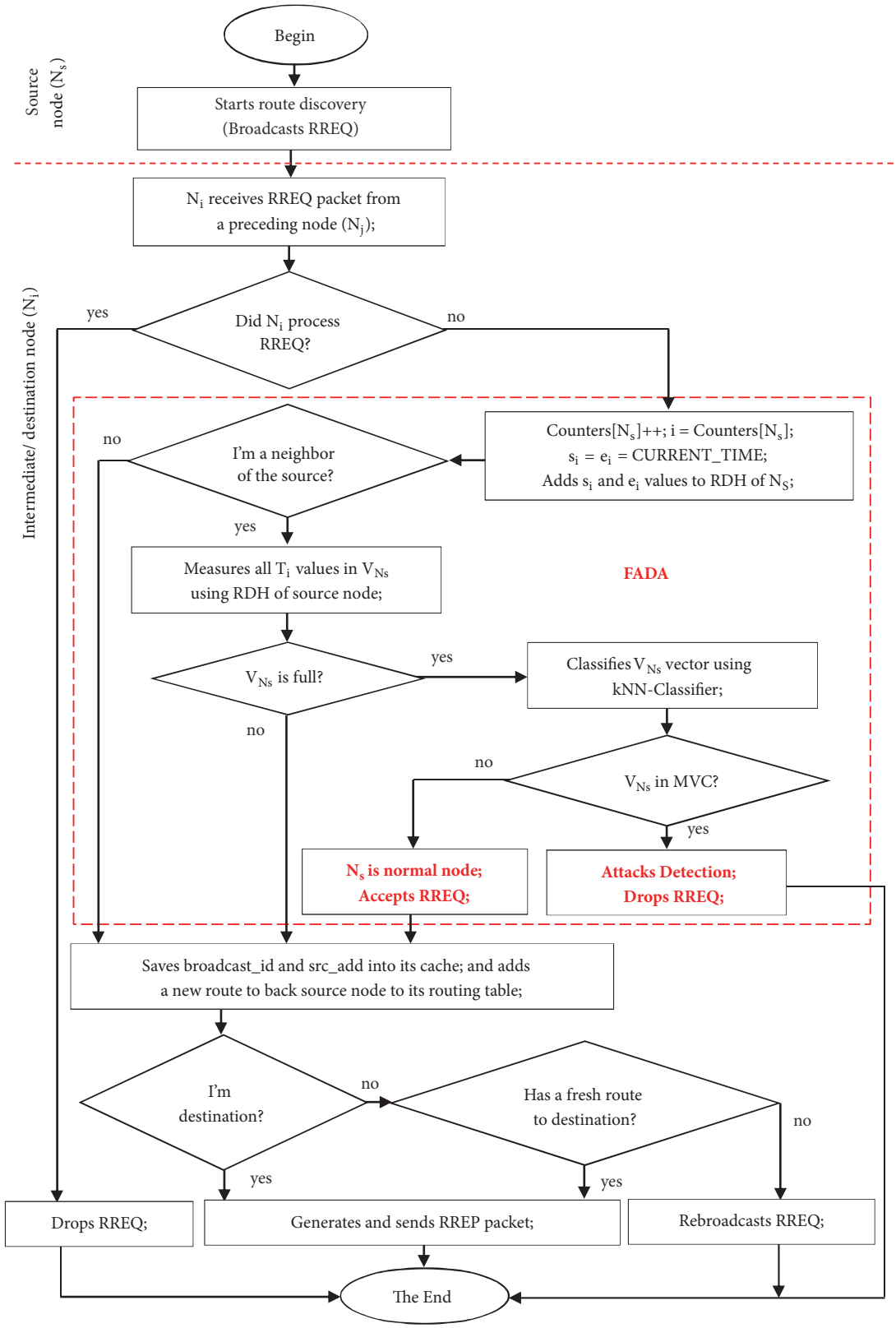


Figure 7: Request route process of FAPRP routing protocol.

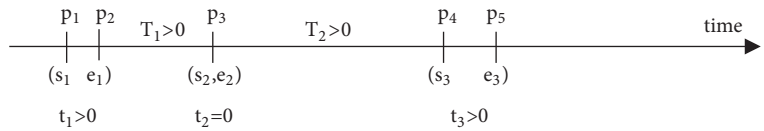


Figure 8: Route discovery history of the source node and 1 destination node.

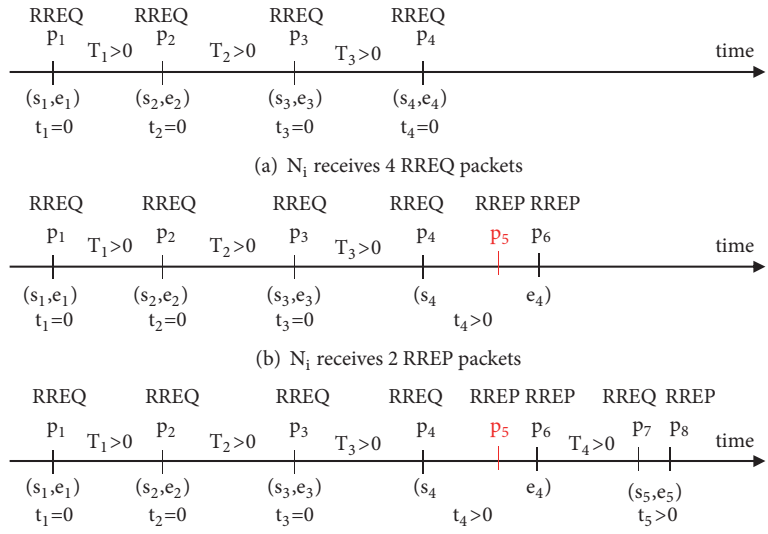


Figure 9: Route discovery history of a source and 3 destination nodes.

p_4, on receiving the next RREQ packet, Counters[N_S] is set to 3 and s_3=e_3=p_4. Finally, N_i records RREP at time p_5 and updates e_3=p_5. Because p_1<p_2 <p_3<p_4<p_5, $T_i > 0$, ∀i = 1..2. . ..

3.5. Discussion. RREQs may originate from the same N_S to many destination nodes (N_{D1}, N_{D2}, N_{Dn}). In this case, FADA only keeps the counter for N_S regardless of the destinations. This case is of interest because in detecting a malicious node, FADA only wants to see how often that node generates RREQ and does not care about the destinations.

Example 3. Using a network topology with n nodes, consisting of one source node N_S and three destination nodes N_{D1}, N_{D2}, and N_{D3}. Assume that N_S made route discovery seven times to three destination nodes N_{D1}, N_{D2}, and N_{D3}. Because of the mobile and noisy environment, 3 RREQ packets were lost, and N_i received only 4 RREQ packets at p_1, p_2, p_3, and p_4, respectively. The value of Counters[N_S] at N_i was then 4, which meant that as far as N_i was concerned, N_S has route discovered 4 times up to that point. Figure 9(a) shows the RDH of the N_S source node as recorded in N_i.

After p_4, N_i receives two RREP response packets to the source at p_5 and p_6. When receiving RREP at time p_5, N_i updates e_4=p_5, and N_i continues to update e_4=p_6 when receiving RREP packet at p_6. Figure 9(b) shows the RDH of the N_S source node after receiving two RREP packets.

Finally, N_i receives another RREQ packet from the N_S at time p_7 and a RREP packet at time p_8. On receiving this last RREQ, N_i increases Counters[N_S] by 1 (Counters[N_S]=5) and sets s_5=e_5=p_7, and on receiving the last RREP packet, N_i updates e_5=p_8. Figure 9(c) shows the RDH of the N_S source node at p_8.

Thus, based on the RDH of the source node, N_i can compute all T_i in V_{Ns} and use kNN-Classifier to decide if the source node is normal or malicious. In addition, all T_i values are larger than zero and it does not depend on the order of RREQ packets and the number of destination nodes.

4. Performance Evaluation by Simulation

In this section, we use NS2 [23] version 2.35 to evaluate the impact of RREQ flooding attacks on AODV and the proposed FAPRP protocol.

4.1. Simulation Settings. Similar to [13], our simulation scenarios cover a 1000 meter by 1000 meter flat space, accommodating 50 normal mobile nodes. We consider 2 scenarios: one with a malicious positioned at the center (Figure 10(a)) and the other with two malicious nodes positioned as shown in Figure 10(b). Each malicious node may flood the network at the rate of 10 or 20 packets per second.

The random waypoint [25] model is utilized as the mobility model. The minimum node speed for the simulations is 1 m/s while the maximum is 30m/s. In each simulation scenario, 20 sources transmit data at a constant bit rate (CBR). Each source transmits 512-byte data packets at the rate of 2 packets/second. The first source emits data at time 0, and

Table 3: Simulation parameters.

Parameters	Setting
Simulation area	1000 x 1000 (m^2)
Simulation time	500 (second)
Number of normal nodes	50 (nodes)
Node transmission range (R)	250 (m)
Number of malicious nodes	1, 2 (nodes)
Attacks frequency	10, 20 (packet/second)
Maximum speeds	1..10, 1..20 and 1..30 (m/s)
Transport protocol	UDP
Traffic type	CBR (constant bit rate)
Number of traffic	20
Data rate	2 (packet/second)
Packet size	512 (bytes)
Queue type	FIFO (DropTail)
Routing protocols	AODV, B-AODV [18], FAPRP
Size of vector (m)	10, 15, 20, 25, 30, 35, 40 and 60
Cutoff value (k)	10, 15, 20, 25, 30, 35, 40, 45 and 50
Distance type	Euclid

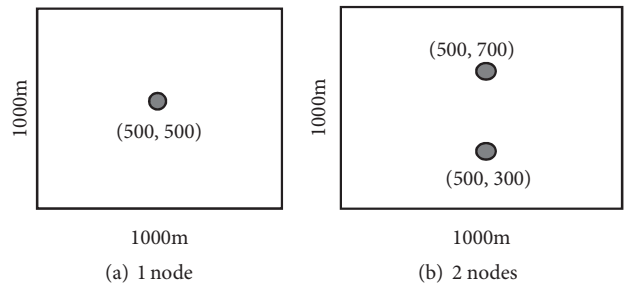


Figure 10: Malicious nodes location.

the following sources transmit data at 10 seconds apart. All parameters are described in Table 3.

We evaluate the original AODV, the B-AODV, and the FAPRP and compare their performance with and without RREQ flooding attacks in terms of attacks detection ratio, packet delivery ratio, end-to-end delay, and routing load metrics [18, 26].

(i) Attacks detection ratio (ADR) is calculated using (4). AT is the number of RREQ packets that are accepted true; the packets come from normal nodes. AF is the number of RREQ packets that are accepted false; the packets come from malicious nodes. DT is the number of RREQ packets that are dropped true; the packets come from malicious nodes. DF is the number of RREQ packets that are dropped false; the packets come from normal nodes.

$$ADR = \frac{AT + DT}{AT + AF + DT + DF} * 100\% \quad (4)$$

(ii) Packet delivery ratio (PDR) is the ratio of the received packets by the destination nodes to the packets sent by the source nodes (5), where n is number of data packets that are received by destination nodes and m is number of data packets that are sent by source nodes.

$$PDR = \frac{\sum_{i=1}^{n} DATA_i^{recieved}}{\sum_{j=1}^{m} DATA_j^{sent}} * 100\% \quad (5)$$

(iii) End-to-end delay (ETE) is the average delay between the sending time of a data packet by the CBR source and its reception at the corresponding CBR receiver (6), where $Delay_{DATA}^{i}$ is the delay time for sending i^{th} data packet to its destination successfully and n is number of data packets that are received by destination nodes.

$$ETE = \frac{\sum_{i=1}^{n} Delay_{DATA}^{i}}{n} \quad (6)$$

(iv) Routing load (RL) is the ratio of the overhead control packets sent (or forwarded) to successfully deliver data packets (7), where n is number of data

Table 4: AODV performances under flooding attacks.

Level	Number of MN	PDR (%)			RL (pkt)			ETE (Sec)		
		10m/s	20m/s	30m/s	10m/s	20m/s	30m/s	10m/s	20m/s	30m/s
0pkt/s	0	86.26	84.68	82.10	4.92	5.72	7.02	0.506	0.574	0.627
10pkt/s	1	72.63	68.68	64.13	25.45	28.61	30.61	1.032	1.232	1.304
	2	26.40	23.42	17.95	158.96	196.42	263.39	3.188	3.049	3.333
20pkt/s	1	28.75	25.57	19.63	140.55	171.48	228.57	3.292	3.013	3.059
	2	12.06	11.23	8.78	524.18	587.18	898.82	3.668	2.952	4.973
Standard deviation values										
0pkt/s	0	3.09	2.22	1.77	0.91	0.88	0.85	0.14	0.10	0.11
10pkt/s	1	3.92	7.43	2.10	1.86	6.10	1.46	0.19	0.35	0.06
	2	2.31	5.45	3.38	23.26	59.29	56.84	0.65	0.62	0.68
20pkt/s	1	2.69	6.25	3.80	21.13	45.05	44.26	0.32	0.37	0.65
	2	1.25	1.91	3.77	57.13	90.70	474.89	1.33	0.82	1.50

packets that are received by destination nodes and g *is* number of overhead control packets that are sent or forwarded. Routing discovery packets include legitimate RREQ, fake RREQ, RREP, HELLO, and RERR packets.

$$RL = \frac{\sum_{j=1}^{g} CONTROL\_PACKET_j^{overhead}}{\sum_{i=1}^{n} DATA_i^{recieved}} \quad (7)$$

4.2. Simulation Results

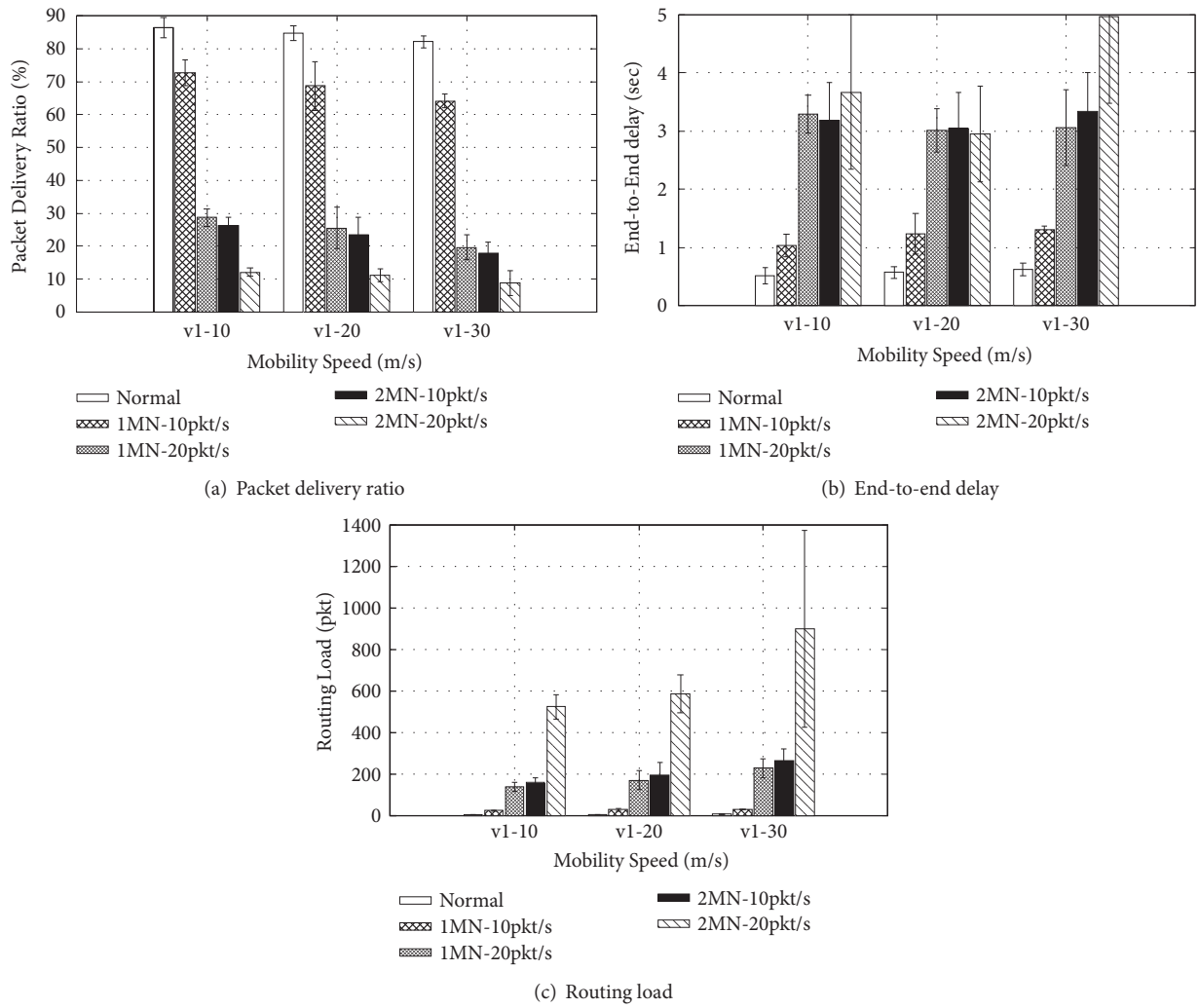
4.2.1. Effects of Flooding Attacks on the Original AODV Protocol. In this section we evaluate the performance of the AODV protocol with and without RREQ flooding attacks. We simulate 75 scenarios to evaluate the impact on the performance of AODV in terms of the above 4 defined metrics under various conditions including node mobility speeds, flooding frequencies, and malicious nodes. The main purpose of an RREQ flooding attack is to inject a large number of fake RREQ packets into the network making it less efficient in delivering legitimate packets. This effect is equivalent to handling excessive overhead packets causing a decrease in the network's packet delivery ratio, an increase in the average end-to-end packet delay, and an increase in the network's routing load. The simulation average results are shown in Table 4.

Figure 11 shows that the packet delivery ratio decreases, the routing load increases, and the end-to-end delay increases when the intruder floods attacking packets. Figure 11(a) shows that without flooding attack, the AODV packet delivery ratio is above 82.10% (1.77% standard deviation) and most packets reach their destination nodes. However, the packet delivery ratio reduced drastically to 12.06% (1.25% standard deviation) when the intruder uses 2 malicious nodes and floods 20 packets every second. Figure 11(b) shows that the average end-to-end delay increases as the flooding attack frequency increases. When the attacker uses 1 malicious node and broadcasts 10 RREQ packets every second, the average end-to-end delay changes from 0.506s before the attack to 1.032s after the attack for the 10m/s scenario. When the 2 malicious nodes broadcast 20 RREQ packets every second, the average end-to-end delay changes from 0.627s before the attack to 4.973s after the attack for the 30m/s scenario. Figure 11(c) shows that the routing load increases as the flooding attack frequency increases. When the attacker uses 1 malicious node and broadcasts 10 RREQ packets every second, the routing load changes from 4.92pkt before the attack to 25.45pkt after the attack for the 10m/s scenario. When the 2 malicious nodes broadcast 20 RREQ packets every second, the routing load changes from 7.02pkt before the attack to 898.82pkt after the attack for the 30m/s scenario.

4.2.2. Flooding Attacks Detection Performance of FAPRP. In this section we evaluate the malicious node detection performance of the proposed solution. Malicious node detection ratio is defined in (4). 216 scenarios are simulated: RDFV of size 10, 15, 20, 25, 30, 35, 40, and 60; the cutoff values of k for the kNN are set at 10, 15, 20, 25, 30, 35, 40, 45, and 50. Nodes move in a Random Way Point pattern with a specified maximum speed of 10m/s, 20m/s, and 30m/s. 20 source-destination UDP connections are set up among nodes. The intruder uses 2 malicious nodes and floods 20 packets every second.

The results in Figure 12 show that by making use of the route discovery history feature vector and the kNN machine data mining algorithm, our method achieves much higher malicious nodes detection ratios than those of existing algorithms and lower mistaken rates. The complexity of the overall detection algorithm is proportional to the size of the route discovery frequency vector. We see that the detection rate of FAPRP is above 99.0% and the mistaken rate is below 1.0% for all scenarios using RDFV vector sizes larger than 35. Figure 12(d) shows that the average of the maximum successful detection rate of FAPRP is above 99.77% when the cutoff value is 25 and RDFV vector size is 60. In brief, the proposed solution is effective in detecting the RREQ flooding attacks.

4.2.3. Performance Evaluation of AODV, B-AODV, and FAPRP. In this section we simulate 135 scenarios to evaluate the performance of the AODV, B-AODV, and FAPRP protocols under RREQ flooding attacks. The cutoff value



(a) Packet delivery ratio

(b) End-to-end delay

(c) Routing load

Figure 11: AODV performance under RREQ flooding attacks.

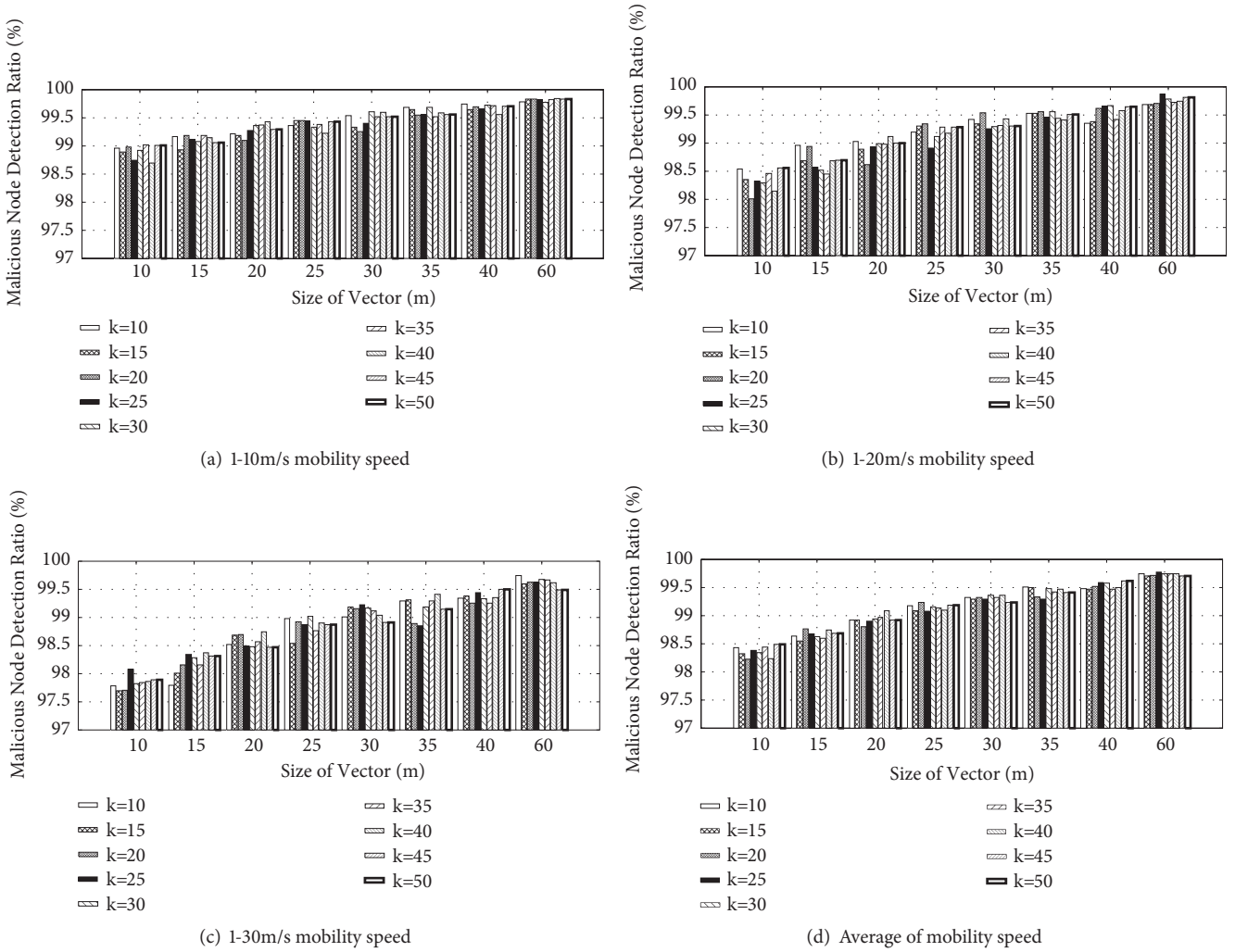
(k) is 25 and vector size (m) is 60. All nodes move in a Random Way Point pattern with specified maximum speeds of 10m/s, 20m/s, and 30m/s. Each of 2 malicious nodes floods 20 packets every second. 20 pairs of communicating nodes are set up among source nodes. The simulation average results are shown in Table 5.

(a) Packet Delivery Ratio. The results in Figure 13(a) show that the average packet delivery ratio for mobility speed by AODV is about 84.35% (1.86% standard deviation) in the absence of a malicious node. When there is one malicious node, the packet delivery ratio is about 24.65% (2.18% standard deviation) and 10.69% for two malicious nodes (0.9% standard deviation). This is due to RREQ flooding of the fake route request packets by the malicious node, resulting in a high consumption of bandwidth and buffer overloads at intermediate nodes with fake RREQs. For B-AODV in normal scenarios, the average packet delivery ratio is about 58.68% (3.16% standard deviation). In flooding scenarios, B-AODV average packet delivery ratio is above 59.32% when the intruder uses one or two malicious nodes. When our proposed solution is deployed, the packet delivery ratio for normal scenarios and high mobility speed is about 83.08% (2.47% standard deviation). Under flooding scenarios, FAPRP packet delivery ratio is above 82.06% when the intruder uses one or two malicious nodes, 2.73% maximum standard deviation. In brief, our solution is more efficient compared to AODV and B-AODV under normal network operation scenarios and more effective in handling RREQ flooding attacks with higher correct detection rates.

(b) End-to-End Delay. The results in Figure 13(b) show that with AODV, the average end-to-end delay is about 0.569s under normal scenarios. The end-to-end delays are about 3.121s and 3.864s for one and two malicious nodes, respectively. This high end-to-end delay is caused by the broadcasting of selective fake route request packets by the malicious nodes. For B-AODV under normal scenarios, the average end-to-end delay is about 1.091s. Under flooding scenarios, B-AODV end-to-end delay is about 1.056s with one malicious node and 1.145s with two malicious nodes. This is caused by the failure of B-AODV in detecting and preventing flooding

Table 5: AODV, B-AODV, and FAPRP performances.

				10m/s					
		PDR (%)			**RL (pkt)**			**ETE (sec)**	
MN	AODV	BAODV	FAPRP	AODV	BAODV	FAPRP	AODV	BAODV	FAPRP
0	86.26	59.89	84.73	4.92	3.11	4.69	0.506	0.790	0.526
1	28.75	55.01	83.94	140.55	4.13	6.05	3.292	0.865	0.566
2	12.06	59.30	83.80	524.18	5.98	7.34	3.668	0.921	0.598
				20m/s					
0	84.68	58.20	83.77	5.72	3.42	5.54	0.574	1.142	0.639
1	25.57	56.61	83.41	171.48	4.60	6.87	3.013	1.120	0.626
2	11.23	62.96	82.96	587.18	6.25	8.23	2.952	1.187	0.680
				30m/s					
0	82.10	57.96	80.75	7.02	3.57	6.60	0.627	1.342	0.703
1	19.63	57.50	79.92	228.57	4.88	8.05	3.059	1.185	0.813
2	8.78	55.69	79.41	898.82	7.09	9.28	4.973	1.327	0.798
				Average					
0	84.35	58.68	83.08	5.89	3.37	5.61	0.569	1.091	0.623
1	24.65	56.37	82.42	180.20	4.54	6.99	3.121	1.056	0.668
2	10.69	59.32	82.06	670.06	6.44	8.28	3.864	1.145	0.692
				Standard deviation values					
0	1.86	3.16	2.47	0.79	0.33	0.69	0.06	0.06	0.07
1	2.18	5.41	2.20	17.87	0.47	0.80	0.25	0.13	0.10
2	0.90	2.35	2.73	146.54	0.48	0.89	0.43	0.19	0.07



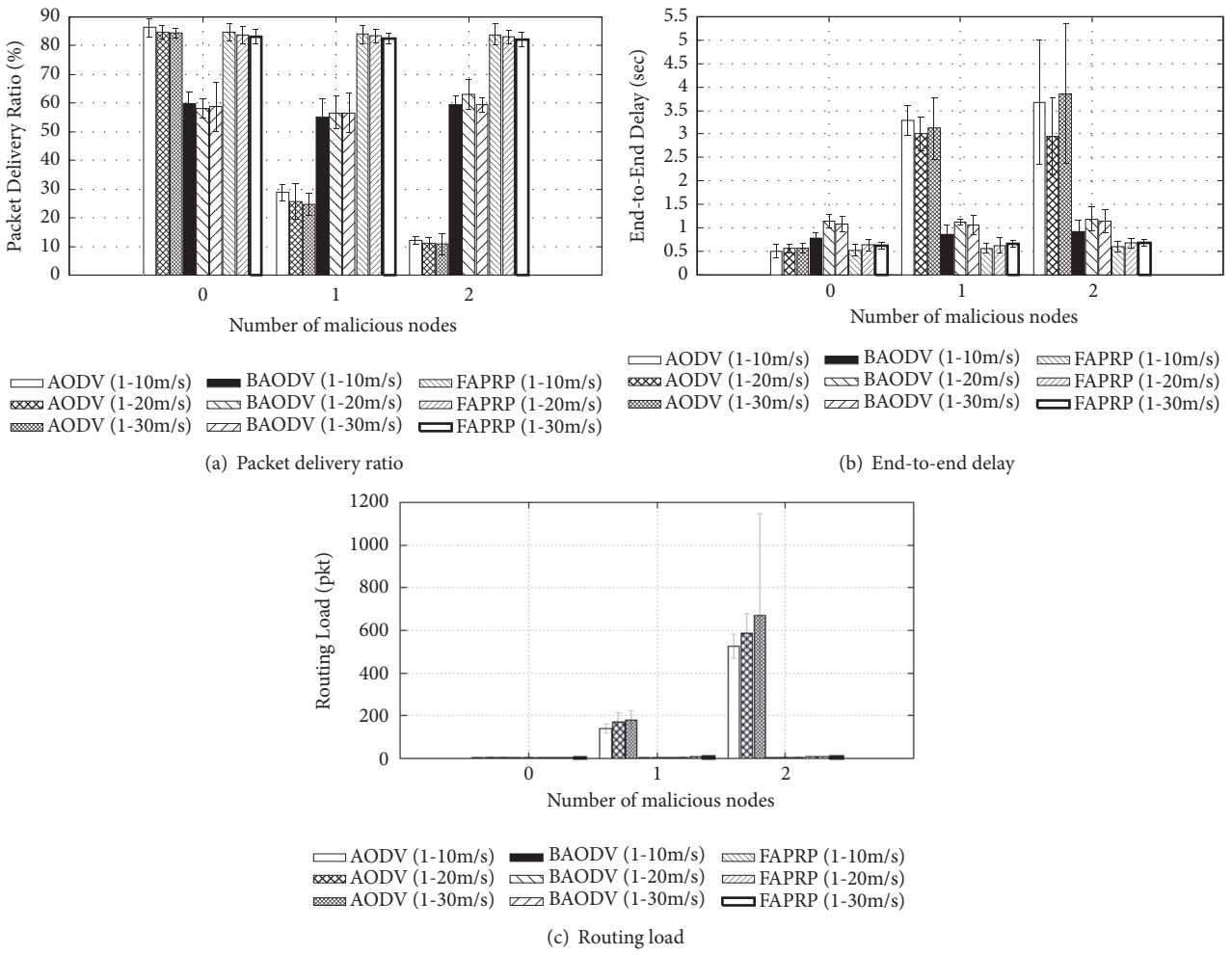
(a) 1-10m/s mobility speed

(b) 1-20m/s mobility speed

(c) 1-30m/s mobility speed

(d) Average of mobility speed

Figure 12: Malicious nodes successful detection ratio.



(a) Packet delivery ratio

(b) End-to-end delay

(c) Routing load

Figure 13: AODV, B-AODV, and FAPRP performances under RREQ flooding attacks.

attacks resulting in lower packet delivery ratios and longer route discovery delays. For our proposed solution, the average end-to-end delay for normal scenarios and mobility speed is about 0.623s. Under flooding attacks, FAPRP average end-to-end delays are about 0.668s and 0.692s when intruder uses one and two malicious nodes, respectively. Clearly, FAPRP achieves shorter end-to-end delay compared to AODV under flooding attack scenarios and B-AODV under both normal and flooding attack scenarios.

(c) Routing Load. The results in Figure 13(c) show that the average routing load for high mobility speed by AODV is about 5.89pkt in the absence of a malicious node. The routing loads are about 180.2pkts and 670.06pkts for one and two one malicious nodes, respectively. The high routing load is caused by the broadcasting of selective fake route request packets by the malicious nodes. For B-AODV in normal scenarios, the routing load is about 3.37pkt. B-AODV average routing load in attacks state is about 4.54pkt when the intruder uses one malicious node and 6.44pkt for two malicious nodes. For our proposed solution, the routing load for normal scenario and high mobility speed is about 5.61pkt. Under flooding attacks, FAPRP average routing load is about 6.99pkts and 8.28pkts when the intruder uses one and two malicious nodes, respectively. B-AODV routing load is, however, better as compared to AODV as it drops many route request packets due to mistake detection. Overall, FAPRP performs as well as AODV in the routing load measure under both normal and flooding attack scenarios due to its high correct detection rate and low mistake rate.

5. Conclusion

In this paper, we introduced the flooding attack detection algorithm based on our proposed route discovery frequency history feature vector and the kNN data mining algorithm to detect and isolate the malicious nodes in the network. We introduced a new FAPRP protocol by integrating FADA into the route request phase of AODV. Using route discovery frequency vector sizes larger than 35, the simulation results show that FADA achieves higher misbehaving detection ratio (above 99.0%) as compared with existing algorithms and lower mistaken rate (below 1.0%). Furthermore, the proposed solution is efficient in that it improves the network

performance in terms of higher packet delivery ratio, smaller end-to-end delay, and reduced routing load compared to AODV and B-AODV protocols.

In the future, we will extend the proposed solution for mitigating the effects of other flooding attacks.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the DThU (Dong Thap University), Vietnam, under the PhD Thesis (62.48.01.01) supervised by the Hue University of Sciences, Hue University.

Supplementary Materials

We submit the source code for the AODV, B-AODV, and FAPRP protocols and analysis files (.tcl, .awk) for the simulation with this revision. ID, file name, description are as follows: 1, aodv_cc.rar, source code of AODV routing protocol for simulation in NS2.35; 2, fdaodv_cc.rar, source code of FDAODV routing protocol for malicious node simulation in NS2.35; 3, baodv_cc.rar, source code of BAODV routing protocol for simulation in NS2.35; 4, fdbaodv_cc.rar, source code of FDBAODV routing protocol for malicious node simulation in NS2.35; 5, faprp_cc.rar, source code of FAPRP routing protocol for simulation in NS2.35; 6, fdfaprp_cc.rar, source code of FDFAPRP routing protocol for malicious node simulation in NS2.35; 7, scen.rar, 15 network topologies for simulation; 8, TCL.rar (TCL source code is used to write simulation script in ns2), analysis files (.awk) for the simulation; 9, DATA.rar, all simulation data; 10, Figures.rar, all scripts (gnuplot) to create the figures in the paper. *(Supplementary Materials)*

References

[1] J. Hoebeke, I. Moerman, B. Dhoedt, and P. Demeester, "An overview of mobile ad hoc networks: Applications and challenges," *Journal of the Communications Network*, vol. 3, no. 3, pp. 60–66, 2004.

[2] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99)*, pp. 90–100, New Orleans, La, USA, February 1999.

[3] C. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *Proceedings of the Conference on Communications Architectures, Protocols and Applications (SIGCOMM '94)*, pp. 234–244, London, UK, 1994.

[4] Z. J. Haas, M. R. Pearlman, and P. Samar, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks," *INTERNET-DRAFT*, pp. 1–11, 2002.

[5] E. Alotaibi and B. Mukherjee, "A survey on routing algorithms for wireless Ad-Hoc and mesh networks," *Computer Networks*, vol. 56, no. 2, pp. 940–965, 2012.

[6] R. Di Pietro, S. Guarino, N. V. Verde, and J. Domingo-Ferrer, "Security in wireless ad-hoc networks - A survey," *Computer Communications*, vol. 51, pp. 1–20, 2014.

[7] R. Mitchell and I.-R. Chen, "A survey of intrusion detection in wireless network applications," *Computer Communications*, vol. 42, pp. 1–23, 2014.

[8] M. Wazid and A. K. Das, "A Secure Group-Based Blackhole Node Detection Scheme for Hierarchical Wireless Sensor Networks," *Wireless Personal Communications*, vol. 94, no. 3, pp. 1165–1191, 2017.

[9] E. C. H. Ngai, J. Liu, and M. R. Lyu, "An efficient intruder detection algorithm against sinkhole attacks in wireless sensor networks," *Computer Communications*, vol. 30, no. 11-12, pp. 2353–2364, 2007.

[10] S. Gurung and S. Chauhan, "A novel approach for mitigating gray hole attack in MANET," *Wireless Networks*, vol. 24, no. 2, pp. 565–579, 2018.

[11] T. L. Ngoc and T. T. Vo, "Whirlwind: A new method to attack Routing Protocol in Mobile Ad hoc Network," *International Journal of Network Security*, vol. 19, no. 5, pp. 832–838, 2017.

[12] T. T. Vo, N. T. Luong, and D. Hoang, "MLAMAN: a novel multi-level authentication model and protocol for preventing wormhole attack in mobile ad hoc network," *Wireless Networks*, 2018.

[13] Y. Ping, D. Zhoulin, Y. Zhong, and Z. Shiyong, "Resisting flooding attacks in ad hoc networks," in *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)*, vol. 2, pp. 657–662, Las Vegas, NV, USA, April 2005.

[14] H. Ehsan and F. A. Khan, "Malicious AODV: Implementation and analysis of routing attacks in MANETs," in *Proceedings of the 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom-2012*, pp. 1181–1187, UK, June 2012.

[15] D. Gada, R. Gogri, P. Rathod et al., "A distributed security scheme for ad hoc networks," *The Crossroads Journal*, vol. 11, no. 1, pp. 1–14, 2004.

[16] J.-H. Song, F. Hong, and Y. Zhang, "Effective filtering scheme against RREQ flooding attack in mobile ad hoc networks," in *Proceedings of the 7th International Conference on Parallel and Distributed Computing, Applications and Technologies, PDCAT 2006*, pp. 497–502, Taiwan, December 2006.

[17] P. Yi, Y. Hou, Y. P. Zhong, and Z. L. Dai, "Flooding attack and defence in ad hoc networks," *Journal of Systems Engineering and Electronics*, vol. 17, no. 2, pp. 410–416, 2006.

[18] M. J. Faghihniya, S. M. Hosseini, and M. Tahmasebi, "Security upgrade against RREQ flooding attack by using balance index on vehicular ad hoc network," *Wireless Networks*, vol. 23, no. 6, pp. 1863–1874, 2017.

[19] S. Gurung and S. Chauhan, "A novel approach for mitigating route request flooding attack in MANET," *Wireless Networks*, vol. 24, no. 8, pp. 2899–2914, 2018.

[20] V. Thanh Tu and L. Thai Ngoc, "SMA2AODV: Routing protocol reduces the harm of flooding attacks in mobile ad hoc network," *Journal of Communications*, vol. 12, no. 7, pp. 371–378, 2017.

[21] M. Patel, S. Sharma, and D. Sharan, "Detection and prevention of flooding attack using SVM," in *Proceedings of the 3rd International Conference on Communication Systems and Network Technologies, CSNT 2013*, pp. 533–537, India, April 2013.

[22] W. Li, P. Yi, Y. Wu, L. Pan, and J. Li, "A New Intrusion Detection System Based on KNN Classification Algorithm in Wireless Sensor Network," *Journal of Electrical and Computer Engineering*, vol. 2014, Article ID 240217, 8 pages, 2014.

[23] DARPA, *The network simulator NS2*, 1995, https://www.isi.edu/nsnam/ns/.

[24] S. K. Sahu, P. Kumar, and A. P. Singh, "Modified K-NN algorithm for classification problems with improved accuracy," *International Journal of Information Technology*, vol. 10, no. 1, pp. 65–70, 2018.

[25] J. Yoon, M. Liu, and B. Noble, "Random waypoint considered harmful," in *Proceedings of the 22nd Annual Joint Conference on the IEEE Computer and Communications Societies (INFOCOM '03)*, vol. 2, pp. 1312–1321, San Francisco, Calif, USA, March-April 2003.

[26] S. Gurung and S. Chauhan, "A dynamic threshold based algorithm for improving security and performance of AODV under black-hole attack in MANET," *Wireless Networks*, pp. 1–11, 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 1954121, 14 pages
https://doi.org/10.1155/2018/1954121

Research Article

Security Vulnerabilities and Countermeasures for Time Synchronization in TSCH Networks

Wei Yang[1], **Yadong Wan**[2], **Jie He**[2] and **Yuanlong Cao**[1]

[1]*School of Software, Jiangxi Normal University, Nanchang, 330022, China*
[2]*School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing, 100083, China*

Correspondence should be addressed to Wei Yang; yw@jxnu.edu.cn

Received 20 August 2018; Revised 15 November 2018; Accepted 27 November 2018; Published 10 December 2018

Guest Editor: Jiageng Chen

Time-slotted channel hopping (TSCH), which can enable highly reliable and low-power wireless mesh networks, is the cornerstone of current industrial wireless standards. In a TSCH network, all nodes must maintain high-precision synchronization. If an adversary launches a time-synchronization attack on a TSCH network, the entire network communication system can be paralyzed. Thus, time-synchronization security is a key problem in this network. In this article, time synchronization is divided into single-hop pairwise, clusterwise, and three-level multihop according to the network scope. We deeply analyze their security vulnerabilities due to the TSCH technology itself and its high-precision synchronization requirements and identify the specific attacks; then, we propose corresponding security countermeasures. Finally, we built a test bed using 16 OpenMoteSTM nodes and the OpenWSN software to evaluate the performance of the proposed scheme. The experimental results showed that serious security vulnerabilities exist in time-synchronization protocols, and the proposed countermeasures can successfully defend against the attacks.

1. Introduction

Most industrial applications, e.g., steel mills, chemical industries, and oil refineries, need real-time monitoring and management processes [1, 2]. Traditionally, wired industrial automation and monitoring systems have been deployed to monitor temperature, pressure, or tank-fill levels. But it is difficult and expensive to install communication cables in a factory [3]. With the recent advances in wireless technology, industrial wireless sensor networks (IWSNs) have become a trend, instead of the traditional wired industrial systems. Their advantages are easier deployment and cheaper maintenance. In particular, they can be used in mobile objects and explosive environments.

As industrial wireless applications have critical requirements for reliability, low power, and real-time response rates, IWSNs face many challenges. Research has shown that wireless communication is vulnerable to external interference, path obstruction, and multipath fading. Moreover, nodes waste considerable energy in idle listening states, e.g., IEEE802.15.4-2006 networks [4].

The time-slotted channel-hopping (TSCH) technique can be applied to low-power and highly reliable wireless mesh networks. All nodes in a TSCH network must maintain high-precision synchronization. The nodes radios switch on according to the network schedule, which can avoid idle listening. It also uses a channel-hopping technique to improve the wireless communication reliability. Currently, the TSCH technology is the fundamental for the industrial wireless standards, e.g., ISA100.11a [5], WirelessHART [6], and IEEE802.15.4-2015 [7].

Figure 1 shows a sample timeslot-channel schedule in a TSCH network with a 101-slot super-frame. The horizontal axis represents ASN (absolute slot number) value which indicates how many timeslots have elapsed after the network formation. And the vertical axis represents communication channel. The communication of nodes happens in one timeslot and communication channel according to the network schedule. Some nodes can sleep in a specified timeslot to save energy (e.g., node B and node D sleep in timeslot 1). In addition, the network adopts a channel-hopping technology based on time to improve wireless communication reliability.

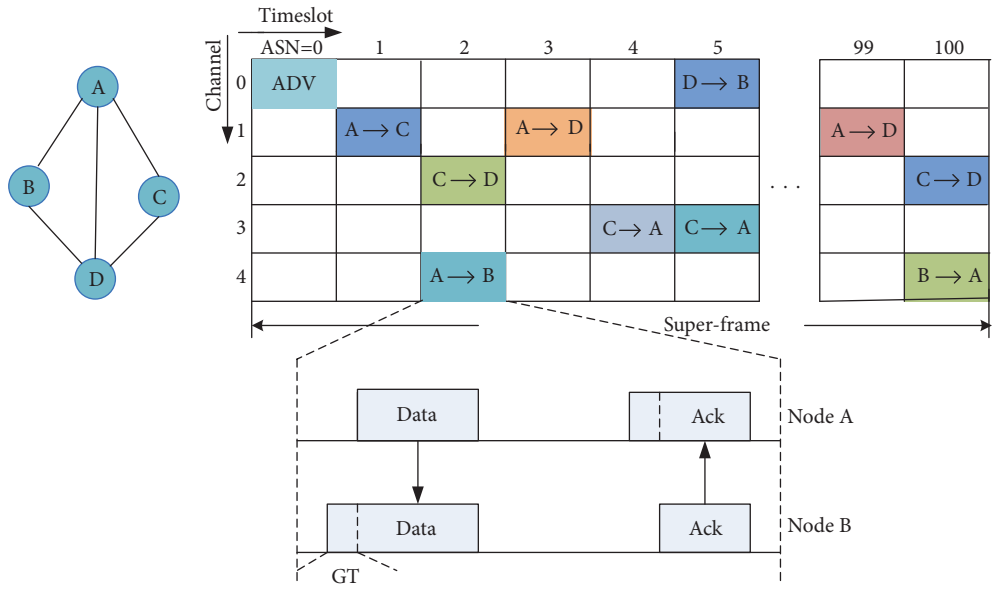


Figure 1: Sample timeslot-channel schedule in a TSCH network.

Overall, high-precision time synchronization is the core feature of TSCH networks.

Time synchronization is fundamental for the TSCH-based wireless networks. However, the time-synchronization protocol in TSCH networks focuses on energy efficiency and clock accuracy while ignoring security issues. If an adversary launches a time-synchronization attack on a TSCH network, the entire network communication system can be paralyzed. There are some secure time-synchronization protocols in WSNs (e.g., SPS [8] and SMTS [9]). But those countermeasures cannot be directly applied to the time-synchronization protocol in TSCH networks. The SPS (secure pairwise synchronization) protocol [8] is designed to defend against the pulse-delay attack in TPSN time synchronization. The SMTS (secured maximum-consensus-based time-synchronization) protocol [9] is designed to defend against message-manipulation attacks in MTS time synchronization. The time-synchronization protocol in TSCH networks is different from the MTS and TPSN protocol. There exist difference security vulnerabilities in the process of time synchronization. And the security countermeasures should also consider the detail of implementation of the time-synchronization protocol. It is necessary to research the secure time synchronization in TSCH-based wireless networks.

In our paper, time synchronization in TSCH networks is divided into single-hop pairwise, clusterwise, and multihop according to the network scope. We analyze in detail their security vulnerabilities, due to the high-precision synchronization requirements and TSCH protocol itself, and identify some specific attacks and then propose corresponding security countermeasures.

The contributions of the paper are threefold. Firstly, we analyze in detail the vulnerability of single-hop pairwise synchronization and propose security countermeasures that include an authentication mechanism and a clock-offset filter (COF) algorithm. The COF algorithm can filter out time-synchronization packets from malicious nodes. Secondly, we analyze in detail the vulnerability of clusterwise synchronization and propose an improved μTESLA scheme that supports immediate authentication. It does not need to wait until it receives the disclosed key before authenticating the packets. Finally, we analyze in detail the vulnerability of multihop synchronization and define an error-accumulation attack. We propose a multipath approach based on trust modeling, which can find a secure path to the root node by establishing a trust model between nodes.

A test bed using 16 OpenMoteSTM nodes and the OpenWSN software was built to validate the effectiveness and feasibility of the security countermeasures.

The paper is organized as follows. Section 2 introduces secure time-synchronization protocols in wireless sensor networks. Section 3 presents time synchronization in TSCH networks and the attack model. Sections 4, 5, and 6 describe secure pairwise, secure clusterwise, and secure multihop time synchronization, respectively. Section 7 presents the experimental evaluation of the secure time synchronization in a network with 16 OpenMoteSTM nodes. Section 8 presents a comparison with other secure time-synchronization protocols. Section 9 concludes the paper.

2. Related Work

Huang et al. [10] showed the seq_num attacks and global time attacks on the flooding time-synchronization protocol (FTSP). Then they proposed a series of countermeasures which include new root-selection and blacklist filter mechanisms to protect against the above attacks. Zhang et al. [11] observed a novel time-synchronization attack (TSA) that can manipulate the timing information in a smart grid. The attack

can reduce the fault-location performance and disable the voltage-instability alarm.

Ganeriwal et al. [8] defined a pulse-delay attack on the timing-synchronization protocol for sensor networks (TPSN). This attack causes legitimate nodes to calculate an error clock offset. Then, they proposed a SPS protocol that adopts message-authentication codes and end-to-end delay-estimation methods. The experiment, which is conducted on Mica2 motes, shows that SPS can successfully detect the attack.

He et al. [9] analyzed the vulnerability of the MTS protocol and described message-manipulation attacks. They then proposed a secure MTS protocol using hardware and logical clocks to detect the attacks. The result showed that the protocol can quickly compensate the clock offset. And they also proposed a SATS protocol [12] to protect against message-manipulation attacks in the average-consensus-based time-synchronization (ATS) protocol.

Dong et al. [13] showed Sybil attacks and compromise attacks can destroy a distributed time-synchronization protocol. They proposed RTSP that employs a graph-theoretical technology to detect attacks. But the security mechanism can not be directly used in centralized time-synchronization protocols.

Yang et al. [14] analyzed the vulnerability of the timing-synchronization protocol in IEEE802.15.4e networks. They pointed out the ASN and timeslot-template synchronization attack. Then, they proposed security countermeasures which fully consider the characteristic of IEEE802.15.4e networks to defend against the attacks. However, the paper only focused on secure single-hop synchronization, and only simulations were conducted to verify the effectiveness of the countermeasures.

3. Time Synchronization in TSCH Networks

In this section, we briefly describe time synchronization in TSCH networks. Then, we present a model for time-synchronization attacks.

3.1. Time-Synchronization Process. Every node keeps synchronized in TSCH networks, and the communication happens in timeslots (e.g., 15 ms long). Before a new node joins the network, it should receive the enhanced beacon (EB) packets from neighborhood nodes. The contents of EB packets mainly contain Join Priority (JP) value and ASN value. The new node prefers to choose a neighborhood node with a lower JP value to do ASN synchronization.

After the node successfully joins the network, it needs to maintain synchronization. The timeslots of network nodes should remain aligned. Usually, the nodes are equipped with an inexpensive oscillator to keep time. Because of the differences in temperature or fabrication, there is a clock drift between two oscillators (typically 30 ppm). So the nodes need to do device-to-device synchronization to compensate for the clock offset. Every node can use frame-based or acknowledgment-based synchronization methods to synchronize with the network.

Figure 2(a) illustrates a detailed process of frame-based synchronization between a transmitter and receiver. The receiver should turn on the radio a little earlier than the transmitter. The duration is defined as the guard time (GT). And the receiver needs to record the arriving time of the frame. Based on (1), the receiver can get the value of clock offset. And the receiver can synchronize to the transmitter when it updates the period based on (2). Figure 2(b) illustrates a process of acknowledgment-based synchronization. Here, the transmitter is a child node. It should first send a request frame to the receiver. The receiver needs to calculate the time offset based on (1). And the transmitter gets the time offset and synchronizes to the receiver based on (2).

$$Offset = Arrivetime - TsTxoffset \quad (1)$$

$$CurrentPeriod = NormalPeriod + Offset. \quad (2)$$

3.2. Attack Model. The Dolev-Yao threat model is a typical attack model where the attacker can eavesdrop on, modify, or forge communication messages in the network. Our attack model is based on this model. We consider two types of attack: external and internal.

In the external attack model, an attacker can eavesdrop on or modify messages but cannot obtain the secret key. So it is unable to impersonate a legitimate node. But an internal attacker can get a legal identity because it knows the secret key. The compromise attack is a typical internal attack. The compromised node can not only eavesdrop on network messages, but also forge legitimate network messages. Our paper will consider the impact of the two types of attacks on the time-synchronization protocol.

4. Secure Pairwise Time Synchronization

Two neighbor nodes usually adopt single-hop pairwise synchronization to establish relative clock offsets. In this section, we first analyze its security in depth and then propose a security countermeasure that includes authentication mechanisms and a COF algorithm.

4.1. Vulnerabilities in Pairwise Synchronization. ASN synchronization occurs when new nodes join the network. And the EB packet is an important control packet in the process of ASN synchronization. Figure 3 shows the detailed format of the EB packet. The EB packets mainly contain Join Priority (JP) value, ASN value, and channel-hopping template information for a new node to join the network. The EB packet may be encrypted to defend against eavesdrop attacks. It only adopts a shared key for encryption; otherwise, it will prevent new nodes from joining the network. However, from the ASN synchronization perspective, it should adopt a secret key to hide the *ASN* value. If a new node receives a forged EB packet that contains a false ASN value, it may deduce the wrong channel frequency for all the given pairwise communication according to (3). This is the first security vulnerability in the single-hop pairwise synchronization process.

$$frequency = (ASN + channelOffset)\,\%16. \quad (3)$$

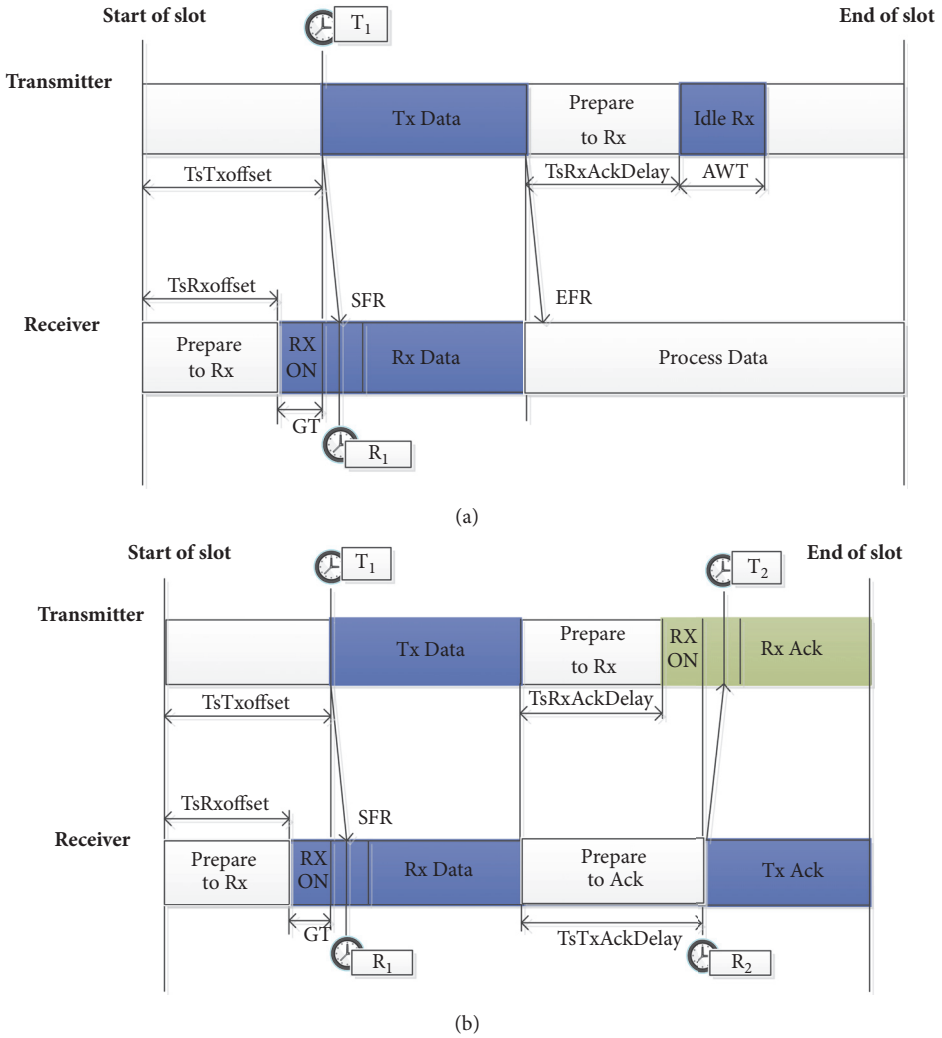


Figure 2: Two methods in device-to-device synchronization. (a) Frame-based synchronization. (b) Acknowledgment-based synchronization.

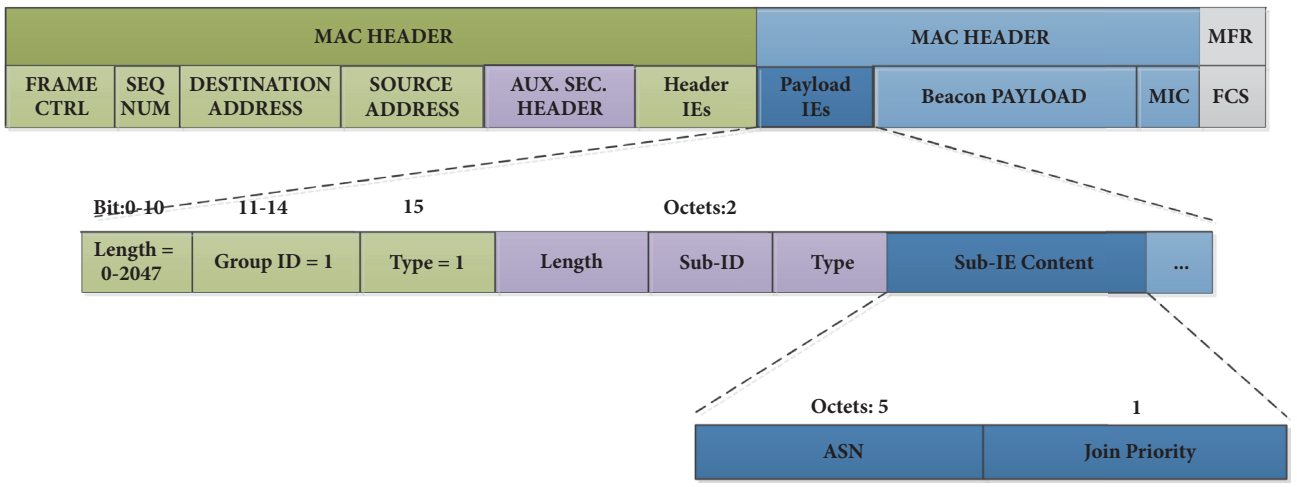


Figure 3: Enhanced beacon packet format.

In the process of device-to-device synchronization, the receiver may deduce the time offset according to (1). However, if an adversary modifies the $TsTxoffset$ value in the timeslot template, the receiver may calculate an incorrect time offset, according to (4). Δt represents the modification size. It is defined as a timeslot-template attack. However, the attacker cannot modify the $TsTxoffset$ value arbitrarily. If the value of Δt is larger than the GT, it may have no effect, as the receiver may turn off the radio in this case. This is the second security vulnerability in the single-hop pairwise synchronization process.

$$offset = timeReceived - TsTxoffset + \Delta t. \quad (4)$$

The pulse-delay attack which was first proposed by Ganeriwal et al. [8] may destroy device-to-device synchronization. To quote from Ganeriwal et al. [8]: "The adversary first sends jamming signals and replays it after a little delay, which will cause the receivers to calculate an incorrect clock offset." Both the frame-based and acknowledgment-based ways may be affected by this attack. This is the third security vulnerability in the single-hop pairwise synchronization process.

4.2. Secure Pairwise Synchronization. In the ASN synchronization process, a new node may receive a forged EB packet that contains a false ASN value. This occurs when joining a network. Fortunately, the IETF 6TiSCH working group recently made some progress in the secure Join protocol [16]. If a new node can securely join a network or choose a legitimate time source, it can successfully do ASN synchronization. Therefore, in our study, we mainly focus on the second and third security vulnerabilities in the pairwise synchronization process.

Generally, all nodes periodically perform device-to-device synchronization with their time parents in TSCH. The offset is usually less than a threshold when the period of synchronization is fixed. Therefore, the criterion of attack success is that the legitimate node receives a clock offset that exceeds the threshold value. Here, we present a mathematical model to describe the device-to-device time-synchronization process (see (5) and (6)). Let $T1$ represent the transmitter-recorded sending time of the synchronization frame and $R1$ represent the receiver-recorded receiving time of the synchronization frame. Similarly, $T2$ and $R2$ represent the relation time. For more detail, refer to Figure 2(b). Using (5) and (6), the time offset and transmission delay are easily calculated.

$$offset = \frac{(R1 - T1) - (T2 - R2)}{2} \quad (5)$$

$$delay = \frac{(R1 - T1) + (T2 - R2)}{2}. \quad (6)$$

First, let us consider the timeslot-template attack where an adversary modifies the value of $TsTxoffset$. Assume that the receiver is a malicious node that is compromised by an adversary. The $TsTxoffset$ value of receiver is modified to $TsTxoffset - \xi$. According to (1), the receiver would calculate a larger offset than the normal case. From the perspective of the mathematical model, the R1 and R2 value increase. Thus, according to (5) and (6), the adversary can conduct the offset increases, as indicated by (7), but the delay does not increase.

$$offset = \frac{(R1 - T1) - (T2 - R2)}{2} + \xi. \quad (7)$$

Second, the pulse-delay attack may destroy pairwise synchronization in TSCH. Assuming the adversary introduces delay Δ. The $T1$, $T2$, and $R2$ value do not change. But the $R1$ value may increase to $R1 + \Delta$. According to (5) and (6), it can conduct the offset increases, as indicated by (8), and the delay increases, as indicated by (9).

$$offset = \frac{(R1 - T1) - (T2 - R2) + \Delta}{2} \quad (8)$$

$$delay = \frac{(R1 - T1) + (T2 - R2) + \Delta}{2}. \quad (9)$$

In order to defend against time-synchronization attacks in the pairwise synchronization in TSCH, a secure algorithm is proposed as shown in Algorithm 1.

Our proposed scheme combines the message-integrity authentication mechanism and COF algorithm. In the process of child node A sending a synchronization request to parent node B, it contains random nonce N_A and MAC (message authentication codes). The random nonce N_A is used to protect against a reply attack. The MAC is an effective way to defend against an external attack which has been proved. However, the adversary may launch an internal attack such as a compromise attack or pulse-delay attack. Here we propose a COF algorithm to protect against the internal attack.

The core idea of the COF algorithm is that the synchronization packet will be filtered out when the clock offset or delay is larger than a threshold value. Therefore, how to accurately estimate the threshold value is a key issue. Let Q represent clock threshold and d^* represent delay threshold. When the period of synchronization T is fixed, the Q value can be estimated based on (10), where Max_drift represent the maximum clock drift between two nodes. The value of Max_drift can refer to the crystal manually. The value of d^* can also be estimated theoretically. Ganeriwal et al. [8] have proved that the transmission delay usually follows a Gaussian distribution. Let d represent the delay in the process of exchanging data packets, which is calculated according to (6). As the delay follows a Gaussian distribution, it can conclude that most of the delay is among $[d_{avg} - 3\sigma, d_{avg} + 3\sigma]$. It can reach 99.7% confidence. So the value of d can be estimated theoretically according to (11).

$$Q \leq T * Max\_drift \quad (10)$$

$$d^* = d_{avg} + 3\sigma. \quad (11)$$

5. Secure Clusterwise Time Synchronization

In this section, we first analyze in detail the vulnerability of clusterwise time synchronization and then propose a lightweight security countermeasure.

1: Node A sends a synchronization request to parent node B:
2: $A \longrightarrow B : A, B, T_1, N_A, MAC(K_{AB}, A, B, T_1, N_A)$;
3: Node B sends back an ACK message to A:
4: $B \longrightarrow A : B, A, T_1, R_1, R_2, N_A, ACK, MAC(K_{AB}, B, A, R_1, R_2, N_A, ACK)$;
5: Node A calculates:
6: $offset = \frac{(R_1 - T_1) - (T_2 - R_2)}{2}$, $delay = \frac{(R_1 - T_1) + (T_2 - R_2)}{2}$;
7: **if** ($offset \leq Q$&&delay <= d*)
8: $CurrentPeriod = NormalPeriod + offset$;
9: **else**
10: Filter out the synchronization packet, $num\_attack + +$;
11: **if** ($num\_attack > N$)
12: Blacklist(B), Send an alarm message to the Root;
13: **end**

Algorithm 1: Secure pairwise time synchronization.

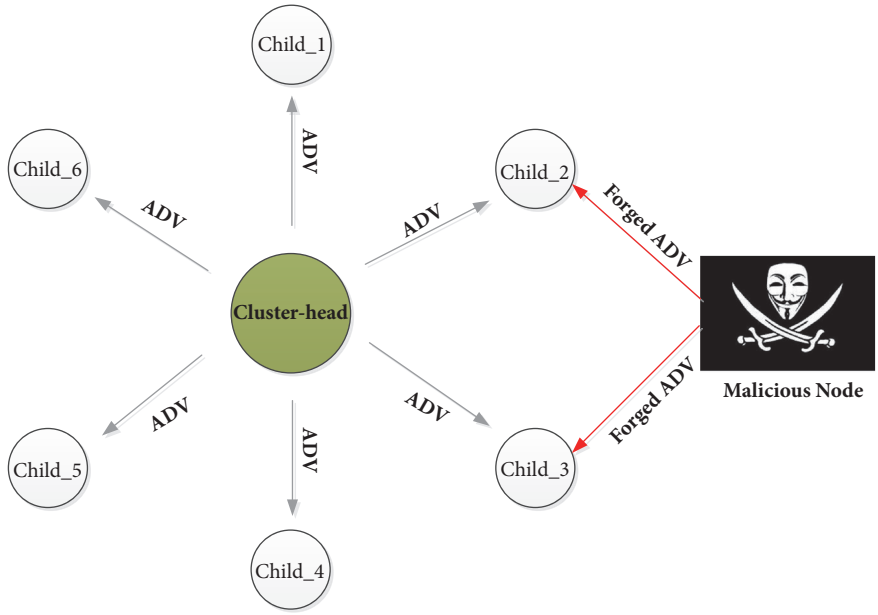


Figure 4: A malicious node can easily forge broadcast time-synchronization packets.

5.1. Vulnerabilities in Clusterwise Synchronization. To improve the energy efficiency, a cluster of nodes in TSCH networks may sleep in the given timeslots and wake simultaneously. Many other WSN applications (e.g., fire monitoring, speed estimating, and movement detection) also need consistent distributed coordination and sensing. Thus, a secure and low-power clusterwise synchronization protocol is critical.

In TSCH networks, a node has two methods for maintaining clusterwise time synchronization: KA-based and ADV-based. In the process of KA-based synchronization, every child node must periodically send a KA packet to the cluster-head to maintain synchronization. ADV-based synchronization is different because the cluster-head needs to periodically broadcast a time-synchronization packet to the child node. Vilajosana et al. [17] observe that the ADV-based synchronization scheme is more practical and energy efficient. Thus, our paper mainly focuses on the security vulnerability of ADV-based synchronization.

The ADV-based synchronization process has no security mechanisms and, in particular, no broadcast-authentication mechanism. A malicious node can easily forge broadcast time-synchronization packets to disturb the clusterwise synchronization.

Figure 4 shows the scenario of a malicious node launching an attack on ADV-based clusterwise synchronization. In the normal case, the cluster-head node should periodically broadcast an ADV packet. All the child nodes receive the ADV packet to do synchronization. However, the malicious node launches a time-synchronization attack on child_2 and child_3 by forging an ADV packet with a strong power. Child_2 and child_3 will calculate a false time offset and may lose synchronization with the cluster-head.

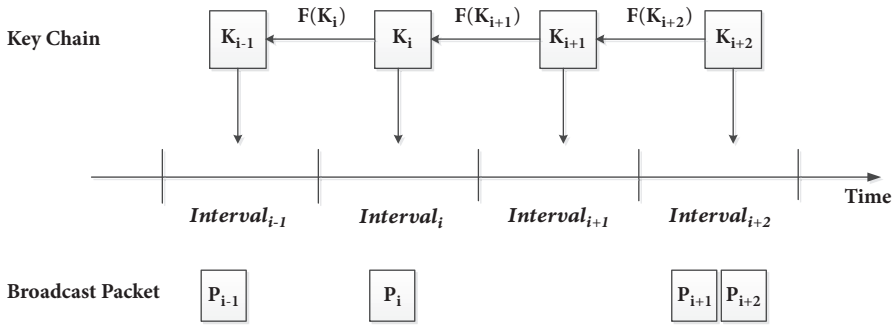


Figure 5: μTESLA broadcast-authentication mechanism.

In TSCH networks, all communications are based on high-precision synchronization. A false time offset can effectively reduce the performance of a cluster network. Thus, a secure clusterwise synchronization protocol should be designed for the TSCH network.

5.2. Secure Clusterwise Synchronization. In TSCH networks, the ADV-based synchronization scheme is more energy efficient than the KA-based synchronization scheme at making a cluster of nodes' synchronization. However, it has a serious security vulnerability. As it lacks a broadcast-authentication mechanism, a malicious node can easily impersonate a cluster to send false ADV packets. Therefore, it is necessary to adopt a broadcast-authentication mechanism to secure the network against this vulnerability.

Generally, digital signatures [18] and μTESLA [19] are typical mechanisms for authenticating broadcast packets in WSNs. Many studies have shown that digital signatures are computationally expensive, which cannot be used in resource-constrained WSNs. The μTESLA scheme is based on symmetric cryptography, which introduces low computation and communication overheads. However, the original μTESLA scheme exhibits several shortcomings.

One critical shortcoming is that the receivers must wait to authenticate packets until the cluster-head discloses the key. The longer delay means the receivers need more memory to buffer the synchronization packets. It also increases the risk of denial-of-service (DoS) attacks. We propose an improved μTESLA scheme in which receivers do not need to delay or buffer synchronization packets. In the following, we first review μTESLA and then propose an improved μTESLA scheme that allows receivers to authenticate most packets immediately upon arrival.

5.2.1. Overview of μTESLA. The core idea of μTESLA is that every broadcast packet attaches a MAC which is hard to forge. The child nodes first buffer the incoming packets until the cluster-head discloses the secret key. Only the cluster-head owns the secret key. A malicious node cannot forge a legitimate MAC unless it obtains the secret key K. Figure 5 illustrates the original μTESLA broadcast-authentication mechanisms. In μTESLA, the time axis is divided into many equal-length time intervals. Every interval is assigned a different secret key. When one or more broadcast packets are generated in a time interval, it uses the assigned secret key for authentication.

Usually, a hash function is adopted to generate a one-way key chain. According to (12), all the other keys can be calculated when the last key K_n is selected. Depending on the mathematical properties of the hash function, it is easy to authenticate the current disclosed key by previous keys, but it is hard to forge the later keys.

$$Key_i = Hash\left(Key_i + 1\right), \quad (i = 0, 1 \ldots n). \tag{12}$$

5.2.2. Improved μTESLA Scheme. The shortcoming of the original μTESLA scheme is that the receivers must wait until they receive the disclosed key before they authenticate the packets. Buffering packets may incur serious DoS attacks and might not be practical for secure time-synchronization applicants. We propose an improved μTESLA scheme that supports immediate authentication. Its main idea is that the cluster-head needs to buffer some packets during an interval and send the current packet with the hash value of a later packet. The receivers can then immediately authenticate current packets according to the hash value of the earlier packet.

Figure 6 shows the improved μTESLA scheme for secure clusterwise time synchronization in TSCH networks. The time is composed of a repeated super-frame. The super-frame is divided into many equal-length timeslots (15 ms). Packet P_j contains M_j, H(M_{j+1}), MAC (K_{i-1}, D_j), and K_{i-1}. Packet P_{j+1} contains M_{j+1}, H(M_{j+2}), MAC (K_i, D_{j+1}), and K_i. When the receivers receive packet P_{j+1}, they can immediately authenticate the current packet using H(M_{j+1}) of the earlier packet.

6. Secure Multihop Time Synchronization

In this section, we first briefly present the multihop time synchronization in TSCH and then perform an in-depth security analysis. Finally, we propose a secure multihop synchronization method that adopts a multipath scheme based on trust modeling.

6.1. Multihop Synchronization in TSCH Networks. The synchronization protocol introduced in the above sections aims to provide a clock offset between neighborhoods of nodes. Multihop TSCH networks usually consist of hundreds of

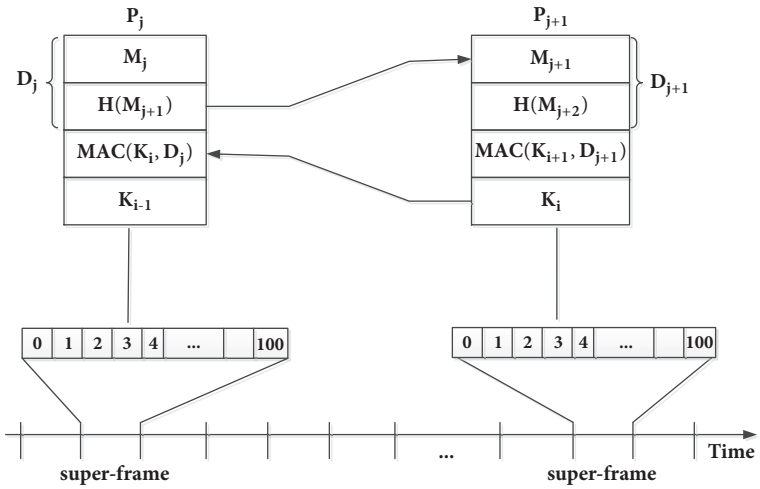


Figure 6: Improved μTESLA for secure clusterwise time synchronization.

nodes, and the distant child nodes must synchronize with the network root. In the multihop time-synchronization process, building a hierarchical network topology is a critical problem. The IETF 6TiSCH working group recommends the RPL (routing protocol for low power and lossy networks) [20] (see below) as a routing protocol to build the time-synchronization tree. The nodes far away from the root can synchronize step by step along the path of the time-synchronization tree.

Here, we briefly review the RPL; for more detail, refer to [20]. The RPL is an IPv6 routing protocol for low-power and lossy networks (LLNs) where both the routers and sensor nodes are resource-constrained. It is a distance-vector routing protocol. The RPL organizes the network topology as a Destination-Oriented Directed Acyclic Graph (DODAG) using Objective Functions (OFs). IETF 6TiSCH suggests reusing the DODAG structure: i.e., a node's routing parent is also its clock source. It not only prevents synchronization loops, but also reduces the energy cost to construct the synchronization tree.

6.2. Vulnerabilities in Multihop Synchronization. In multihop TSCH networks, the RPL plays a key role in constructing the time-synchronization tree. The distant child nodes synchronize along the edges of the tree. Having the procedure go smoothly relies on the assumption that none of the side nodes are malicious. Even a single malicious node along the multihop path can affect the time-synchronization performance. Unfortunately, the current RPL is not resilient against all of the attacks, especially the compromise attack. The adversary can easily control or instruct the compromised nodes to damage the time-synchronization services in an Internet-of-Things (IoT) environment. This may cause an error-accumulation phenomenon in the multihop time synchronization.

If a compromised node brings an incorrect offset to its immediate child node, the incorrect offset will propagate along the time-synchronization tree. Because the error-accumulation phenomenon exists in multihop networks, this attack may seriously damage the performance of TSCH networks, which require high-precision synchronization. In this paper, we define this attack as an error-accumulation attack.

Figure 7 illustrates the impact of an error-accumulation attack on multihop synchronization in a TSCH network. The network consists of five nodes. The root node is the reference clock source for the network. Node A1 is the immediate child node of the root. And node A2 is the immediate child node of A1, etc. As the phenomenon of clock drift exists, nodes need to synchronize periodically. Right before synchronizing, a clock offset exists between the node and its clock source (e.g., 200 s). Assume the GT is set to 1,000 s. This means the node may lose synchronization if the clock offset is bigger than 1,000 s.

We assume that, at some point, each node will synchronize just before its clock-source neighbor does. The phenomenon of the top half of Figure 7(a) may occur; i.e., the clock offset is 200 s between each node and its clock source. Let us further assume that the nodes synchronize from left to right; i.e., A1 synchronizes with the root, and then A2 synchronizes with A1. After the node A3 synchronization, the clock offset is about 800 s between A4 and A3. Because the clock offset is lower than the GT, it can successfully synchronize.

In the malicious setting, node A1 is a compromised node, as illustrated in Figure 7(b). The clock offset between compromised node A1 and the root may be 600 s if A1 waits a long time before synchronizing. The clock offset for each pair of legitimate neighbor nodes is 200 s. If the nodes synchronize from left to right, the last node, A4, may lose synchronization. After the node A3 synchronization, the clock offset between nodes A3 and A4 reaches 1,200 s (larger than the GT). This example fully shows the vulnerability of multihop synchronization in TSCH networks. The security of multihop synchronization must be enhanced for TSCH networks to be resilient against attacks.

6.3. Secure Multihop Synchronization. The error-accumulation attack seriously threatens multihop synchronization in TSCH networks. To effectively defend against such attacks,

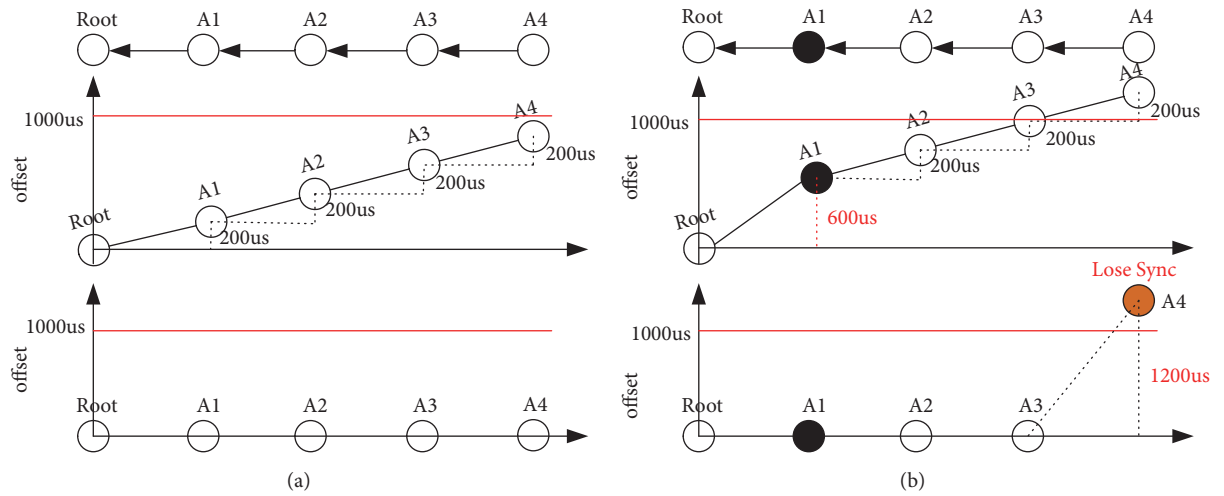


Figure 7: Error-accumulation attack on multihop synchronization in TSCH. (a) In the nonmalicious setting. (b) Under the malicious setting.

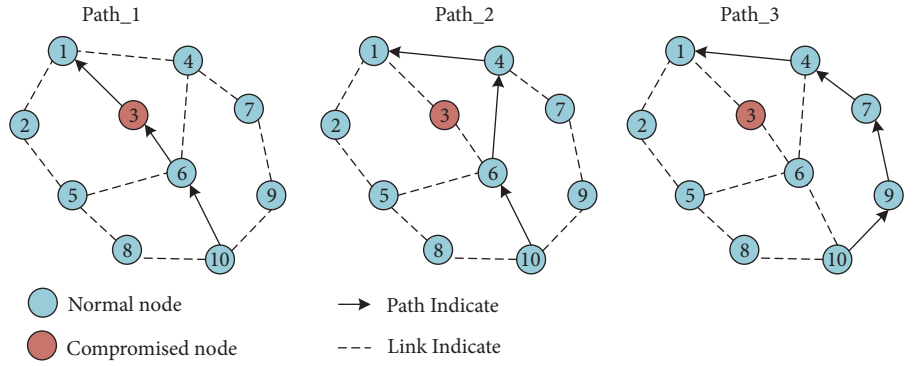


Figure 8: Diagram of multipath-based time synchronization.

we propose a multipath time-synchronization scheme based on trust modeling. The scheme can ensure that the nodes can find a secure path to the root nodes by establishing a trust model. Figure 8 shows node 10 finding a secure path to node 1, bypassing compromised node 3. If node 10 chooses Path_1 for synchronization, compromised node 3 can easily damage the multihop synchronization. If node 10 chooses Path_2 or Path_3 for synchronization, it will be successful. Thus, choosing a secure path from multihop paths is a challenge. Our paper proposes a scheme based on a trust model to solve this problem.

Here, we adopt an entropy-based trust model proposed by Sun et al. [21]. They proposed four axioms and derived a method for calculating a trust value, as follows:

$$P\{A : X, action\} = \frac{1 + \sum_{j=1}^{I} \beta^{t_c - t_j} k_j}{2 + \sum_{j=1}^{I} \beta^{t_c - t_j} N_j}, \tag{13}$$

where t_j represents the observation time, t_c represents the current time, k_j and N_j represent action times, and $0 \le \beta \le 1$ is the remembering factor.

The trust model quantifies the trust relationship between nodes. The trust values can be used to determine whether a node is legitimate or malicious. When a node has multiple neighbor nodes, it should select the node which owns the maximum trust value as the forward node. This way, it can bypass malicious nodes. Let $numTx\_A$ represent the total number of packets sent by node A, $numTx\_K$ represent the number of packets forwarded through node K, and $numTx\_K\_ACK$ represent the number of packets successfully forwarded through node K.

In our paper, the source of the nodes trust value is mainly considered from two aspects. One is the packet-forwarding rate $rate1 = numTx\_K/numTx\_A$; it reflects the cooperative relationship of the nodes. The other is the packet-forwarding success rate $rate2 = numTx\_K\_ACK/numTx\_K$. The trust value of a node is $T^d = \theta_1 T^w + \theta_2 T^r$, where $\theta_1, \theta_2 \in [0, 1]$ and $\theta 1 + \theta 2 = 1$. T^w can be calculated according to (14), and T^r can be calculated according to (15).

$$\begin{aligned} T^w &= P\{A : K, rate1\} \\ &= \frac{1 + \sum_{j=1}^{I} \beta^{G_A(t_c) - G_A(t_j)} numTx\_K}{2 + \sum_{j=1}^{I} \beta^{G_A(t_c) - G_A(t_j)} numTx\_A} \end{aligned} \tag{14}$$

$$\begin{aligned} T^r &= P\{A : K, rate2\} \\ &= \frac{1 + \sum_{j=1}^{I} \beta^{G_A(t_c) - G_A(t_j)} numTx\_K\_ACK}{2 + \sum_{j=1}^{I} \beta^{G_A(t_c) - G_A(t_j)} numTx\_K}. \end{aligned} \tag{15}$$

Node A observes node K at frequent intervals. $G_A(t_j)$ represents the observation time of node A, where $j = 1, 2\cdots, I$. $G_A(t_c)$ represents the current time. β is the remembering factor. The entropy-based trust model can bypass malicious nodes and find a secure path between a distant node and the root node. However, the value of β needs to be adjusted according to practical network applications.

7. Experiment Evaluation

In this section, we perform an evaluation through real experiments on a network of 16 OpenMoteSTM nodes running OpenWSN [22]. In the following, we first introduce an OpenMoteSTM node, the OpenWSN software, and the experimental setup. Then, we show the performance evaluation result of the proposed scheme.

7.1. OpenMoteSTM Nodes and OpenWSN Software. To verify the effectiveness of the proposed scheme, a low-power wireless sensor node called OpenMoteSTM was designed. Figure 9 shows the detailed structure of an OpenMoteSTM node. It incorporates a high-performance ARM-based 32-bit microcontroller and a short range radio. The microcontroller (STM32F103) can operate at 72 MHz, with 64 kB of embedded SRAM, and 512 kB of flash. It has a variety of peripherals, e.g., GPIOs, UART, SPI, ADC, and a timer.

The radio (AT86RF231) operates in 2.4 GHz and fully supports the IEEE802.15.4-2006 standard. The OpenMoteSTM node is also equipped with a 32.768 kHz crystal to drive the microcontroller timers. Because of the difference in manufacturing and working temperatures, the crystal has a typical drift of up to 30 ppm.

OpenWSN [22] is open-source software developed by UCB which support the IEEE802.15.4e TSCH protocol. The original OpenWSN implements time synchronization in TSCH but has no security mechanisms. Currently, OpenWSN can support a variety of hardware platforms, e.g., Guidance and Inertial Navigation Assistant (GINA), TelosB, and OpenMoteSTM nodes. It also offers the OpenVisualizer software, which can show the internal state (neighbor table, scheduling table, error reports, etc.) of each node in the network.

7.2. Experimental Setup. Figure 10 shows the test-bed scenario. The OpenMoteSTM nodes, which hang from the testing shelf, are all connected to the USB hub by USB lines. It not only provides power to every node, but also can view the state of the nodes through the OpenVisualizer software.

OpenWSN implements a time-synchronization protocol in TSCH. It adopts a hard-coded resource-scheduling algorithm [23], where a slotframe consists of 11 timeslots. The first timeslot is allocated for EB packets. And there are five shared timeslots allocated for data packets. The period for sending EB packets is set to 10 s. However, the original OpenWSN does not adopt any security mechanisms. We added code to implement the proposed security scheme on the OpenWSN platform.

7.3. Performance Evaluation. Here we adopt the following metrics for performance evaluation: synchronization error, energy consumption, and synchronization rate. The synchronization error reflects the synchronization precision in the process of synchronization. The energy consumption is a key metric in resource-constrained WSNs, which directly determines the feasibility of the scheme. The synchronization rate can reflect the percentage of nodes successfully synchronizing to the networks. Next, we will carefully analyze the performance of the secure pairwise and multihop synchronization.

7.3.1. Secure Pairwise Time Synchronization. We adopted a pair of OpenMoteSTM nodes to evaluate the performance of secure pairwise time synchronization. We introduce node A and node B. Node A needs to synchronize with node B at a period of 5 s. The GT is set to 1 ms. Assuming the adversary intermittently launches a pulse-delay attack, the synchronization error may vary with time. In real deployment, the attack parameter Δ is set to 0.8 ms.

Figure 11 shows that the comparison of synchronization error values in both the nonmalicious and malicious settings. In the nonmalicious setting, the value on the vertical axis is almost below 0.1 ms, in both the original and our proposed scheme. This illustrates that our proposed scheme does not affect the synchronization error. And we used a green wave line to show the fluctuation of the synchronization error. The synchronization error may increase with time because of the clock drift of each pair of nodes. After synchronizing, child node A compensates its clock according to (2). Thus, the synchronization error is reduced to a very small value.

In the malicious setting (where Δt is set to 0.8 ms), the synchronization error is very different between the original and our proposed scheme. Figure 11 shows the maximum value of the synchronization error almost reaches 0.9 ms in the original scheme. However, it only rises slightly in our proposed scheme because our proposed scheme can successfully detect the delay attack using the threshold-filter mechanism. As we know, the period of synchronization in our experiment is 5 s. According to (10), threshold Q is calculated to be 0.3 ms when Max_drift sets 60 ppm (a typical value). In our proposed scheme, when node A receives a synchronization packet from the malicious node, whose clock offset is larger than threshold Q, it will ignore the packet and increase the number of attacks. In the real experiment, the attack parameter Δt is 0.8 ms, larger than the threshold Q. Therefore, the synchronization packet will be ignored, and the synchronization error may go up a little with the clock drift of the pair of nodes. The experimental results not only show the impact of the attack, but also validate the effectiveness of our proposed scheme.

Energy consumption is a key metric in resource-constrained WSNs, which directly determines the feasibility of our scheme. The secure pairwise synchronization proposed in our paper adopts encryption and authentication mechanisms to protect against attacks. In the ASN synchronization process, an EB packet may be encrypted using a shared key to defend against eavesdrop attacks. The device-to-device

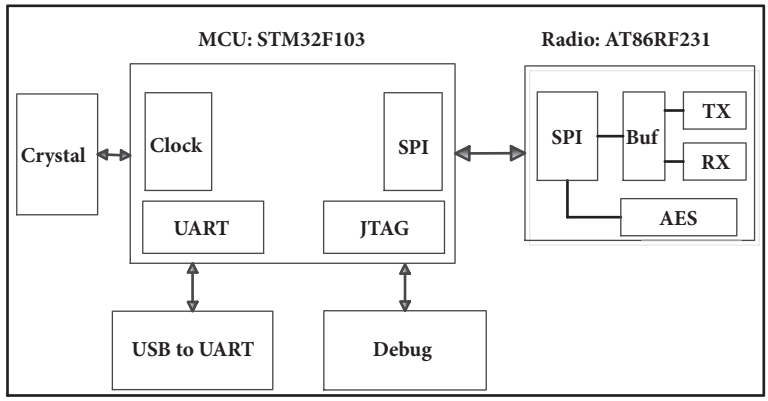


Figure 9: Structure of an OpenMoteSTM node.

Figure 10: Test-bed implementation of time synchronization in TSCH networks.

Table 1: Energy consumption in different security modes.

Security Mode	Energy
No Security	156 μJ
Encryption	171 μJ
Authentication	192 μJ
Encryption + Authentication	207 μJ

synchronization process needs message-integrity authentication to defend against attacks. OpenMoteSTM nodes adopt an AT86RF231 radio, which supports hardware-accelerated encryption and authentication. The AT86RF231 radio uses the Electronic Codebook (ECB) mode to encrypt messages and the Cipher Block Chaining (CBC) mode to generate the MAC.

The typical operating voltage U of a node is 3.3 V. The operating current can be measured by a current probe. The energy consumption can be calculated according to

$$E = U * I * T. \quad (16)$$

Table 1 illustrates the energy consumption in different security modes when the node transmits a 16-byte message. The energy consumption in the encryption mode only increases 9% compared to the no-security mode. However, it

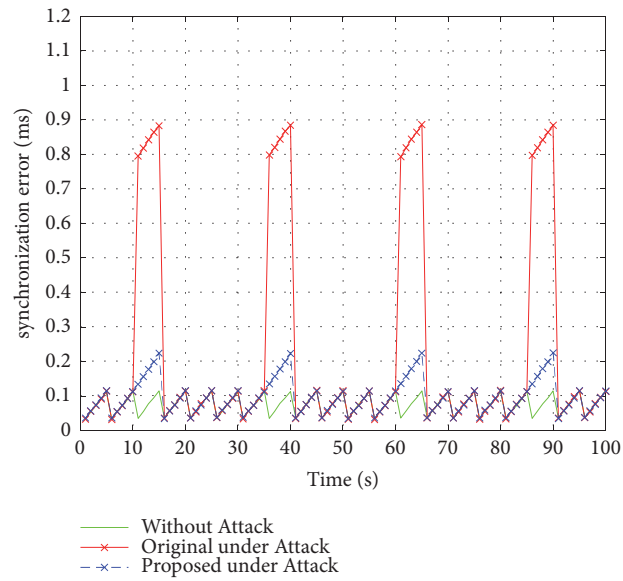


Figure 11: The comparison of synchronization error values under the different scenarios.

may increase 23% in the authentication mode. Generally, the device-to-device synchronization period is about 16 s. Thus, the energy consumption may not increase significantly from the network life cycle.

7.3.2. Secure Multihop Time Synchronization. We implemented a test bed as illustrated in Figure 10 to evaluate the effectiveness of secure multihop time synchronization. The real experiment uses 16 OpenMoteSTM nodes to build a three-hop-deep wireless network. The multihop network adopts the RPL to build the time-synchronization tree.

In the nonmalicious setting, all nodes are legitimate. The nodes far away from the root can synchronize step by step along the path of the time-synchronization tree. The pairs of neighborhood nodes use acknowledgment-based synchronization. The period of synchronization is set to 10 s and the GT is set to 1 ms. In the malicious setting, a compromised node launches an error-accumulation attack.

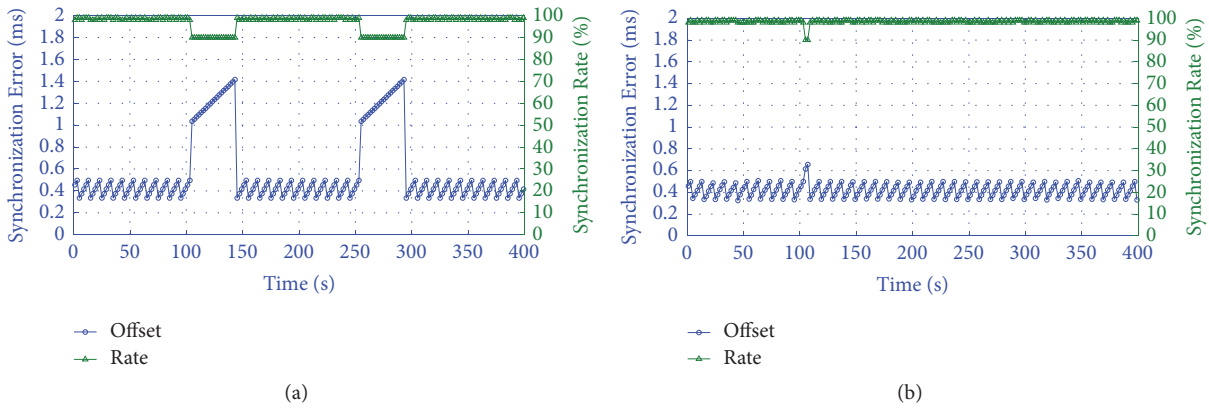


Figure 12: The synchronization error and synchronization rate vary with time in malicious settings. (a) Original scheme. (b) Our proposed scheme.

In real deployment, the compromised node intermittently launches attacks.

Figure 12 illustrates that the synchronization error and synchronization rate vary with time in malicious settings. The experiments were run 30 times and the average value was calculated. Figure 12(a) indicates the fluctuation of the synchronization error of the nodes and the synchronization rate in the original scheme. In the 0-100 s interval, the average synchronization error of the network stayed in a relatively stable state and the value was less than 0.55 ms. The synchronization rate almost reaches 99%. It indicates all nodes synchronize to the network. In the 100-150 s interval, the malicious node launches an error-accumulation attack. The average synchronization error of the network increases rapidly. And the synchronization rate drops to about 90% as some nodes have lost synchronization. In the 150-200 s interval, the synchronization error and synchronization rate return to the normal level as the malicious node stops attacking. Figure 12(b) indicates the fluctuation of the synchronization error of the nodes and the synchronization rate in our proposed scheme. In the 0-100 s, the average synchronization error and synchronization rate matched those for the original scheme. It illustrates that our proposed scheme does not affect the synchronization error. In the 100-150 s interval, the malicious node launches an error-accumulation attack. Compared to the original scheme, the impact of attacks on the synchronization error and network synchronization rate is much smaller. And in the 150-400 s interval, the synchronization error and network synchronization rate stay in the normal level, because our proposed scheme can ensure that the nodes can bypass the malicious node and find a secure path to the root nodes.

8. Comparison with Related Protocols

Time-synchronization protocols in WSNs usually divide into two categories: centralized synchronization and distributed synchronization. And secure time-synchronization protocols can also be divided into two categories. Table 2 shows the comparison of different secure time-synchronization protocols. He et al. [12] analyzed the vulnerability of the ATS protocol and described message-manipulation attacks. They then proposed a SATS (secure ATS) protocol using hardware and logical clocks to detect the attacks. The result showed that the SATS protocol can successfully defend against message-manipulation attacks. Ganeriwal et al. [8] pointed out the possible attacks on TPSN, such as a forge and modify attack, pulse-delay attack, and compromise attack. These attacks can mislead legitimate nodes to calculate an error clock offset. They then proposed a SPS protocol, which adopted message-authentication and end-to-end delay-estimation methods, to defend against the above attacks. The experiments showed that SPS can successfully protect against the attacks. Maximum-consensus-based time synchronization (MTS) is a typical distributed synchronization in WSNs. An adversary may easily destroy the MTS protocol by launching message-manipulation attacks. The noise-resilient MTS (NMTS) protocol [15], which adopted skew estimation, clock skew, and offset compensation mechanisms, can successfully defend against the attacks. However, the above secure protocols cannot be directly applied to the time-synchronization protocol in TSCH networks. The time-synchronization protocol in TSCH networks is different from the ATS, MTS, and TPSN protocol. There exist difference security vulnerabilities in the process of time synchronization. To the best of our knowledge, the current article is the first to provide an in-depth security analysis of the time synchronization in TSCH networks. We conducted an in-depth security analysis of the single-hop pairwise, clusterwise, and multihop time synchronization. The adversary may easily destroy the time-synchronization protocol by launching the reply attack, pulse-delay attack, timeslot-template attack, and error-accumulation attack. Security countermeasures, which include authentication mechanisms, the COF algorithm, improved μTESLA, and the multipath approach based on trust modeling, were proposed to protect against those attacks. The experimental results validated the effectiveness and feasibility of the security countermeasures. Moreover, the energy cost of our proposed protocol is very low.

9. Conclusions

The current time-synchronization protocol in TSCH networks has security vulnerabilities and is easily exploited by adversaries. We developed a suite of protocols for secure

Table 2: Comparison of different secure time synchronization protocols.

Secure Protocols	SATS [12]	SPS [8]	NMTS [15]	Our Proposed
Time Synchronization	ATS (Distributed Synchronization)	TPSN (Centralized Synchronization)	MTS (Distributed Synchronization)	TSCH (Centralized Synchronization)
Vulnerabilities	Reply Attack Pulse-Delay Attack Message Manipulation Attack	Forge/Modify Attack Pulse-Delay Attack Compromise Attack	Pulse-Delay Attack Message Manipulation Attack	Reply Attack Pulse-Delay Attack Timeslot-template Attack Error-accumulation Attack
Countermeasures	Logical Clock Checking Hardware Clock Checking	Encryption Authentication μTESLA	Maximum Consensus Skew Estimation Offset Compensation	Authentication COF Algorithm Improved μTESLA Multipath Approach
Energy Cost	High	Medium	High	Low

pairwise, secure clusterwise, and secure multihop time synchronization.

We conducted an in-depth security analysis of the single-hop pairwise time synchronization and proposed a security countermeasure that includes authentication mechanisms and a COF algorithm. The COF algorithm produced a filter based on a typical clock model to filter out time-synchronization packets from malicious nodes; it was used to defend against timeslot-template attacks. The experimental results showed that the synchronization error was very low, even in the malicious settings.

In clusterwise time synchronization, an adversary can easily forge broadcast time-synchronization packets to disturb normal synchronization. We proposed an improved μTESLA scheme that supports immediate authentication. The receivers did not need to wait a long time to receive the disclosed key and authenticate the packets. We provided an in-depth security analysis of multihop time synchronization and described an error-accumulation attack. We proposed a multipath approach based on trust modeling, which could find a secure path to the root node by establishing a trust model between nodes. Finally, a test bed using 16 OpenMoteSTM nodes and the OpenWSN software was built. The experimental results validated the effectiveness and feasibility of the security countermeasures.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant No. 61741125, the Natural Science Foundation of Jiangxi Province under Grant No. 20171BAB212014, and the National High Technology Research and Development Program under Grant No. 2014AA041801-2.

References

[1] Z. Sheng, C. Mahapatra, C. Zhu, and V. C. Leung, "Recent advances in industrial wireless sensor networks toward efficient management in IoT," *IEEE Access*, vol. 3, pp. 622–637, 2015.

[2] Q.-P. Chi, H.-R. Yan, C. Zhang, Z.-B. Pang, and L. D. Xu, "A reconfigurable smart sensor interface for industrial WSN in IoT environment," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1417–1425, 2014.

[3] G. Han, L. Liu, J. Jiang, L. Shu, and G. Hancke, "Analysis of energy-efficient connected target coverage algorithms for industrial wireless sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 1, pp. 135–143, 2017.

[4] IEEE Standard 802.15.4-2006, "IEEE Standard for Information Technology, Local and Metropolitan Area Networks, Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs)," 2006.

[5] International Society of Automation, "ISA-100.11a-2011: Wireless systems for industrial automation: process control and related applications," 2011.

[6] HART Communication Foundation, "WirelessHART Specification 75: TDMA Data-Link Layer," *IEEE Transactions on Vehicular Technology*, 2008.

[7] IEEE Standard 802.15.4-2015, "IEEE 802.15.4-2015: IEEE Approved Draft Standard for Low-Rate Wireless Personal Area Networks (WPANs)," 2015.

[8] S. Ganeriwal, C. Pöpper, S. Čapkun, and M. B. Srivastava, "Secure Time Synchronization in Sensor Networks," *ACM Transactions on Information and System Security*, vol. 11, no. 4, pp. 23–57, 2008.

[9] J. He, J. Chen, P. Cheng et al., "Secure time synchronization in wireless sensor networks: A maximum consensus-based approach," *IEEE Transactions on Parallel & Distributed Systems*, vol. 25, no. 4, pp. 1055–1065, 2014.

[10] D.-J. Huang, W.-C. Teng, and K.-T. Yang, "Secured flooding time synchronization protocol with moderator," *International Journal of Communication Systems*, vol. 26, no. 9, pp. 1092–1115, 2013.

[11] Z. Zhang, S. Gong, A. D. Dimitrovski, and H. Li, "Time synchronization attack in smart grid: Impact and analysis," *IEEE Transactions on Smart Grid*, vol. 4, no. 1, pp. 87–98, 2013.

[12] J. He, P. Cheng, L. Shi, and J. Chen, "SATS: secure average-consensus-based time synchronization in wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 61, no. 24, pp. 6387–6400, 2013.

[13] W. Dong and X. Liu, "Robust and secure time-synchronization against Sybil attacks for sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 6, pp. 1482–1491, 2015.

[14] W. Yang, Y. Wan, and Q. Wang, "Enhanced secure time synchronisation protocol for IEEE802.15.4e-based industrial Internet of Things," *IET Information Security*, vol. 11, no. 6, pp. 369–376, 2017.

[15] J. He, X. Duan, P. Cheng, L. Shi, and L. Cai, "Distributed time synchronization under bounded noise in wireless sensor networks," in *Proceedings of the 2014 IEEE 53rd Annual Conference on Decision and Control (CDC)*, pp. 6883–6888, Los Angeles, CA, USA, December 2014.

[16] D. Dujovne, T. Watteyne, X. Vilajosana, and P. Thubert, "6TiSCH: Deterministic IP-enabled industrial internet (of things)," *IEEE Communications Magazine*, vol. 52, no. 12, pp. 36–41, 2014.

[17] X. Vilajosana, Q. Wang, F. Chraim, T. Watteyne, T. Chang, and K. S. J. Pister, "A realistic energy consumption model for TSCH networks," *IEEE Sensors Journal*, vol. 14, no. 2, pp. 482–489, 2014.

[18] K.-A. Shim, Y.-R. Lee, and C.-M. Park, "EIBAS: an efficient identity-based broadcast authentication scheme in wireless sensor networks," *Ad Hoc Networks*, vol. 11, no. 1, pp. 182–189, 2013.

[19] K. Ren, S. Yu, W. Lou, and Y. Zhang, "Multi-user broadcast authentication in wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 8, pp. 4554–4564, 2009.

[20] T. Winter, P. Thubert, A. Brandt et al., "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC Editor RFC6550, 2012.

[21] Y. L. Sun, W. Yu, and Z. Han, "Information theoretic framework of trust modeling and evaluation for ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 305–315, 2006.

[22] T. Watteyne, X. Vilajosana, B. Kerkez et al., "OpenWSN: a standards-based low-power wireless development environment," *European Transactions on Telecommunications*, vol. 23, no. 5, pp. 480–493, 2012.

[23] T. Zhang, T. Gong, C. Gu et al., "Distributed Dynamic Packet Scheduling for Handling Disturbances in Real-Time Wireless Networks," in *Proceedings of the 2017 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pp. 261–272, Pittsburg, PA, USA, April 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 5929828, 10 pages
https://doi.org/10.1155/2018/5929828

Research Article

A Hierarchical Matrix Decomposition-Based Signcryption without Key-Recovery in Large-Scale WSN

Chi Yuan, Wenping Chen, and Deying Li

School of Information, Renmin University of China, China

Correspondence should be addressed to Deying Li; deyingli@ruc.edu.cn

Received 16 July 2018; Revised 19 October 2018; Accepted 13 November 2018; Published 2 December 2018

Guest Editor: Georgios Kambourakis

The sensors in wireless sensor network (WSN) are vulnerable to malicious attacks due to the transmission nature of wireless media. Secure and authenticated message delivery with low energy consumption is one of the major aims in WSN. The identity-based key authentication scheme is more suitable for the WSN. In this paper, the Hierarchical Matrix Decomposition-based Signcryption (HMDS) algorithm was proposed, which is a kind of identity-based authentication scheme. In HMDS scheme, three-layer architecture, base station (BS), cluster head, and intracluster, is employed to adapt to the common structure of WSN. As the key generation center (KGC), the BS adopts matrix decomposition to generate the identification information and public key for cluster head, which not only reduces the cost of calculation and storage but also avoids the collusion attack. Experiments show that the HMDS algorithm has more advantages over other algorithms and is very suitable for the large-scale WSN.

1. Introduction

Wireless sensor network (WSN) combined sensor and network communication technology has real-time sensing and information acquisition functions. It is applied to various fields such as national defense, environment monitoring, transportation management, medical treatment, and public health [1, 2]. The wireless sensor network has the characteristics of large-scale, multihop communication and complex deployment environment, and so forth [3, 4]. At the same time, since the sensor network nodes mostly have small size and are deployed in unattended harsh environment, the sensor node energy as well as the computing capability is greatly limited. Therefore, low energy cost and computation complexity are important requirements for WSN application [5]. In addition, WSN is different from the wired network, the communication channel of wireless network is public, and it is not difficult for an adversary to manipulate the sensors in an unprotected WSN. As a result, the sensors in a WSN are vulnerable to malicious attacks. Thus, the energy efficiency and network security must be considered when designing WSN application [6–9].

Cryptography plays a very important role in security [10–12]. The common public key encryption methods can be classified into three categories: Public Key Infrastructure (PKI) based encryption, identity-based encryption (IBE) and the certificate-free encryption [13]. Among them, the identity-based encryption (IBE) method has obvious advantage in both security and calculation complexity. The literature [14] pointed out that the IBE method is the most suitable public key system for wireless sensor networks (WSN). However, there is a flaw in the key escrow scheme of IBE algorithms. Because the Private Key Generator (PKG) holds private keys of all users, it can easily impersonate any node user, decrypt its ciphertext, and forge user signatures.

In this paper, the Hierarchical Matrix Decomposition-based Signcryption (HMDS) algorithm was proposed to resolve the above problems. In HMDS algorithm, the clustering management scheme is employed to adapt to the data aggregation architecture in WSN. The matrix decomposition-based method is adopted to generate the keys for cluster head nodes, which makes use of identification information and avoids absolute control of KGC on the private keys. Such mechanism really solves the key escrow problem in IBE algorithm. To evaluate the performance of the HMDS scheme, we compare it with the Hierarchical Identity-Based Signcryption Scheme (HIS) algorithm [15]. The experiment results show that the HMDS algorithm is more suitable

for the WSN. Especially in the large-scale WSN, the HMDS algorithm is very stable and has advantage over HIS scheme.

2. Related Works

2.1. Public Key Authentication. In recent years, many researchers have done a lot of work on the key authentication for WSN. There are three kinds of authentication methods, PKI-based encryption, identity-based encryption, and certificate-free encryption.

2.1.1. PKI-Based Encryption. In a typical PKI-based scheme, the public key certificate is signed by a certificate authority (CA) to achieve the binding between the user's public key and its identity. The certificate-based key management scheme is the most common authentication scheme. However, the PKI-based key management scheme tends to consume too much storage, calculation, and communication resource, which is unaffordable for the energy-constrained sensor nodes in WSN [16, 17].

2.1.2. Identity-Based Encryption. In 1984, the identity-based encryption algorithm was first proposed by Shamir [18], which enabled any pair of users to communicate securely and verify each other's signatures without exchanging private or public keys. In 2003, Chen studied the key management protocol in identity-based encryption scheme [19]. In order to simplify the management of the key, the user's public key was used for directly calculating the user's identity information, and the private key was credibly generated by PKG (Private Key Generator). Without the public key certificate, such scheme reduced the storage and computing cost for issuance, cancellation, and certification.

To reduce the energy consumption of the identity-based authentication node, some identity-based key management protocols for WSN were proposed [20, 21]. The literature [11] pointed out that the identity-based encryption was the most suitable public key scheme for wireless sensor networks (WSN). Neal Koblitz [22] and Victor Miller [23] proposed elliptic curve cryptosystem. Then, the similar algorithm was proposed by Zhang et al. in [24], which presented that the security of ECC was based on the discrete logarithm problem. Compared with other public key systems, this scheme is a promising method with high security, low cost, and high efficiency. However, in most elliptic curves based key management schemes, grouping management mechanism was not introduced. With the increasing of nodes, more communication with BS is needed, which was not suitable for the nodes with limited energy in WSN.

In the key-insulated cryptography proposed by Qin Zhiguang et al., the key was divided into two parts: one part was managed by the users and the other part was saved by a physical security helper [12]. When the key was needed, the two parts of the key were spliced into a complete key. But it failed to take advantage of the identity information from each effective node. Chen Yuan et al. [13] proposed an identity-based encryption scheme without bilinear pairings. Both algorithms used the idea of noncertificate; the private key was generated by the node and the Private Key Generator (PKG). Therefore, the two algorithms could not really solve the key escrow problem.

Guo Jianghong et al. [16] proposed a new key agreement scheme for WSN. The node established the pairing key through the Diffie-Hellman protocol, and the required key parameters are obtained through broadcast. The scheme still has the advantages of the identity-based encrypted key agreement scheme, but the time and energy consumption for bilinear pairing operation is very high. Guo et al. [25] combined the identity-based RSA mechanism with the lightweight Certificate Authority (CA) to construct an identity-based mixed model, ECC-CA, for sensor network cryptography scheme. But the scheme has relatively huge cost of calculation and communication.

2.1.3. Certificate-Free Encryption. To resolve the certificate problem in the PKI-based scheme and the key escrow problem in the identity-based scheme, the certificate-free encryption method was proposed. In certificate-free encryption algorithm, a credible third-party Key Generation Center (KGC) was still employed to jointly generate the user's private key. KGC did not directly hold the users' private keys but only generated a partial private key. The users themselves generated the final private key instead. In such scheme, the public key information could not be directly obtained from the user identity information [26, 27]. At present, these provable security certificate-free construction methods are based on the Waters hash function, which leads to long system parameters and requires more pairing calculations. Thus, these methods are not suitable for WSN with limited storage capacity and calculation capability.

2.2. Cluster-Based Authentication for WSNs. Zhang et al. [28] used two cluster heads to realize the system authentication work. They assumed that the cluster head (CH) had limited processing capacity and could not meet the requirements of data calculation. In a cluster, one CH was responsible for intercluster communication and the other CH was responsible for the collection and processing of data in its cluster. Dai et al. [29] also proposed a dual cluster head authentication scheme, whose main principle was similar to the system proposed by Zhang. The main CH and the sub-CH were selected in the larger clusters; the main CH was assisted by the sub-CH. Wang et al. [30] proposed a nonuniform clustering management scheme achieved by establishing clusters with different sizes, which was different from dual cluster head structure. However, it did not consider the fact that the influence of fixed factors on network varies with the number of rounds. Based on the dynamic search strategy, in literature [31], dynamic CH was introduced to collect the interference data of illegal nodes, but it could not realize adaptive switching data collection path. Yu et al. [32] proposed a DEER algorithm to find the optimal path between a CH and the coordinator, but it may lead to unbalanced load among the CHs.

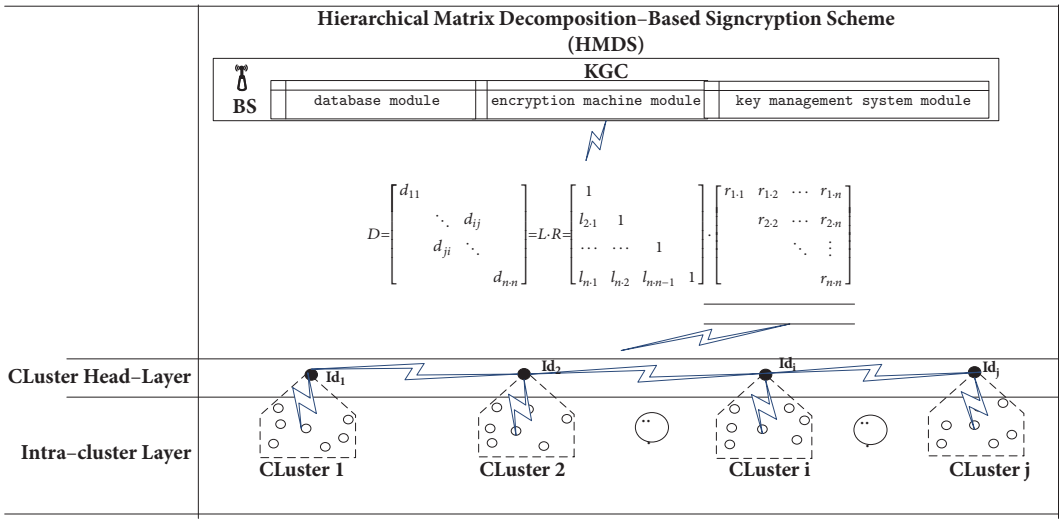


Figure 1: Architecture of HMDS.

Rohbanina et al. [33] proposed a hierarchical WSN for key management, which first constructed the shortest path and uses the elliptic curve-based cryptography scheme for session key distribution. DENG et al. [34] made use of the network differentiation to improve system communication efficiency, which adopted the grouping encryption algorithm. By using the summary information comparison method and the singular point exclusion strategy, the system recoverability was enhanced. Klaoudatou et al. [35] thoroughly evaluated the cluster-based Group Key Agreement (GKA) protocols for WSNs. The authors examined many related literatures to study their performance and energy consumption. They thought that clustering is ideal for large-scale environments and time-critical applications. The use of cluster-based approaches optimizes network bandwidth and service discovery while addressing the needs for scalability at the same time.

3. Hierarchical Matrix Decomposition-Based Signcryption (HMDS) Scheme

3.1. Overview of HMDS Scheme. In the WSN, cluster-based hierarchical architecture could decrease the communication overhead of data collection and aggregation. Moreover, it is good to extend the lifetime if nodes are assigned to different roles according to their resources. Therefore, we propose a Hierarchical Matrix Decomposition-based Signcryption (HMDS) scheme, which adopts the cluster-based hierarchical architecture illustrated as in Figure 1. There are three layers, base station (BS), cluster head layer, and intracluster layer. Usually, the WSN consists of a few clusters and a BS. When a node communicates with other nodes in different cluster, the message must be forwarded by the corresponding CHs.

In the HMDS scheme, the BS is used as KGC. There are three modules for performing the KGC functions, database module, encryption machine module, and key management system module. The BS assigns an identification number and generates the public key for each CH in cluster head layer. Matrix D is stored in the database on BS. Given a symmetric matrix D, it is Doolittle-decomposed into two triangular matrices L and R; thus, the ID of each node can be assigned quickly according to the polynomial function. Furthermore, the authentication and key updating of the node can be realized.

The nodes with more storage and computing resource are assigned as CHs. Each CH is responsible for managing a number of ordinary sensor nodes and performing more complex operations. In the cluster head layer, each CH computes the public key for each sensor node in its cluster.

Before presenting the detail of the HMDS scheme, it is necessary to introduce some deployed notations as described in Table 1.

3.2. Signcryption for Intercluster Communication

3.2.1. Initialization. The BS, as the KGC, is trustable to generate key for CHs. Before generating keys, the steps of initialization are described as follows.

(A) BS selects a (n×n) symmetric matrix D and saves it. Each order principal minor determinant of D is not equal to 0; the matrix D is represented as follows:

$$D = \begin{bmatrix} d_{11} & & \\ & \ddots\ d_{ij} & \\ & d_{ji}\ \ddots & \\ & & d_{n\cdot n} \end{bmatrix} \tag{1}$$

where $d_{ij}=d_{ji}$.

Table 1: Deployed notations.

Notations Guide	
D	(n×n) symmetric matrix
L	The lower triangular matrix of Doolittle-decomposed for matrix D
R	The upper triangular matrix of Doolittle-decomposed for matrix D
S_i/P_i	Secret key/ Public key of applicant ID_i
Id_i	Identification number of cluster head node i
G	The base point on the elliptic curve
U_i	The verification parameter of node i
H	Hash function

(B) As the KGC, the BS performs Doolittle decomposition for the matrix D as follows:

$$
\begin{aligned}
D &= L \cdot R \\
&= \begin{bmatrix} 1 & & & \\ l_{2\cdot1} & 1 & & \\ \cdots & \cdots & 1 & \\ l_{n\cdot1} & l_{n\cdot2} & l_{n\cdot n-1} & 1 \end{bmatrix} \cdot \underline{\begin{bmatrix} r_{1\cdot1} & r_{1\cdot2} & \cdots & r_{1\cdot n} \\ & r_{2\cdot2} & \cdots & r_{2\cdot n} \\ & & \ddots & \vdots \\ & & & r_{n\cdot n} \end{bmatrix}} \\
&= L \cdot R^0 \cdot R^1 \\
&= \begin{bmatrix} 1 & & & \\ l_{2\cdot1} & 1 & & \\ \cdots & \cdots & 1 & \\ l_{n\cdot1} & l_{n\cdot2} & l_{n\cdot n-1} & 1 \end{bmatrix} \\
&\quad \cdot \underline{\left(\begin{bmatrix} r_{1\cdot1} & & & \\ & r_{2\cdot2} & & \\ & & \ddots & \\ & & & r_{n\cdot n} \end{bmatrix} \cdot \begin{bmatrix} 1 & \frac{r_{1\cdot2}}{r_{1\cdot1}} & \cdots & \frac{r_{1\cdot n}}{r_{1\cdot1}} \\ & 1 & \cdots & \frac{r_{2\cdot n}}{r_{2\cdot2}} \\ & & \ddots & \vdots \\ & & & 1 \end{bmatrix} \right)}
\end{aligned} \tag{2}
$$

(C) The KGC saves all the information of the matrix R, and according to the diagonal elements of the matrix R, the following one-way polynomial function is used to generate the identification number Id_i for each CH node i.

$$
\begin{aligned}
Id_i &= f_{(x)} \\
&= \left(r_{11} \cdot x^{(n)} + r_{22} \cdot x^{(n-1)} + \cdots + r_{nn} \cdot x^{(1)} \right) \bmod (p), \\
&\qquad \text{where } x = r_{ii},\ i \in (1, n).
\end{aligned} \tag{3}
$$

(D) KGC broadcasts the matrix L to all the CHs; that is to say, each CH knows all the information of the matrix L.

(E) According to the CH number *Idi* and matrix L, each CH node only stores the corresponding row elements in matrix L and ignores other elements. For example, if Id_i=4, the CH only stores the 4th row elements of the L matrix as follows:

$$
\begin{bmatrix} 1 & & & \\ l_{2\cdot1} & 1 & & \\ \cdots & \cdots & 1 & \\ l_{n\cdot1} & l_{n\cdot2} & l_{n\cdot n-1} & 1 \end{bmatrix} \underset{i=4}{\overset{Id_4}{\Longrightarrow}} \begin{bmatrix} & & & \\ & & & \\ & & & \\ \underline{l_{4\cdot1}} & \underline{l_{4\cdot2}} & \underline{\cdots} & \underline{1} \end{bmatrix}. \tag{4}
$$

3.2.2. User Key Generation (Extract). After initialization, the steps of key generation are described as follows.

(A) The applicant node i, named as CH_i, selects a point $P(x_i, y_i)$ on the elliptic curve as its own private key, that is, $S_i=(x_i, y_i)$, and then randomly selects a number $z_i \in Z$ and calculates the verification shares $V_i=z_iS_i=(V_{xi}, V_{yi})$.

(B) CH_i saves the random number z_i, which will be used in the signcryption process; then it combines the self verification shares V_i with the identity information Id_i and sends it to the KGC. It should be noted that random number z_i will not be sent to KGC, which prevents KGC from leaking secrets, and z_i is only used for the intercluster communication.

(C) After receiving the information, the KGC selects a random number j. According to the value of j together with CH_i's number i, the KGC chooses the element r_{ji} in the matrix R and obtains the verification parameter $V_{Kgc} = (r_{ji}/r_{jj})$ of the KGC.

(D) The KGC calculates the public key $P_i = G \cdot V_i$ for CH_i based on the received verification shares $V_i(V_{xi}, V_{yi})$.

(E) The KGC sends the random number j, the verification parameter V_{KGC}, and the user public key P_i to the applicant CH_i.

(F) After receiving the information returned by the KGC, according to the received random number j, CH_i selects the j_{th} element l_{ij} from the stored i_{th} row vector in matrix L and verifies whether $l_{ij}= V_{KGC}$. If the verification passes, CH_i accepts the public key P_i and saves its own private key S_i. Then, the KGC announces the user's public key P_i to other CHs. Otherwise, it returns an error and the application fails (the correctness will be proven in Section 4.1).

Thus, the configuration of the public and private keys for each CH has been completed.

3.2.3. Signcryption. When CH_i needs to send a message to CH_j, the following steps will be performed.

(A) CH_i first takes the random number z_i which is saved at the public key application stage; together with the base point G on the elliptic curve, CH_i calculates $K_i = z_i \cdot G = (k_x^{Gi}, k_y^{Gi})$. Then, CH_i calculates the verification parameter $U_i = k_x^{Gi} \cdot P_j = (u_x^{ij}, u_y^{ij})$ by k_x^{Gi} and the public key P_j of CH_j. After that, CH_i encrypts the message Msg by using $u_x^{ij}$and obtains $C = E_{u_x^{ij}}(Msg)$.

(B) CH_i calculates the parameters $M_i = \mathrm{H1}(k_x^{Gi} \mid C)$, $N_i = \mathrm{H0}(k_x^{Gi} \cdot M_i)\mathrm{G}$,

$$\begin{aligned} \mathrm{O}' &= \mathrm{S}i - \left(k_x^{Gi} \cdot M_i\right) \bmod \mathrm{q}, \\ \mathrm{O} &= \mathrm{M} + \mathrm{O}' \end{aligned} \tag{5}$$

(C) CH_i generates the ciphertext $\sigma_i = (K(k_x^{Gi}, k_y^{Gi}) \| C = E_{u_x^{ij}}(Msg) \| N_i \| O \| z_i)$ according to the stored random number z_i and all the above parameters.

(D) CH_i sends the ciphertext to CH_j.

3.2.4. Unsigncryption. After CH_j receives the ciphertext, the following steps will be performed to decrypt the received message.

(A) CH_j extracts the first half k_x^{Gi} of K_i and calculates the verification parameters $U_j = k_x^{Gi} \cdot P_j = (u_x^j, u_y^j)$ with its own public key.

(B) CH_j calculates

$$\begin{aligned} M_j &= H_1\left(k_x^{Gi} \| C\right) \\ O'' &= O - M_j. \end{aligned} \tag{6}$$

(C) CH_j decrypts the encrypted message by using u_x^j to obtain $Msg = D_{u_x^j}(C)$.

(D) CH_j calculates $P_j{}' = N_i + O'' \cdot G$ and verifies if $z_i \cdot P_j = P_i$; if they are equal, the common communication key of CH_i and CH_j is u_x^{ij}, and u_y^{ij} is stored as the intercluster shared key after receiving the text Msg; otherwise, the verification fails.

Thus, communication between the clusters finished.

3.3. Signcryption for Intracluster Communication

3.3.1. Key generation and Unsigncryption. In each cluster, CH_i and CH_j obtained the shared key u_y^{ij} between cluster i and cluster j through the above process. The key generation, signcryption, and unsigncryption for the ordinary nodes in a cluster are similar to those for cluster head, as described in Section 3.2, so it is not necessary to repeat them here.

3.3.2. Intracluster Key Updating. In order to ensure security, the BS will periodically inform each CH to perform the intracluster key updating operation. For instance, in the cluster whose cluster head is CH_i, the updating steps are as follows:

(A) CH_i generates a new cluster shared key $(u_y^{ij})'$.

(B) CH_i uses its own private key S_i for identity signature, and the ordinary nodes authenticate it.

(C) The signature and the new cluster shared key $(u_y^{ij})'$ together with the original shared key are encrypted to obtain $C' = E_{u_x^{ij}}(S_n(Id_i)S_i \| Id_i \| Clock \| (u_x^{ij})')$ and the information is broadcasted in this cluster.

(D) After the ordinary nodes receive the broadcasted information, they decrypt the information by original shared key u_y^{ij} and verify it by the public key of the cluster head; if the verification passes, the new cluster shared key $(u_y^{ij})'$ is saved to replace the old one and answers CH; otherwise, discard the data package.

4. Correctness and Security Analysis

4.1. Correctness Analysis

Theorem 1. *In the process of generating public key (described in Section 3.2.2) for CH, if l_{ij}=V_{KGC}, the public key generation satisfies the correctness requirements.*

Proof. Firstly, according to the Doolittle decomposition rules for the symmetric matrix D, combined with the knowledge of Order Principal Minor Determinant, we have

$$\begin{aligned} D = L \cdot R &= \begin{bmatrix} 1 & & & \\ l_{2\cdot1} & 1 & & \\ \cdots & \cdots & 1 & \\ l_{n\cdot1} & l_{n\cdot2} & l_{n\cdot n-1} & 1 \end{bmatrix} \\ &\cdot \begin{bmatrix} r_{1\cdot1} & r_{1\cdot2} & \cdots & r_{1\cdot n} \\ & r_{2\cdot2} & \cdots & r_{2\cdot n} \\ & & \ddots & \vdots \\ & & & r_{n\cdot n} \end{bmatrix} \end{aligned} \tag{7}$$

because $|D| = |L \cdot R| = |L| \cdot |R|, \quad |D| \neq 0, \ |L| = 1$

that is $|D| = 1 \cdot |R| = \prod_{i=1}^{n} r_{\mathrm{ii}} \neq 0 \Longrightarrow$

$r_{\mathrm{ii}} \neq 0 \, (i \in 1, 2, \cdots, \mathrm{n})$

Furthermore, the matrix R can be decomposed as follows:

$$\begin{aligned} R &= \begin{bmatrix} r_{1\cdot1} & r_{1\cdot2} & \cdots & r_{1\cdot n} \\ & r_{2\cdot2} & \cdots & r_{2\cdot n} \\ & & \ddots & \vdots \\ & & & r_{n\cdot n} \end{bmatrix} = R^0 \cdot R^1 \\ &= \begin{bmatrix} r_{1\cdot1} & & & \\ & r_{2\cdot2} & & \\ & & \ddots & \\ & & & r_{n\cdot n} \end{bmatrix} \cdot \begin{bmatrix} 1 & \frac{r_{1\cdot2}}{r_{1\cdot1}} & \cdots & \frac{r_{1\cdot n}}{r_{1\cdot1}} \\ & 1 & \cdots & \frac{r_{2\cdot n}}{r_{2\cdot2}} \\ & & \ddots & \vdots \\ & & & 1 \end{bmatrix} \end{aligned} \tag{8}$$

Table 2: Function y=f(x).

Factor	r_{11}	r_{22}	r_{33}	...	$r_{(n-1)(n-1)}$	r_{nn}
Y/x_i	$Y_1=r_{11}$	$Y_2=x_i \cdot Y_1+r_{22}$	$Y_3=x_i \cdot Y_2+r_{33}$	...	$Y_{n-1}=x_i \cdot Y_{n-2}+r_{(n-1)(n-1)}$	$Y_n=x_i \cdot Y_{n-1}+r_{nn}$

Therefore, we have

$$\underline{D = L \cdot R = L \cdot R^0 \cdot R^1}$$

because $D = D^T,\ D = L \cdot R^0 \cdot R^1$

$$\begin{aligned} \textit{so}\quad D = D^T &= \left(L \cdot R^0 \cdot R^1\right)^T = \left(R^0 \cdot R^1\right)^T \cdot L^T \\ &= \left(R^1\right)^T \cdot \left(R^0\right)^T \cdot L^T \\ &= \left(R^1\right)^T \cdot R^0 \cdot L^T \\ &= L \cdot R^0 \cdot R^1 \end{aligned} \tag{9}$$

Based on uniqueness of the Doolittle decomposition, we have $L = (R^1)^T$; that is,

$$\begin{aligned} L &= \begin{bmatrix} 1 & & & \\ l_{2\cdot1} & 1 & & \\ \cdots & \cdots & 1 & \\ l_{n\cdot1} & l_{n\cdot2} & l_{n\cdot n-1} & 1 \end{bmatrix} = R^T \\ &= \begin{bmatrix} 1 & \frac{r_{1\cdot2}}{r_{1\cdot1}} & \cdots & \frac{r_{1\cdot n}}{r_{1\cdot1}} \\ & 1 & \cdots & \frac{r_{2\cdot n}}{r_{2\cdot2}} \\ & & \ddots & \vdots \\ & & & 1 \end{bmatrix}^T \end{aligned} \tag{10}$$

$$V_{Kgc} = \left(\frac{r_{ji}}{r_{jj}}\right)$$

$$\Downarrow$$

$$\underline{l_{ij} = V_{Kgc}}$$

Therefore, the KGC public key generation process satisfies the correctness requirements. □

Theorem 2. *The recalculation process of the verification parameter P_i satisfies the correctness requirements.*

Proof.

$$\begin{aligned} \textit{since}\quad O' &= S_i - H_0\left(k_x^{Gi} \cdot M_i\right) \bmod q, \\ N_i &= H_0\left(k_x^{Gi} \cdot M_i\right) G, \\ O'' &= O - M_j \\ \textit{then}\quad P_j{}' &= N_i + O'' \cdot G \\ &= H_0\left(k_x^{Gi} \cdot M_i\right) G + \left(O - M_j\right) G \\ &= H_0\left(k_x^{Gi}\right) G + M_i \cdot G + O \cdot G - M_i \cdot G \end{aligned}$$

$$\begin{aligned} \textit{because}\quad P_i &= V_i \cdot G, \\ V_i &= z_i \cdot S_i \\ \textit{so}\quad P_i &= \left(z_i \cdot S_i\right) \cdot G = z_i \left(S_i \cdot G\right) = z_i \cdot P_j{}' \end{aligned} \tag{11}$$

□

4.2. Security Analysis. In the HMDS algorithm, the private key of an applicant is managed by the applicant itself, and the KGC only generates the public key for the applicant. The authentication parameters transferred during the communication process are also based on the public keys of the applicant. Therefore, even if a node is compromised, it will not affect the security of other nodes.

In the HMDS algorithm, one characteristic is that the monomial function is adopted to generate the identification number for an applicant, which ensures the partial public secret not to leak. From the polynomial, $Id_i = f_{(x)} = (r_{11} \cdot x^{(n)} + r_{22} \cdot x^{(n-1)} + \cdots + r_{nn} \cdot x^{(1)}) \bmod (p)$; according to Honer's Rule, we can continuously extract xi as a common factor from the residual polynomial recursively to compute Y_i, as illustrated in Table 2. It can be observed that if x, r_{ii}, and p are known, it is very easy to compute the value of Y_i. Conversely, if Y_i and p are known and we try to get the value of xi, it will cost at least $n^2(\log_2 P)^2$ times of multiplication. When n and p are very large, it is very difficult to obtain the value of x_i,. Therefore, it is ensured that the identification number of each node is quickly generated, but it is difficult to crack. Another characteristic of the HMDS algorithm is that the information kept by each applicant is very simple, which saves the cost of calculation and storage resource.

5. Performance Evaluation

5.1. Computation Complexity Analysis. We compare the HMDS algorithm with the HIS algorithm; the main operations and complexity of the two algorithms are shown in Table 3, which include four steps, system initialization, user key generation (extraction), signcryption, and unsigncryption.

5.2. Simulation Experiments. We set up six classical simulation configurations for HMDS and HIS scheme to represent different scenarios, denoted as S1~S6. To evaluate the efficiency of the HMDS scheme, we take the time cost as the metric to compare the HMDS scheme with the HIS scheme. The experiment parameters setting of the 6 scenarios is as shown in Table 4, where *Num_CH* is the number of clusters and *n* is the number of nodes.

Since the key steps of the HMDS scheme are setup (initialization) stage and key generation (extract) stage, the efficiency of the HMDS scheme depends on the two stages.

Table 3: Complexity of calculation.

Process	HMDS		HIS	
	Operation	cost of calculation	Operation	cost of calculation
Initialization	(*A*) Select D; (*B*) Doolittle decomposition : (C) $r_{ii} \xrightarrow{f(x)} Id_i$; (*D*) $Matrix[L] \longrightarrow Id_i$; (*E*) $Id_i \xleftarrow{Saved} Row_i(L)$	(A) O(1); (B) $n(n+1)/2$ (C) O(n); (D) O(n); (E) O(1)	(A) Select q, Gq, E(G); (B) Select Symmatrix D; (C) Genetate H0, H1; (D) Vandermonde Matris ID; (E) Generate key; (F) Broadcast;	(A) O(1); (B) O(1); (C) O(1); (D) O(n∗n); (E) O(1); (F) O(n);
Extraction	(A) Select Pi; Generate Vi; (*B*) $Id_i \xrightarrow{\text{Vi}\|\text{Idi}} KGC$; (C) Calculate VKgc; (*D*) Calculate Pi; (*E*) $KGC \xrightarrow{j\|V_{\text{kgc}}\|P_i} Id_i$; (*F*) $l_{ij}? = V_{Kgc}$	(A) O(1); (B) O(1); (C) O(1); (D) O(1); (E) O(1); (F) O(n);	(*A*) Select Pi; Generate Vi; (*B*) $Id_i \xrightarrow{\text{ViIdi}} KGC$; (*C*) $D \xrightarrow{y=f(x) \bmod (p)} D'$; (D) Caculate (D′ ∗ ID) : (E) $KGC \xrightarrow{K_{ij}\|B_1\|P_i} Id_i$; (F) Caculate (B ∗ ID) :	(A) O(1); (B) O(1); (C) O(n∗n); (D) O(n∗n); (E) O(n); (F) O(n∗n);
Signcryption	(*A*) Generate Ki, Ui, C; (*B*) $\begin{cases} M_i = H_1(k_x^{Gi} \| C) \\ N_i = H_0(k_x^{Gi} \cdot M_i)G \\ O' = S_i - H_0(k_x^{Gi} \cdot M_i) \bmod q \\ O = M + O' \end{cases}$ (*C*) $Id_i \xrightarrow{\sigma_i=(K(k_x^{Gi}, k_y^{Gi}))\|C} Id_j$;	(A) O(1); (B) O(1); (C) O(1);	(A) IDi : Generate K, U, C = Eu(m); (B) Calculate M, N; (C) Calculate O′, O = M + O′; (*D*) $Id_i \xrightarrow{\sigma=(K=(k1,k2),C,N,O,z)} Id_j$;	(A) O(1); (B) O(1); (C) O(1); (D) O(1);
Unsigncryption	(*A*) Calculate Uj; (*B*) $\begin{cases} M_j = H_1(k_x^{Gi} \| C) \\ O'' = O - M_j \end{cases}$ (*C*) $Msg = D_{u_x^j}(C)$ (*D*) $\begin{cases} P_j{}' = N_i + O'' \cdot G \\ z_i \cdot P_j{}'? \equiv P_i \end{cases}$	(A) O(1); (B) O(1); (C) O(1); (D) O(1);	(A) Calculate U′ = $k_1 P_j$ = (u, v); (B) M = H1(k1‖C), O′O − M; (C) m = $D_u(C)$; (*D*) $\begin{cases} P_j{}' = N_i + O' \cdot G \\ z_i \cdot P'? \equiv P_i \end{cases}$	(A) O(1); (B) O(1); (C) O(1); (D) O(1);
Total	O(HMDS)=n∗n		O(HIS)=6n∗n	

Table 4: Experimental setting.

OS	Cpu	HZ	RAM	Developing environment	Library files	Initial energy of ch (Joule)	Energy range	Number of cluster layers
Win10 home	i5,x64	1.9Ghz	8G	Visual studio 2010	<time.h>clock()	20	1.5-3J (random)	2
		S1:n=25,Num_CH=2;		S2:n=60,Num_CH=3;		S3:n=100,Num_CH=4;		
		S4:n=300,Num_CH=10;		S5:n=600,Num_CH=20;		S6:n=1000,Num_CH=35;		

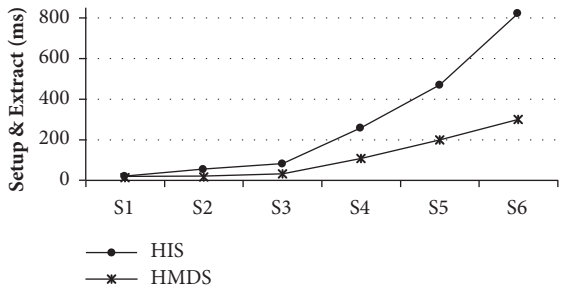


Figure 2: Time consumption of the two stages.

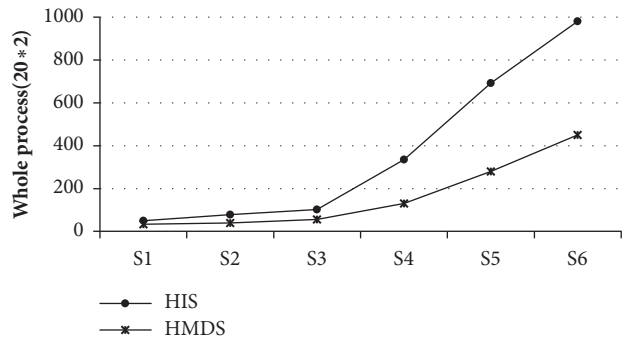


Figure 3: Overall time consumption.

Therefore, besides evaluating the overall efficiency of the HMDS scheme, we specially evaluate the efficiency of the two stages, respectively.

5.2.1. Efficiency of Initialization and Extract Stage. As shown in Figure 2, it can be observed that the time consumption of the two algorithms in the three scenarios, S1, S2, and S3, is all within 100ms and the HMDS algorithm has a slight advantage over the HIS algorithm. It is because the number of the nodes is small. With the number of nodes increasing from 100 (in S3) to 1000 (in S6), the time consumption of the two algorithms also has a large increment, and the HMDS algorithm has an obvious advantage over the HIS algorithm. Especially in the scenario S6, the HIS algorithm takes about 850ms, while the HMDS algorithm only takes 300ms, which saves about two-thirds of the time. In general, for the HIS algorithm, the time consumption of the two stages increases steeply when the network size n increases quickly. In comparison, the HMDS algorithm is relatively stable in time consuming when the network size n is large. Therefore, the HMDS algorithm is more applicable to the WSN than the HIS algorithm. Especially when the network size n is very large, the HMDS algorithm has more obvious advantages.

5.2.2. Overall Efficiency. We randomly take 20 pairs of nodes to evaluate the overall efficiency; that is to say, we measure the time consumption of the four stages, from initialization to the end of unsigncryption. As shown in Figure 3, the overall time consumption for both algorithms increases with the number of nodes increasing, but the HMDS algorithm still takes less time than the HIS algorithm. This trend becomes more and more obvious with the increasing of the network size and is consistent with the results of the above two-stage experiments.

Therefore, the HMDS algorithm outperforms the HIS algorithm. Furthermore, the HMDS algorithm does not cause rapid increase of time consumption when the network scale grows. So the HMDS algorithm is more suitable for larger-scale WSN.

6. Conclusion

In this paper, the identity-based HMDS authentication algorithm was proposed. The public key of the applicant is generated by KGC and the private key is computed by the node itself to solve the key escrow problem, which is a disadvantage of the common identity-based authentication algorithm. The HMDS algorithm adopts hierarchical structure to adapt to WSN requirement. By employing matrix decomposition, the cost of calculation and storage is reduced. Through analysis and experimental comparison, the HMDS algorithm can ensure communication security in the WSN environment and has characteristics of low energy consumption and high stability, which is very suitable for WSN. Furthermore, when the network scale is large, the performance of the HMDS algorithm is very stable and has advantages over HIS scheme. In the future, we will improve the existing matrix decomposition-based HDMS algorithm, especially to optimize the authentication method between cluster heads. Meanwhile, we will study the key updating mechanism to improve the authentication efficiency.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This paper is partly supported by National Natural Science Foundation of China (Grant no. 11671400).

References

[1] M. Guo, Y. Liu, H. Yu, B. Hu, and Z. Sang, "An overview of smart city in China," *China Communications*, vol. 13, no. 5, pp. 203–211, 2016.

[2] H.-D. Ma, "Internet of things: objectives and scientific challenges," *Journal of Computer Science and Technology*, vol. 26, no. 6, pp. 919–924, 2011.

[3] W. Chen, X. Jiang, Z. Tang et al., "Context-based global multi-class semantic image segmentation by wireless multimedia sensor networks," *Artificial Intelligence Review*, vol. 43, no. 4, pp. 579–591, 2015.

[4] H. Y, W. L, and P. Z, "Intelligent wireless sensor network system," *Science Publishing Company*, 978-7-03-037323 -6, 2013.

[5] K. T. Tran and S.-H. Oh, "UWSNs: A Round-Based Clustering Scheme for Data Redundancy Resolve," *International Journal of Distributed Sensor Networks*, vol. 2014, Article ID 383912, pp. 19–26, 2014.

[6] Y. Z. Song and L. W. Li, "Dynamic triggering and cooperation based lifespan optimization of WSN nodes," *Control Engineering of China*, vol. 20, no. 2, pp. 339–343, 2013.

[7] H. J. Cheng, X. B. Huang, and N. Xiong, "Minimum-Energy broadcast algorithm for wireless sensor networks with unreliable communi-cations," *Journal of Software*, vol. 25, no. 5, pp. 1101–1112, 2014 (Chinese).

[8] P. Khandare and N. P. Kulkarni, "Public key encryption and 2Ack based approach to defend wormhole attack," *International Journal of Computer Trends & Technology*, vol. 4, no. 3, pp. 421–428.

[9] Y. L. Zhang, D. R. Zhou, and C. Y. Li, "Certificateless-based efficient aggregate signature scheme with universal designated verifier," *Journal of Communications*, vol. 36, no. 2, pp. 1–5, 2015.

[10] http://en.wikipedia.org/wiki/RSA_(algorithm).

[11] J. Huang and B. Huang, "Public key based key distribution scheme for wireless sensor networks," *Tongxin Xuebao/Journal on Communication*, vol. 32, no. 10, pp. 52–58, 2011.

[12] Z. G. Qin, J. J. Liu, Y. Zhao et al., "A survey of key-insulated cryptography," *Chinese Journal of Computers. Jisuanji Xuebao*, vol. 38, no. 4, pp. 759–774, 2015.

[13] X. Chen, S. Y. Liu, and Y. Wang, "Emergency resources scheduling based on improved backtracking search optimization algorithm," *Computer Applications and Software*, vol. 32, no. 12, pp. 235–238, 2015.

[14] L. B. Oliveira, R. Dahab, J. Lopez, F. Daguano, and A. A. Loureiro, "Identity-Based Encryption for Sensor Networks," in *Proceedings of the Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PerComW'07)*, pp. 290–294, White Plains, NY, USA, March 2007.

[15] C. Yuan, W. Chen, and D. Li, "A Hierarchical Identity-Based Signcryption Scheme in Underwater Wireless Sensor Network," in *11th China Conference on Wireless Sensor Network (CWSN 2017)*, 2017.

[16] J. Guo, X. Li, and J. Wu, "New Identity-based key agreement scheme for WSN," *Computer Science*, vol. 38, no. 3, pp. 127–130, 2011.

[17] J. Katz and M. Yung, "Scalable protocols for authenticated group key exchange," *Journal of Cryptology*, vol. 20, no. 1, pp. 85–113, 2007.

[18] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proceedings of CRYPTO 84 on Advances in cryptology*, vol. 196, LNCS 196, pp. 47-53. New York, NY, USA, Springer-Verlag New York, Inc., 1985.

[19] L. Chen and C. Kudla, "Identity based authenticated key agreement protocols from pairings," in *Proceedings of the 16th IEEE Computer Security Foundations Workshop, CSFW 2003*, pp. 219–233, USA, July 2003.

[20] J.-M. Bohli, B. Glas, and R. Steinwandt, "Towards provably secure group key agreement building on group theory," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 4341, pp. 322–336, 2006.

[21] F. Zhang, S. Liu, and K. Kim, "ID-Based One Round Authenticated Tripartite Key Agreement Protocol with Pairings," *Cryptology ePrint Archive*, Report 2002/122, 2002.

[22] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, no. 177, pp. 203–209, 1987.

[23] V. S. Miller, "Use of Elliptic Curves in Cryptography," in *Lecture Notes in Computer Science*, vol. 218, pp. 417–426, 1 edition, 1985.

[24] W.-F. Zhang, X.-M. Wang, W. Guo, and D.-K. He, "An efficient inter-enterprise authentication scheme for VE based on the elliptic curve cryptosystem," *Tien Tzu Hsueh Pao/Acta Electronica Sinica*, vol. 42, no. 6, pp. 1095–1102, 2014.

[25] P. Guo, H. Zhang, D. Fu, and M. Zhou, "Hybrid and Lightweight Cryptography for WSN," *Computer Science*, vol. 39, 2012.

[26] D. W. Zhou, G. H. Wei, and H. G. Zhang, "Key-management scheme based on public-key institution to clustered wireless sensor networks," *Journal of Beijing University of Technology. Beijing Gongye Daxue Xuebao*, vol. 42, no. 5, pp. 707–712, 2016.

[27] G. Lippold, C. Boyd, and J. Gonzalez Nieto, "Strongly secure certificateless key agreement," in *Pairing-based cryptography—Pairing 2009*, vol. 5671 of *Lecture Notes in Comput. Sci.*, pp. 206–230, Springer, Berlin, 2009.

[28] R. S. Zhang and K. S. Qu, "Energy hole alleviation of WSNs based on dual cluster head grid scheduling," *Transducer and Micro-system Technologies*, vol. 33, no. 10, pp. 133–136, 2014.

[29] Z. Dai, C. Yan, and Z. Wu, "New uneven double cluster head clustering algorithm for WSN-PUDCH algorithm," *Chinese Journal of Sensors and Actuators*, vol. 29, no. 12, pp. 1912–1918, 2016.

[30] L. Wang, "Improved algorithm of non-uniform clustering routing protocol[J]," *Computer Science*, vol. 44, no. 2, pp. 152–156, 2017.

[31] X. Liu, "A novel transmission range adjustment strategy for energy hole avoiding in wireless sensor networks," *Journal of Network and Computer Applications*, vol. 67, pp. 43–52, 2016.

[32] M. Simões, L. C. Simões, S. Cleto, I. Machado, M. O. Pereira, and M. J. Vieira, "Antimicrobial mechanisms of *ortho*-phthalaldehyde action," *Journal of Basic Microbiology*, vol. 47, no. 3, pp. 230–242, 2007.

[33] R. Linsker, "Self-organization in a perceptual network," *The Computer Journal*, vol. 21, no. 3, pp. 105–117, 1988.

[34] S. Deng and Y. Wang, *S. Deng*, vol. 39, Chinese, Computer Engineering, 2013.

[35] E. Klaoudatou, E. Konstantinou, G. Kambourakis, and S. Gritzalis, "A survey on cluster-based group key agreement protocols for WSNs," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 3, pp. 429–442, 2011.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 1032761, 15 pages
https://doi.org/10.1155/2018/1032761

Review Article

Towards Privacy Preserving IoT Environments: A Survey

Mohamed Seliem, Khalid Elgazzar, and Kasem Khalil

Centre for Advanced Computer Studies (CACS), University of Louisiana at Lafayette, LA 70503, USA

Correspondence should be addressed to Mohamed Seliem; mohamed.seliem1@louisiana.edu

Received 9 July 2018; Accepted 30 October 2018; Published 18 November 2018

Guest Editor: Constantinos Kolias

The Internet of Things (IoT) is a network of Internet-enabled devices that can sense, communicate, and react to changes in their environment. Billions of these computing devices are connected to the Internet to exchange data between themselves and/or their infrastructure. IoT promises to enable a plethora of smart services in almost every aspect of our daily interactions and improve the overall quality of life. However, with the increasing wide adoption of IoT, come significant privacy concerns to lose control of how our data is collected and shared with others. As such, privacy is a core requirement in any IoT ecosystem and is a major concern that inhibits its widespread user adoption. The ultimate source of user discomfort is the lack of control over personal raw data that is directly streamed from sensors to the outside world. In this survey, we review existing research and proposed solutions to rising privacy concerns from a multipoint of view to identify the risks and mitigations. First, we provide an evaluation of privacy issues and concerns in IoT systems due to resource constraints. Second, we describe the proposed IoT solutions that embrace a variety of privacy concerns such as identification, tracking, monitoring, and profiling. Lastly, we discuss the mechanisms and architectures for protecting IoT data in case of mobility at the device layer, infrastructure/platform layer, and application layer.

1. Introduction

The Internet of Things (IoT) is a group of connected physical devices that exchange data about themselves and their environments and may take actions on it. The 2017 report of the International Data Corporation [1] forecasts that 50 billion devices will be connected by 2020 with a $8.9 trillion market value. Gartner [2] published a similar study expecting that, in 2020, 50 Billion devices will have their own unique Identifier with a $19 trillion market share opportunity. Not only are IoT devices equipped with a varying level of computational power (e.g., microcontrollers) but also many run a full stack operating system (e.g., Contiki [3] and RIOT [4]) that enables these devices to perform high-level functionality.

The main strength of IoT is the huge impact it will have on several aspects of the user's everyday interactions and the surrounding environment (e.g., smart spaces). This will improve our quality of life in different domains [5] as shown in Figure 1 including Energy [6], Safety, Security, Industry [7], Environment, Entertainment, and Healthcare [8]. However, IoT devices are intrinsically resource-constrained in terms of computation, battery power, intermittent connectivity, and network protocols.

These constraints directly impact the choice of technology applicable to maintain user privacy. Hence, promoting IoT adoption from the user perspective and mitigating potential risks of data misuse and security concerns. IoT devices do not commonly implement a standard security scheme [3, 4], which means there is a huge risk to connect this large number of unsecured devices to the Internet [9]. Proofpoint Inc. has uncovered the first proven cyberattack based on IoT household smart appliances between December 23, 2013, and January 6, 2014. This attack involved more than 750,000 malicious email communications coming from more than 100,000 everyday consumer gadgets such as home routers, smart TVs, and smart refrigerators [10]. In 2015, two ethical hackers, Charlie Miller and Chris Valasek, gained control of a Jeep Cherokee remotely through vulnerability in its onboard entertainment system. According to the hackers, they were able to break the Uconnect system that the Chrysler's line-up of cars and trucks use due to open vulnerability. Hundreds of thousands of vehicles could be affected. In October 2016,

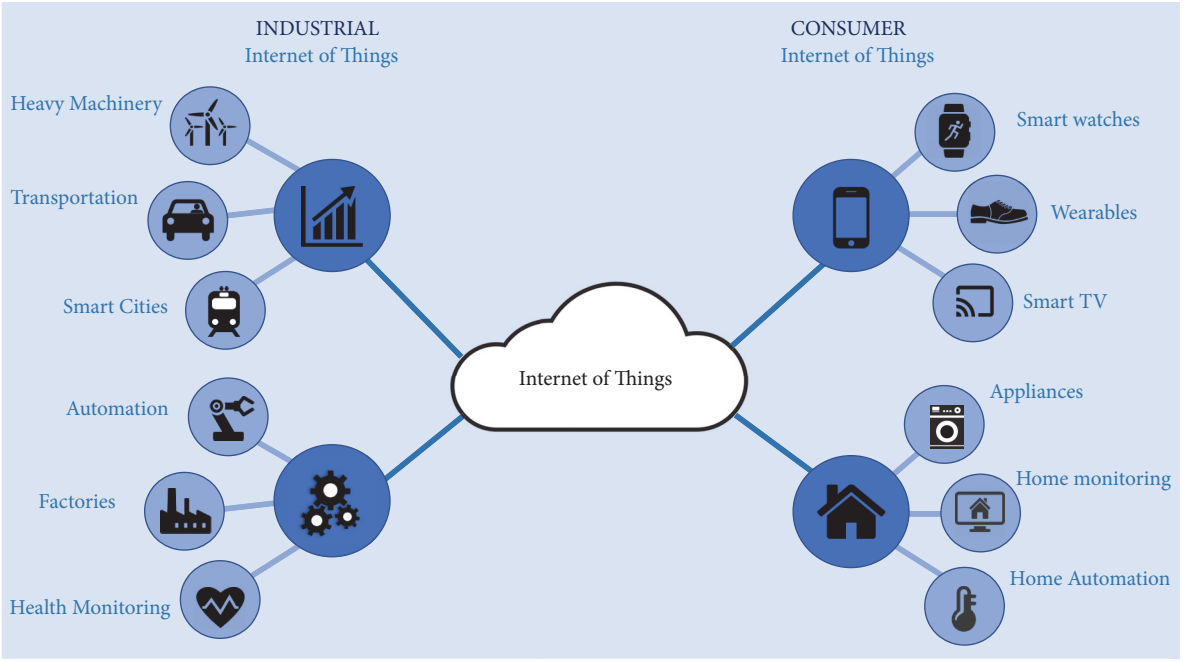


Figure 1: Internet of Things applications.

a major DDoS attack took down several giant servers such as Airbnb, Reddit, Etsy, Sound Cloud, New York Times, Amazon, Twitter, and Spotify. Though it is not clear yet that the blame is on IoT, it is quite likely.

IoT devices are considered not only a security threat, but also a major privacy concern, as these devices collect much personal data such as user identity, location, energy consumption, and telephone numbers. This information can reveal a lot about the user's daily life activities (e.g., using washing machines, watching TV, and leaving or returning home). The major concern yet is that these devices not only can collect users' private data but also can control their environments. Thus, users are highly uncomfortable exposing personal data to public or private servers without a well-established trust model [11]. Therefore, the lack of well-designed IoT-oriented privacy and security techniques will inhibit the user adoption to any IoT technology [12].

The following scenario embodies much of the semantic meaning of user privacy leakage and associated risks. Ahmed looks at his smartwatch while doing his usual workout and sees that his heart rate is a little high. Once he arrived home, he speaks to the smart speaker installed in his room to provide him a list of nearby cardiologists to check his heart and blood vessels. Next day after finishing his work, Ahmed visited a cardiologist, and he felt relieved when his doctor reassured everything is fine. "You just did more exercises than usual and nothing to worry about", the doctor said. The next day, every time Ahmed uses his browser he finds too many advertisements related to heart medication, heart monitoring devices, and many tutorials about diagnosing a heart attack and how to handle it. Things got more serious when he received a phone call from his insurance company for check-up survey. "Goodbye privacy", he whispers to himself as he speaks with the company representative. This is only one scenario of too many that shows high privacy risks associated with modern technology usage that has become an inevitable part of our everyday life.

There exist many published surveys on IoT privacy and security issues, challenges, and solutions. Ziegeldorf et al. [13] analyze the privacy issues in IoT. Their focus is on classifying the various privacy threats and pointing out the challenges in IoT scenarios. Sadeghi et al. [14] introduce the security and privacy challenges of industrial IoT systems. They also discuss possible solutions towards a complete and secure framework for industrial IoT. Sicari et al. [15] focus on the main security challenges and the current solutions. They categorize the issues into authentication, access control, confidentiality, privacy, trust, secure middleware, mobile security, and policy enforcement. Suo et al. [16] review the security and privacy in IoT, where they analyze the security architecture and features. They also discuss ongoing research status and the challenges of secure technologies including encryption mechanism, communication security, protecting sensor data, and cryptographic algorithms to support privacy preservation in IoT. Although security and privacy are highly correlated, this paper primarily focuses on privacy challenges and discusses a broad range of privacy-related aspects in open IoT environments to provide better insights on the design principles and development of privacy preserving IoT environments.

The remainder of this paper is organized as follows. Section 2 summarizes the unique characteristics of IoT posing significant challenges on resource-constrained IoT devices. We also discuss the notion of privacy in IoT by pointing out some scenarios related to privacy concerns. Section 3 describes major IoT privacy issues and concerns

such as identification, tracking, monitoring, and profiling. Then, we discuss existing IoT privacy solutions in Section 5. Section 6 provides a comprehensive analysis of privacy issues and mechanisms at the different layers of the IoT stack, namely, Device Layer, Platform Layer, and application layer, respectively. Lastly, Section 7 concludes the paper and provides closing notes.

2. Privacy in the Internet of Things

IoT provides consumers with a high degree of automation and control on how to carry out everyday tasks through saturating the environment with smart things. IoT smart things refer to a broad spectrum of nonstandard computing devices including microcontrollers, sensors, and actuators that can transmit and exchange data to enable smarter interactions and support informed decision making. Things are embedded in consumers' devices and industrial machinery to collect and exchange data about the surroundings. Things can also cause physical changes to their environments and can be controlled directly from proximity or remotely via Internet. Although privacy concerns discussed in this paper are generally valid for any IoT deployments, we provide examples related to consumers' connected devices such as smart appliances to make the point clear. In addition, the pace at which consumers' smart devices are developed is much higher than the development of safeguard techniques that can protect these devices and their data collection from growing privacy threats. Given the high penetration of IoT devices and their impact on our everyday life, we need to fully understand the risks and challenges such devices pose on our privacy. More importantly, we need to answer the following question: Is it possible to support privacy preserving and safe environment for IoT users such those offered in traditional Internet? To precisely answer this question, we need first to discriminate between relevant terminologies such as privacy, trust, and security.

Privacy means that information about individuals must be protected and should not be exposed without explicit consent under any circumstances. Every individual has the ultimate right to decide with whom to share their data. For example, in our previous scenario, Ahmed should be the only one to decide whether to share his heart conditions with the insurance company or not.

Trust is defined as the product of attack probability and the damage it can cause. Trust is derived from two crucial terms: transparency and consistency. Transparency means that IoT devices collecting information inform the user about what data is collected, the purpose of collection, and how the collected data will be used. Consistency means that the behaviour of IoT devices consistently meets user expectation. For example, if a user asks his smart speaker to control the room light, it must do nothing unintended, but the specific requested task. Security refers to the protection of devices and connection from unauthorized access.

Based on the aforementioned definitions, it is evident that privacy is more general than both security and trust. For instance, an IoT service could gain user trust and provide proper security but still violates the user's privacy by exposing personal data without clear and explicit permissions.

Table 1: List of top countries with vulnerable IoT devices.

Country	# of vulnerable devices
United States	57,598
China	17,455
Germany	17,273
France	10,708
India	9,427
United Kingdom	9,268
Russian Federation	7,897
Korea	7,525
Brazil	7,095
Japan	5,302

2.1. Impact of Device Limitations on Privacy. IoT smart things are typically resource-constrained with limited capabilities due to size and weight (e.g., memory, processing, and battery power) and network connectivity (e.g., IEEE 802.15.4). For example, IEEE 802.15.4 specification is constrained with respect to (1) low data rates, which range from 20 Kbits/s (868 MHz) to 250 Kbits/s (2.45 GHz), (2) unreliable and lossy links compared to wired links, (3) small packet size (127 bytes), which means less room for payload when including other headers, and (4) aggressive power cycle, by which IoT smart things aim to save power by staying longer in low-power mode. Such constraints directly impact the type and complexity of functionality that IoT devices can run.

Recently, significant efforts have been made towards standardizing the IoT protocol stack [17]. For example, to enable low-power connectivity among smart objects in IoT systems, the IEEE 802.15.4–2006 low-power physical (PHY) layer and the IEEE 802.15.4e link layer based on Time Synchronized Channel Hopping (TSCH) have been developed in 2006 [18]. Several scholars pointed out the technical challenges that face IoT environments due to various resource constraints.

Yu et al. [19] consider IoT devices as weak access points to vital infrastructures (e.g., a medical or military facility) and can be misused to leak sensitive data. The authors have made two main observations regarding IoT systems: (1) network-based approaches are less vulnerable than host-based approaches due to inherent limitations and possible unpatched vulnerabilities on IoT devices; (2) traditional static perimeter defenses are unable to secure IoT devices, since these devices are deployed deep inside the network, with their physical and computational context constantly changing. Therefore, resource limitations make it challenging to secure IoT layers individually. Table 1 lists the top countries, in Sept 2017, with IoT devices (237,539 devices) vulnerable to Heartbleed [20] according to SHODAN [21] and other sources.

2.2. Impact of Complex Heterogeneity on IoT Privacy. IoT has intrinsic complexity, since multiple diverse objects located

in different contexts can exchange information among each other. This complicates the design and deployment of efficient, interoperable, and scalable mechanisms to preserve users' privacy. Heterogeneity also has significant influence on the design of IoT protocols. Recourse-constrained devices will interact with one another/infrastructure (e.g., web servers and cloudlets) either directly or through gateways. In this case, it is essential to implement or create lightweight security protocols that support an end-to-end secure communication channel. These protocols require implementing distributing management system to distribute credentials and facilitate keys session establishment between peers.

The data flooding caused by billions of IoT devices is a big threat, in a way that hurts and violates the user's privacy. One of the main violations of the large volume of data exchange is linking this data to a certain user. So, the user's anonymity is another dimension that must be taken into consideration to support privacy in connected environments. Further, creating mechanisms to provide data summarization and access policies related to private data will enable transparency and avoid IoT silently taking control of our lives.

There exist several surveys focusing on the impact of IoT heterogeneity on users' privacy. Heer, T. et al. [22] provide a number of requirements to secure IoT environments and preserve user privacy through overcoming specific technical limitations including (1) complex heterogeneity of IoT systems, which complicates protocol design and system operation; (2) scarce CPU and memory resources, which limit the use of resource-demanding cryptoprimitives, such as public-key cryptography as used in most Internet security standards; and (3) end-to-end security measures that are IoT-oriented, since traditional Internet-based approaches are typically inapplicable due to resource limitations. The lessons we learned so far suggests that resource limitations are a major inhibitor to the adoption of traditional techniques as is. IoT architectures must implement privacy by design from the ground up [23], to provide users with central control over their security and privacy.

3. Motivating Scenarios

The vulnerabilities of IoT devices can lead to huge security breaks and threaten user privacy by exposing vital personal information. In this section, we provide some scenarios of IoT applications where personal data can be breached.

3.1. Smart Home Utility Monitoring. According to a recent report published by US Energy Information Administration (EIA) in 2017 [24], 64.7 million smart meters have been installed in electric utilities in 2015. Around 88% of these installations were residential upgrades as shown in Table 2. These smart meters measure and log electricity usage at a minimum of 1-hour intervals but can report readings in real-time at a much higher sampling rate. This data is collected by utility companies at least once a day. Smart meters range from basic hourly interval meters to real-time meters equipped with two-way communication that is capable of recording and transmitting instantaneous data. Natural gas meters and water meters are also on the rise and likely count to similar numbers. However, EIA does not publish their data publicly. The extensive deployment of smart meters has serious privacy implications since they unintentionally leak detailed information about residents' activities. Processing and analyzing this information can lead to serious privacy leakage such as profiling the behaviour of the residents. Furthermore, with existing sophisticated analytics such data may reveal when residents are home and what may be their heath conditions (e.g., toilet flushing rate may indicate that the resident has diarrhoea). This does not stop at breaking the user privacy but may very much lead to life-threating situations.

Table 2: Number of AMI installations by sector, 2015.

Sector	# of smart meters
Residential	57,107,785
Commercial	7,324,345
Industrial	310,889
Transportation	813

Molina-Markham et al. [25] raise awareness around privacy issues related to the use of smart meters. They show that even without knowing any information about household activities, applying off-the-shelf statistical methods can easily extract complex usage patterns from smart meter data. Their work is based on data collected over a 2-month period from three homes. The data contains household power consumption aggregated at 1-second rate. Even with this small-scale deployment, the authors managed to demonstrate the latent for power consumption patterns to reveal a range of personal information, such as how many people are in the house, sleeping routines, and eating routines. The proposed privacy-enhancing smart meter is based on 3 components: household smart meters, neighbourhood gateways, and a remote utility server. The server applies Zero-Knowledge (ZK) protocols [26] that allow a prover (smart meter) to demonstrate the knowledge of a secret (collected data) to a verifier (gateway/server), without revealing any information that help the verifier to infer the secret. This enables utility companies to accomplish their goals without compromising the customer's privacy.

Apthorpe et al. [27] discuss the same concern. However, they point out that an Internet Service provider (ISP) or other network observers can gather private and sensitive information about home activities by analyzing Internet traffic from smart homes containing IoT devices even if the devices use secure encryption. Several strategies have been investigated to avoid the privacy risks associated with smart home traffic monitoring such as traffic blocking, tunnelling, and rate-shaping. However, the user cannot block outgoing traffic from their home; otherwise their devices will be unusable. While traffic tunnelling via a VPN is more secure, it does not totally guarantee privacy preserving. Authors propose the use of traffic shaping using Independent Link

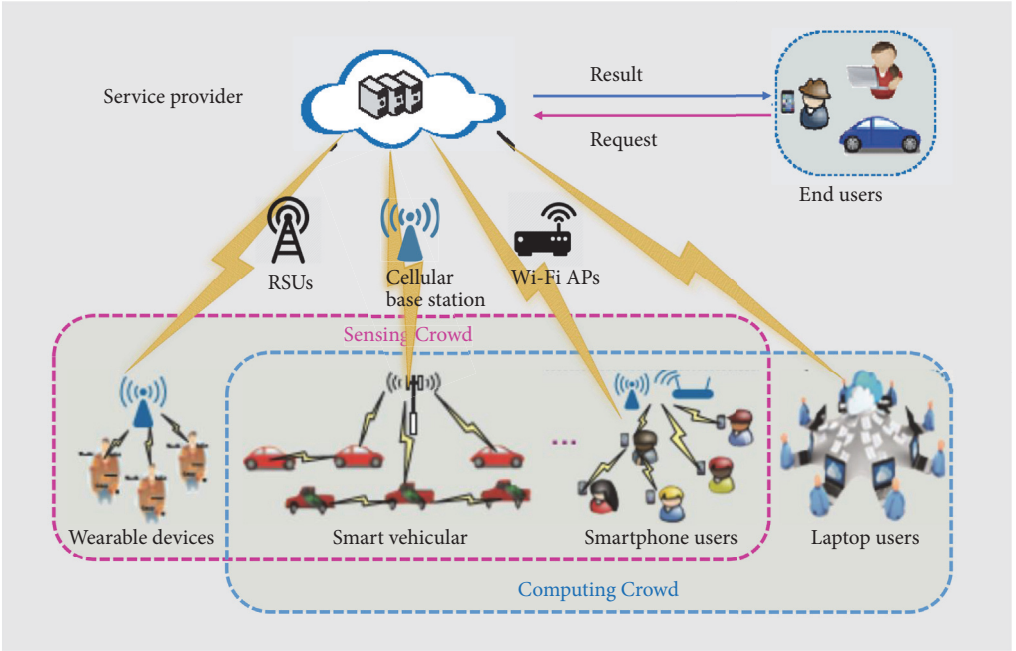


Figure 2: General architecture of mobile crowdsourcing networks (reproduced from [28]).

Padding (ILP) to prevent the leak of rate information and thus render attacks impossible.

3.2. Crowdsourcing and Public Monitoring. Crowdsourcing is the use of collective knowledge of a large crowd to help solving a specific problem. Crowdsourcing benefits from high penetration of smartphones, which becomes the common mobile platform for users worldwide.

Developers and average users with limited programming experience can create feature-rich/personalized applications by simply requesting access to private information such as location, contact lists, media files, etc. However, this model raises serious privacy concerns with no trust mechanisms that govern how applications access and handle such private information. It is also technically possible that applications may gain unauthorized access to the device's camera, networks, and system settings. Mobile Crowdsourcing Networks (MCN) include four basic types of entities as depicted in Figure 2:

(i) The service provider is a crowdsourcing platform that offers crowdsourcing services to both end users and public crowds.

(ii) End users are the clients who purchase or rent crowdsourcing services at a certain cost.

(iii) Sensing crowd is a crowd of mobile users who accept and participate in crowdsourced sensing tasks.

(iv) Computing crowd is a crowd of users who accept and participate in crowdsourced computing tasks (sensing-based computing tasks, or pure computing tasks).

Yang et al. [28] point out that the privacy may be leaked out from either data or tasks. Privacy threats resulting from data leakage can be divided into three categories. (1) privacy of sensed data contains personal information about participants, such as identities, location information, and biometric information. For example, the location information can be easily obtained either from GPS receivers embedded in mobile devices or triangulation-based approaches on Wi-Fi or cellular networks. Moreover, environmental context such as ambient temperature, light, noise level may also reveal the location information. The disclosure of the location information may leak the privacy of participants, such as home and workplace locations, routines, and habits. (2) Privacy of crowdsourced data may contain sensitive information, such as business and financial records, proprietary research data, or personal health information. (3) Privacy of crowdsourcing results can be analyzed to infer sensitive information not authorized/known by the service provider.

4. Evaluation of Privacy Threats in IoT

IoT increasingly evolve with new emerging technologies and services. In this section, we discuss the various privacy threats and challenges associated with IoT environments. We provide a solid definition and a concrete example of privacy violation for each threat. Then, we discuss the impact of IoT evolution on this threat. Lastly, we point out the main challenges associated with these threats as well as any correlation when exists.

4.1. User Identification. From privacy perspectives, user identification is the ability to distinguish a person (or an entity) or revealing their identity based on a piece of acquired data

(e.g., name, address, or personal information). The risk of such a threat is that privacy-violating actions could be carried out after the customer is identified. This threat enables and aggravates other threats, e.g., profiling and tracking of individuals' behaviour. It also allows for linking information from different sources for the same identified target. Analyzing this information can easily result in exposing the target's life pattern. For example, developers that have access to user traces can utilize machine learning techniques to infer personal information about users' interests, which can be exploited to flood user interfaces with ad-ware and targeted advertisements.

The wide adoption of IoT facilitates the collection of a huge amount of data using IoT devices that can be stored and analyzed beyond the user's control domain. Thus, user identification becomes the dominant threat regarding user's privacy. Further, with the increasing number of IoT deployments, the user identification threat and associated risks will significantly scale up.

Different IoT technologies bring their own benefits and challenges. Radio Frequency Identifier (RFID) is commonly used in IoT scenarios to recognize/identify things, record metadata, and control distinct targets through radio waves [29]. The basic RFID system architecture contains tags and readers [30]. Tags are associated with objects for identification and readers read these tags using a close proximity communication technology. When connected to the Internet, RFID remote readers can automatically recognize, track, and monitor any object with a global tag, and in real time if needed [31]. Pateriya et al. [32] point out that vulnerable tags are subject to spying, spoofing, traffic analysis, and denial of service attacks. Unauthorized reader can access these vulnerable tags without proper access privileges. Although the tag information could potentially be protected using lightweight security mechanisms, tracking is easy to accomplish through tag replies.

User identification is currently implemented in almost all mobile platforms, using a variety of mechanisms including face recognition, fingerprints, and/or voice recognition. Surveillance systems also implement face and voice recognition using embedded cameras and microphones, respectively. An unauthorized attacker could gain full access to a surveillance camera and tamper its firmware to send the data to the legitimate server and copy the attacker. Identifying customers by authorized controllers also remains a privacy threat that leads to other threats such as profiling and utility monitoring. Surveillance cameras deployed in public settings (e.g., for video analytics and customer profiling [33, 34]) utilize public facial databases (e.g., MIT-CBCL Face Recognition Database) to track users. Such systems have become available to the public use for free like marketing platforms [35]. User identification through facial recognition has indeed become an inevitable reality and significantly hurts user privacy.

4.2. User Tracking. User tracking is primarily based on user identification, and it becomes a threat when the data collected about a certain user is maintained and used to track this user's behaviour. The most famous type of user tracking is based on location. When a user is identified, binding the location history enables tracking. Location-based services require that users share their location information. Thus, user location can be tracked without users' explicit consent and likely without their knowledge. Several technologies significantly affect user tracking such as positioning techniques, which has made great developments in recent years. Positioning techniques are typically based on Global Positioning Systems (GPS), GSM, RFID, and the Wireless LAN [36, 37]. The work done in [38] shows that average Facebook users significantly underestimate the amount of data to which they allow third-party applications access.

The evolution of such technologies provides service providers with a tool to learn about personal patterns (e.g., home location, work location, and visited places), which raises the concern of location privacy intrusion. Since the IoT market is open, user location information can be abused or sold to third parties for targeted advertisements purposes. More seriously, criminals could exploit such data to perform various types of criminal activities that risk individual's life. For example, the use of GPS to stalk customers [39–41] and, generally, the uncomfortable feeling of being observed are discussed in [42]. Even when fake identity is used, the system cannot overcome such a privacy threat with location-based services enabled [43].

4.3. Profiling. Profiling [44] refers to recording and analyzing data to characterize personal behaviour to assess or infer their personal interests in a certain domain or for discrimination purposes. Off-the-shelf data mining tools can draw a clear picture of the customer needs and easily provide a detailed customer profile. Following the rule "know your customer" [45, 46], in e-commerce, online profiling is a key tool for companies to better understand their customer needs. Profiling data is increasingly used for target advertisements, Web sites personalization, and service matching. However, profiling leads to privacy violation when used to learn a customer's political and religious views, sexual orientation, and/or medical conditions [47–49], valuable information that can be shared and sold without further consent [50–54]. The rising of Internet-connected systems and the evolution of data mining algorithms and tools significantly contributed to the emergence of big data [55]. From IoT and big data perspectives, the argument is that limiting access to private/personal data negatively impacts the accuracy of the data mining exercise. Besides this conflict of interest between privacy and profiling, we noticed that identification and tracking threats further aggravate the possibilities for profiling and increase the risks of privacy leakage by data hunting black markets.

4.4. Utility Monitoring and Controlling. This threat is directly relevant to gathering data related to customers' utility usage. Such data could be used to infer user's daily life patterns. This sensitive information represents major privacy threat if acquired through an unauthorized access. However, it becomes more serious when attackers gain privileged access to control utility usage without the user's explicit permission or knowledge. Gubbi et al. [56] categorize IoT applications

into four domains: (1) Personal and Home, (2) Enterprise, (3) Utilities, and (4) Mobile. In personal and home applications, Wi-Fi is typically used to provide high bandwidth for video streaming services and support high sampling rates for audio streaming as well as control of home appliances such as air conditioners, refrigerators, and washing machines. In Enterprise applications, IoT devices collect data from workplace environment. For example, environmental monitoring applications keep track of the number of occupants and manages the utilities within the building (e.g., HVAC, lighting). If attackers gain access to these devices, they can cause financial and personal harm to owners. Although utility companies claim that they collect data to optimize their service, the granularity at which such data is collected raise concerns. Fine granularity data may reveal private information that users do not want to share. For example, collecting water usage at high sampling rate can reveal whether customers are home or not. Data analytics tools also can show when customers take showers or use the bathroom. All these are private information that users would not feel comfortable sharing with others. For example, smart grid and smart metering [57], image processing, computer vision to support video based IoT [58], and irrigation monitoring in the agricultural industries [59]. It is highly challenging to control the disclosure of all this information from these different applications. It is also readily viable that companies could store customers' data and retain it indefinitely due to increasingly advancing storage technology with continuously decreasing prices.

The increasing evolution and adoption of IoT continue to aggravate user privacy and present new challenges on supporting infrastructure to provide more robust privacy preserving techniques. Users need to be aware of entities collecting their private data, understand how this data is shared outside their control domain, evaluate the purpose of access, estimate potential data misuse, and assess associated risks and consequences. Such requirements pose additional challenges on IoT infrastructures to provide users with safe and privacy preserving environments.

Parker Higgins [60] tweeted about the unsettling similarity of the Samsung Smart TV privacy policy, which warned consumers not to discuss sensitive topics near the device [61]. This incident led Samsung to edit its privacy policy and clarify the Smart TV's data collection practices [62]. With IoT becoming an important part of everyday life, people must pay extra attention to their privacy and systems must implement ethical practices in dealing with private data. Users should always be aware of the exact purpose of data collection and understand the spectrum of potential misuse. There must be also continuous enforcing mechanisms for access policies. The control should ultimately be placed at the users' hands to make informed decisions on how their private data is collected and shared beyond their control domain.

5. Classifications of IoT Privacy Solutions

Privacy issues in traditional Internet mostly impact connected users surfing the Internet. However, in IoT scenarios, privacy concerns may affect people who are not even using any IoT service but happen to be present in the environment. In traditional Internet services, the W3C group has defined the Platform for Privacy Preferences (P3P) [63], which provides a standard language for the description of privacy preferences and policies. P3P allows for automatic negotiation of the privacy concerning parameters based on data needed to run the service and the privacy requirements set by the user. Internet applications can implement well-established authentication procedures to capture the data flow and determine whether there are any potential privacy violations and immediately notify the user. However, in IoT settings it is far complex to precisely capture privacy violations due to the lack of well-defined control domain boundaries. Therefore, IoT environments must respect the privacy of individuals and ensure that collected personal data must be used for absolutely nothing, but the intended purpose. Lastly, collected data must be stored only until it is strictly needed.

This section discusses the proposed solutions, summarized in Table 3, to overcome privacy challenges and related security issues as follows.

(i) Authentication and authorization

(ii) Edge computing and plug-in architectures

(iii) Data anonymization

(iv) Digital forgetting and data summarization

5.1. Authentication and Authorization. Authentication in IoT scenarios is challenging due to the limitations of IoT devices. However, many researchers have proposed lightweight solutions to address these limitations and support authentication in constrained environments. Lee et al. [64] proposed simple and secure key establishment to be used in IoT networks. The authors introduce an encryption method based on XOR operations to implement a lightweight cryptography protocol. The hardware implementation of this protocol is demonstrated and can be used to establish the mutual authentication procedure in a typical RFID system for IoT applications.

Porambage et al. [65] propose PAuth Key protocol, an authentication scheme and keying mechanism suitable for resource-constrained WSNs (a.k.a. IoT), irrespective of their vendor or form factor. PAuth provides application-level end-to-end security through two phases: registration and authentication. In the registration phase, end users and edge devices obtain their cryptographic credentials. The authentication phase establishes key-based authentication using mutual communication. The protocol allows end users to authenticate with the sensing nodes directly and acquire sensor data and services. The protocol supports distributed IoT applications since the certificates are lightweight and can be handled by resource-constrained devices.

Sharaf-Dabbagh et al. [66] propose a new authentication framework for IoT environments based on device fingerprinting techniques. According to their model, each IoT device has a unique fingerprint, which can be used to communicate with the cloud infrastructure. The model provides

Table 3: Summary of privacy preserving proposed solutions in IoT environments.

Solution	Summary	References
Authentication and Authorization	(i) Lightweight authentication and key establishment mechanisms (ii) Frameworks based on device fingerprinting techniques (iii) Context-aware access control models and enforcing mechanisms	[64–68]
Edge Computing and plug in architecture	(i) Software modules on the edge to overcome privacy concerns (ii) Privacy aware systems to allow user control over data (iii) Decentralized architectures based on Personal-Cloud Butlers	[69–74]
Data Anonymizing and denaturing	(i) Data brokers and separation algorithms to offer flexibility to service providers, yet respect user-predefined access rules (ii) Generalization to mask personal data (iii) Frameworks that provide emotion analytics lifecycle to allow denaturing	[75–84]
Digital Forgetting and Data Summarization	(i) Delete encrypted data when decryption key is deleted (ii) Acquire only the strictly needed data rather than all data (iii) Apply knowledge discovery in databases and data mining technologies	[85–90]

authentication of IoT devices through a twofold approach: (1) a generative model to verify that the received messages belong to a certain object; (2) validation of the sender legitimacy to ensure that it is not a malicious object. The authors adopted the infinite Gaussian mixture model (IGMM) as a generative model if the object fingerprint follows a multivariate Gaussian distribution. The second validation method is implemented using Bhattacharyya distance to compare the clustering results from IGMM with the expected cluster shape for the device. Then, the proposed framework uses transfer learning techniques to effectively detect emulation attacks, thus, differentiating between fingerprint abnormalities resulting from environments versus attacks.

Bouij-Pasquier et al. [67] propose SmartOrBAC, a context-aware authorization model that accommodates IoT network requirements. SmartOrBAC leverages real-time context to make informed authorization decisions. The authors separate functionality into multiple layers and resource-constrained devices collaborate to perform tasks using distributed processing.

Salman et al. [68] propose an authentication scheme for heterogeneous IoT environments based on Software Defined Network (SDN). SDN controllers are used to manage security parameters by implementing a trusted certificate authority. All SDN controllers rely on a central SDN controller that translates different technology-specific identities into a single shared identity scheme based on virtual IPv6 addresses. This shared identity is then used to authenticate devices and gateways. The SDN controller authenticates gateways and gateways authenticate their associated devices. The proposed scheme is performed in three steps: (1) the gateway obtains an authentication certificate from a controller, (2) things register with the gateway, and (3) IoT devices send authentication requests to the gateway. Their analysis and experimental results show that the proposed scheme is secure against replay attack, masquerade attack, and man-in-the-middle attack.

5.2. Edge Computing and Plug-In Architectures. There is a growing adoption of the edge computing paradigm [91] in the last few years. In edge computing data processing and storage occur partially at the network edge, rather than completely in the backed. Due to the increasing trend of generating data at the edge of the network, it makes more sense to leverage edge computing to resolve concerns such as latency, device limitations, security, and more importantly user privacy [92].

Davies et al. [69] discuss the concern of data privacy in IoT networks, following Geoffrey Moore's warning [93] about the discontinuity awaiting every new technology. The authors introduce a plug-in mediator solution to overcome the privacy concerns stemming from overcentralization of IoT systems. The proposed architecture suggests deploying privacy mediators (i.e., trusted software modules) into the data distribution pipeline. A mediator runs on a cloudlet [94] to enforce the privacy policy specified by sensor/user. While developers can provide sensor drivers that convert data into common formats, customers can create privacy policies that control the mediator configuration and the sensor data routing to/from that mediator. This architecture enables the implementation of various types of data privacy controls such as deletion, denaturing, summarization, inference, anonymization, and mobility.

Langheinrich [70] presents a privacy-aware system (pawS) to overcome the privacy concerns by guaranteeing that collected data remains private. It provides data collection and processing tools that notify users of what exactly is collected. Thus, the user decides what actions can be taken. The proposed architecture adopts privacy preserving principles in ubiquitous computing [71]. The architecture encompasses four components: (1) machine-readable privacy policies to provide choice and consent, (2) policy announcement mechanisms to give notice, (3) privacy proxies to support access control, and (4) policy-based data access for protected recourses. However, proximity, negotiation, and locality are not implemented in this system. This architecture presents to

customers all available options upfront to choose from, rather than forcing them to negotiate with an automated process to get the best deal.

Bagüés et al. [72] introduce a privacy preserving framework for smart homes (Sentry@HOME). The framework adopts a user-centric approach to control the dissemination of private data according to the privacy policies defined by the user. It consists of five essential components: Sentry Registry (SR), Sentry Implementation (SI), Context Handler (CH), Sentry Manager Interface (SMI), and the Noise Module (NM). The framework embeds privacy enforcements into existing smart home infrastructure. The authors demonstrate that the smart home is a safe harbour for privacy-sensitive data and that their framework acts as a guardian sentry.

Seong et al. [73] present a decentralized architecture of PrPl, which proposes Personal-Cloud Butlers as a safe harbour for personal data indexing. A butler is configured for each user to provide fine-grain access control and storage. A similar work is presented in [74] that investigates on-device sensor abstractions for augmented reality applications to prevent private data from accidental leakage from applications having privileged access to raw sensor data.

5.3. Data Anonymization. Data anonymization is the process of removing identifiable information that may lead to personal identification so that people/objects described by such data remain anonymous [75]. The purpose of data anonymization is generally to protect user privacy. Several attempts have been made to provide anonymization, image blurring, and denaturing mechanisms for IoT applications [77, 78], especially for images and videos. Denaturing is the process of using image processing techniques to blur or alter a specific part of the image to preserve personal privacy. Data anonymization not only protects user privacy but also enables service providers to use collected data to customize the services for users. Data anonymization has three main objectives [76]: protecting the privacy of involved users, hiding any information about the network internal structure, and maintaining the anonymized traffic traces as realistic as possible to the nonanonymized packet stream.

Sliwa [79] presents a new framework for anonymized data exchange that integrates user privacy, system safety, and quality of service. The main challenge is how to design a data broker that takes only general knowledge about the required data communication and possibly unaware of the semantics, yet assures reliable and secure data exchange between partners. The author also points out the challenges of developing data separation algorithms that allow for certain flexibility to service providers, yet respect the predefined access rules (personal identity, granularity).

Berrehili and Belmekki [80] present a deep risk analysis for IoT privacy threats. The authors propose several technical and nontechnical approaches to protect user privacy in IoT scenarios. They provide a recommendation for IoT app developers to inform users about potential privacy violations resulting from private data disclosure. The authors also propose that IoT devices implement an authentication algorithm to verify the source of updated files using a cryptographic mechanism. They also suggest using anonymization techniques to mask the personal information in the data before sharing.

Shinzaki et al. [81] extend the identification-based key sharing scheme to TLS to implement mutual authentication, encrypt data communication, and provide anonymization technology for the safe disclosure of data using multiple layers of meshes on a map for the utilization of positional data. Similarly, Otgonbayar et al. [82] present a new anonymization algorithm based on the k–anonymity privacy model. The proposed algorithm uses the time-based sliding window technique to manipulate IoT streams by partitioning the stream tuples based on their description.

Wang et al. [83] introduce a scalable privacy-aware IoT architecture that enables live video analytics across many cameras by combining OpenFace [95], a high accuracy open-source face recognizer, with face tracking to maintain high accuracy and achieve full frame rate speeds. Authors also use privacy mediators to enforce user-defined privacy policies (e.g., face denaturing), yet the system maintains the original videos for possible future needs (e.g., finding an evidence from a crime scene).

Addo et al. [84] present a reference framework for protecting end user's privacy throughout the emotion analytics lifecycle. They propose Affect-Driven Personalization Lifecycle (ADPL), a model to learn the privacy preferences of end users through implementing a set of privacy rules: personalized anonymity, secure multiparty privacy preservation, encrypted data provenance, image-melding and reshaping techniques, and result aggregation.

5.4. Digital Forgetting and Data Summarization. Digital forgetting is the process of provably deleting all copies of a dataset [96], while data summarization provides a high-level data abstraction to hide details or reduce granularity. Digital forgetting and data summarization are important concepts to relieve people's anxiety around data collection. Users would feel more comfortable sharing their data knowing that collected data will be wiped out once the purpose of collection is void [85].

Data summarization is classified into the following:

(i) Temporal summarization: in which collected data is a function of time (e.g., gather sensor reading per day rather than per minute)

(ii) Spatial summarization: in which collected data is a function of location (e.g., releasing location data at the zip code level rather than raw GPS readings)

As the cost of storage decreases, the ability to store a large amount of data at low cost dramatically increases. This makes it easy for entities collecting data to store it for longer time. Hence, the need for creating efficient mechanisms that periodically delete information that is no longer needed for which it was generated arises. Therefore, IoT environments must take into consideration applying either data forgetting or summarization to protect user privacy. A few experimental solutions have been recently developed that allow users to share private data over the Internet with assurance that such

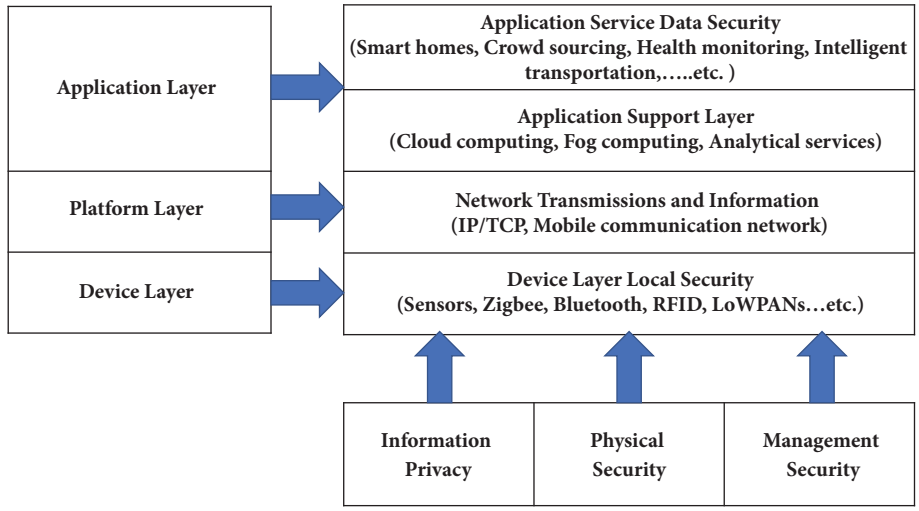


Figure 3: IoT three-layered architecture.

data will be entirely deleted after a certain period of time (e.g., drop.io and the Guest Pass features on Flickr [97]). However, porting such solutions to IoT environments is not straightforward. Therefore, several data forgetting techniques have been proposed using classical cryptography [86]. Most of these techniques assume that encrypted data is deleted when the required decryption key is deleted. Several other techniques have been developed based on distributed data storage so that data is deleted due to unavoidable social and technical processes [87].

Despite digital forgetting is an effective tool to preserve user privacy, it becomes challenging when the data size grows very large due to processing and analytics overhead [98]. Therefore, efficient data analytics techniques are required to reduce such overhead and optimize the process. To address this problem, Baraniuk [88] proposes to enforce IoT devices to acquire only important data instead of everything. Also, effective knowledge discovery in databases (KDD) and data mining techniques [89, 90] are effective solutions to summarize data collected by IoT devices, which can enhance the overall system performance and improve quality of service yet preserve user privacy. These methods are essential in IoT scenarios, in which large amounts of collected data can seriously impact the privacy of individuals and compromise the security of economic entities and government institutions.

6. Privacy Preserving IoT Environments

Applying existing Internet standards to smart devices can simplify the integration of the envisioned scenarios in the IoT contexts. However, the security mechanisms in conventional Internet protocols need to be modified or extended to preserve user privacy in IoT applications. In this section, we discuss privacy preserving at different layers of the IoT stack. We build our discussions based on a three-layer IoT stack as shown in Figure 3.

6.1. Privacy Preservation in IoT Device Layer. The IoT device layer (also known as perception layer) contains all physical resources that collect/control data (sensors and actuators). However, these resources are highly heterogeneous and resource-constrained. Such constraints pose unique challenges on applying privacy preserving techniques. Thus, IoT devices are subject to several attacks discussed in [99] including node capture, fake node, malicious data, denial of service attack (DoS), timing attack, routing threats, replay attack, side channel attack (SCA), and mass node authentication problem. Therefore, several security measures must be considered when designing this layer as follows:

(i) Access control and authentication: to prevent user privacy leaks from open and unauthorized access. Juels et al. [100] present a good solution to implement Selective RFID Jamming as an access control scheme on low-cost tags

(ii) Data encryption: to secure data exchange and guarantee safe delivery. Wang [101] presents a nonlinear key algorithm based on displaced calculation to provide data encryption. This key algorithm requires low computational power to provide high security and good data transmission rate

(iii) Secure channel using IPSec: the IPSec protocol [102] offers both authentication and encryption. Raza et al. [103] present a 6LoWPAN/IPsec extension to provide security for IoT devices. The authors demonstrate that IPSec outperforms the standard IEEE 802.15.4 link layer security in IoT environments

(iv) Cryptography technology: to offer privacy protection, confidentiality, authenticity and data integrity. Secure communication protocols include digital signatures and hash values are used to ensure data integrity

6.2. Privacy Preservation in Platform/Infrastructure Layer. The platform layer represents the classical network layer in

the OSI model [104]. This layer integrates intelligent data preprocessing to reduce resource requirements at the application layer. The network layer poses some general security problems related to data integrity and confidentiality such as unauthorized access to networks, eavesdropping, confidentiality and integrity damage, DDoS attacks, and man-in-the-middle attacks. Although existing network protocols implement highly secured measures, they are not robust enough for M2M communications in resource-constrained environments. As such, existing security mechanisms are weak/inapplicable on IoT devices and may lead to creating barriers rather than connections between different machines. Therefore, the heterogeneity of these networks makes security, interoperability, and coordination of networks becoming worse, leading to security vulnerabilities. New IoT-oriented security mechanisms must be designed from the ground up to fit IoT environments, taking into consideration the following security measures:

(i) Set up an end-to-end authentication and key agreement mechanism, PKI (Public-Key Infrastructure), WPKI for wireless, Security routing, IDS, etc.

(ii) Utilize network virtualization to reduce the network management complexity and the likelihood of improper operations.

(iii) Adopt IPv6 as a standard network layer protocol to support inherited security mechanisms [105].

6.3. Privacy Preservation in Application Layer. The application layer encompasses 2 parts: the support layer where the edge computing and analytical services run and the application service layer that provides necessary support from the IoT infrastructure. IoT applications are also highly versatile and heterogonous with varying needs, which makes it a challenge to offer a standard support. Different applications target different domains with unique data collection requirements, which may require different security measures. Therefore, the application layer security considerations/requirements differ from the previous two layers in the following sense:

(1) Nontechnical:

(a) Privacy awareness: makes users aware of private data collection, potential risks, and how to safely use IoT services and avoid private information leakage.

(b) Security management: strengthens resources, physical security information, password management, etc.

(2) Technical:

(a) Cryptography: fingerprint technology, digital watermarking, anonymous authentication, and homomorphic and threshold cryptography.

(b) Key agreements: incorporate symmetric and asymmetric cryptosystems and certification transfer technology.

7. Conclusion

The Internet of Things has the potential to change the world, just as the Internet did two decades ago. Nevertheless, any new technology faces several technical and nontechnical challenges. The highly diverse IoT application domains, resource-constrained IoT devices, and heterogeneity of both devices and platforms hinder the development of a standard IoT framework. However, privacy stands out as a critical concern that inhibits the widespread adoption IoT. The vulnerabilities of IoT devices can lead to huge security breaks and significantly hurt user privacy by exposing personal data. To promote IoT adoption and relieve user concerns, platforms, applications, and infrastructures must seriously take privacy into consideration. In this survey paper, we outline the major privacy threats in IoT environments and discuss the impact of IoT evolution on each threat. Privacy concerns such as user identification lead to a much bigger threat such as profiling. We surveyed the proposed solutions that overcome various privacy concerns and security threats in IoT environments. Most of the proposed solutions fall into one of the following categories: (1) authentication and authorization, (2) edge computing mediators, (3) data anonymization, and (4) data summarization. Consequently, several efforts were focused on providing lightweight authentication and keying establishment mechanisms, implementing frameworks based on device fingerprinting techniques, and introducing context-aware access control models.

Although the proposed solutions can relieve some of the privacy concerns in IoT scenarios, there is a clear lack of performance evaluation and assessment in real-life scenarios. Furthermore, there is a conflict between protecting user privacy and the granularity of data access needed to provide better services. This raises the challenge of how to support consumer-specific privacy preferences while maintaining the same level of service. Such a challenge could be addressed using data anonymization. The paper then points out the required measures to preserve privacy in the different layers of the IoT stack.

Our recommendation for IoT-oriented privacy preservation in IoT environments is as follows. First, take security measures into consideration at the device layer including access control and authentication, data encryption, secure channel based on IPSec, and cryptography. Second, reduce network management complexity, set up cohesive authentication mechanism, and adoption of IPv6 must be considered in the platform layer.

In conclusion, privacy preserving is a shared responsibility in which all parties much actively engage and cooperate to provide safe IoT environments yet enjoy what IoT may offer. Technology manufacturer must design IoT devices with integrated privacy and security measures. Infrastructures must implement IoT-oriented mechanisms to prevent privacy leaks and address security threats from the ground up. IoT applications must notify users of what data is collected and the purpose of collection. IoT users must take extra cautions when they authorize access to their private data and better understand the potential consequences of any associated risks resulting from any misuse of such data.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] *Worldwide Internet of Things Forecast*, Sep 2017, https://www.idc.com/getdoc.jsp?containerId=IDC_P24793.

[2] Report. Gartner, *Forecast: The Internet of Things, Worldwide*, The Internet of Things, Forecast, 2017.

[3] A. Dunkels, B. Grönvall, and T. Voigt, "Contiki—a lightweight and flexible operating system for tiny networked sensors," in *Proceedings of the 29th IEEE Annual International Conference on Local Computer Networks (LCN '04)*, pp. 455–462, November 2004.

[4] E. Baccelli, O. Hahm, M. Gunes, M. Wahlisch, and T. Schmidt, "RIOT OS: Towards an OS for the Internet of Things," in *Proceedings of the 2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 79-80, Turin, April 2013.

[5] I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," *Business Horizons*, vol. 58, no. 4, pp. 431–440, 2015.

[6] T. Rault, A. Bouabdallah, and Y. Challal, "Energy efficiency in wireless sensor networks: a top-down survey," *Computer Networks*, vol. 67, pp. 104–122, 2014.

[7] L. D. Xu, W. He, and S. Li, "Internet of things in industries: a survey," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.

[8] S. M. Riazul Islam, D. Kwak, M. Humaun Kabir, M. Hossain, and K.-S. Kwak, "The internet of things for health care: a comprehensive survey," *IEEE Access*, vol. 3, pp. 678–708, 2015.

[9] "Internet of Things (IoT): Security Analysis & Security Protocol CoAP," *International Journal of Recent Trends in Engineering and Research*, vol. 3, no. 3, pp. 417–425, 2017.

[10] Your. Proofprint, *Fridge is Full of SPAM*, 2014, [Online]., Available https://www.proofpoint.com/us/threat-insight/post/Your-Fridge-is-Full-of-SPAM.

[11] Cloud. Talkin, *Iot past and present: The history of iot, and where its headed today*, 2016, [Online]., Available http://talkincloud.com/cloud-computing/iot-past-and-present-historyiot-and-where-its-headed-today?page=2.

[12] M. Ritamaki and A. Ruhanen, "Embedded passive UHF RFID seal tag for metallic returnable transit items," in *Proceedings of the 2010 IEEE International Conference on RFID (IEEE RFID 2010)*, pp. 152–157, Orlando, FL, April 2010.

[13] J. H. Ziegeldorf, O. G. Morchon, and K. Wehrle, "Privacy in the internet of things: threats and challenges," *Security and Communication Networks*, vol. 7, no. 12, pp. 2728–2742, 2014.

[14] A.-R. Sadeghi, C. Wachsmann, and M. Waidner, "Security and privacy challenges in industrial Internet of Things," in *Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference, DAC 2015*, USA, June 2015.

[15] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: the road ahead," *Computer Networks*, vol. 76, pp. 146–164, 2015.

[16] H. Suo, J. Wan, C. Zou, and J. Liu, "Security in the internet of things: a review," in *Proceedings of the International Conference on Computer Science and Electronics Engineering (ICCSEE '12)*, pp. 648–651, Hangzhou, China, March 2012.

[17] M. R. Palattella, N. Accettura, X. Vilajosana et al., "Standardized protocol stack for the internet of (important) things," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1389–1406, 2013.

[18] IEEE. std, *Part. 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs) Standard for Information Technology Std*, 19 IEEE std. 802.15.4 Part. 15.4, Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs) Standard for Information Technology Std. (September, 2006.

[19] T. Yu, V. Sekar, S. Seshan, Y. Agarwal, and C. Xu, "Handling a trillion (unfixable) flaws on a billion devices: Rethinking network security for the Internet-of-Things," in *Proceedings of the 14th ACM Workshop on Hot Topics in Networks, HotNets-XIV 2015*, USA, November 2015.

[20] Z. Durumeric, J. Kasten, D. Adrian et al., "The matter of heartbleed," in *Proceedings of the 2014 ACM Internet Measurement Conference, IMC 2014*, pp. 475–488, Canada, November 2014.

[21] Shodan. March, *Devices Vulnerable to Heartbleed [Online]. Available*, 2016, https://www.shodan.io/report/89bnfUyJ.

[22] T. Heer, O. Garcia-Morchon, R. Hummen, S. L. Keoh, S. S. Kumar, and K. Wehrle, "Security challenges in the IP-based Internet of Things," *Wireless Personal Communications*, vol. 61, no. 3, pp. 527–542, 2011.

[23] G. O. Yee, *Privacy Protection Measures and Technologies in Business Organizations*, IGI Global, 2012.

[24] https://www.eia.gov/.

[25] A. Molina-Markham, P. Shenoy, K. Fu, E. Cecchet, and D. Irwin, "Private memoirs of a smart meter," in *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings, BuildSys'10*, pp. 61–66, Switzerland, November 2010.

[26] U. Feige, A. Fiat, and A. Shamir, "Zero-knowledge proofs of identity," *Journal of Cryptology*, vol. 1, no. 2, pp. 77–94, 1988.

[27] T. Datta, N. Apthorpe, and N. Feamster, "A Developer-Friendly Library for Smart Home IoT Privacy-Preserving Traffic Obfuscation," in *Proceedings of the the 2018 Workshop*, pp. 43–48, Budapest, Hungary, August 2018.

[28] K. Yang, K. Zhang, J. Ren, and X. Shen, "Security and privacy in mobile crowdsourcing networks: challenges and opportunities," *IEEE Communications Magazine*, vol. 53, no. 8, pp. 75–81, August 2015.

[29] X. Jia, Q. Feng, and C. Ma, "An efficient anti-collision protocol for RFID tag identification," *IEEE Communications Letters*, vol. 14, no. 11, pp. 1014–1016, 2010.

[30] K. Finkenzeller, "RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and near-Field Communication," *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and near-Field Communication*, 2010.

[31] G. T. Huang, "10 emerging technologies that will change your world," *IEEE Engineering Management Review*, vol. 32, no. 2, pp. 20-20, 2004.

[32] R. K. Pateriya and S. Sharma, "The evolution of RFID security and privacy: A research survey," in *Proceedings of the 2011 International Conference on Communication Systems and Network Technologies, CSNT 2011*, pp. 115–119, India, June 2011.

[33] X. Liu, N. Krahnstoever, T. Yu, and P. Tu, "What are customers looking at?" in *Proceedings of the 2007 IEEE Conference on Advanced Video and Signal Based Surveillance, AVSS 2007*, pp. 405–410, UK, September 2007.

[34] A. W. Senior, L. Brown, A. Hampapur et al., "Video analytics for retail," in *Proceedings of the 2007 IEEE Conference on Advanced Video and Signal Based Surveillance, AVSS 2007*, pp. 423–428, UK, September 2007.

[35] *Handbook of Face Recognition*, Springer-Verlag, New York, 2005.

[36] *Pro PayPal E-Commerce*, Apress, Berkeley, CA, 2007.

[37] J. Wrrior, E. McHenry, and K. McGee, "They know where you are," *IEEE Spectrum*, vol. 40, no. 7, pp. 20–25, 2003.

[38] J. Golbeck and M. L. Mauriello, "User perception of Facebook app data access: A comparison of methods and privacy concerns," *Future Internet*, vol. 8, no. 2, 2016.

[39] J. Voelcker, "Stalked by satellite—an alarming rise in GPS-enabled harassment," *IEEE Spectrum*, vol. 43, no. 7, pp. 15-16, 2006.

[40] M. Z. Newman, "Crazy Ex-Girlfriend," *Film Criticism*, vol. 40, no. 3, 2016.

[41] Cop stalked ex-wife before killing her, https://www.usatoday.com/story/news/nation//07/29/cop-stalked-ex-wife//.

[42] C. Chow and M. F. Mokbel, "Privacy in location-based services," *SIGSPATIAL Special*, vol. 1, no. 2, pp. 23–27, 2009.

[43] E. Toch, Y. Wang, and L. F. Cranor, "Personalization and privacy: A survey of privacy risks and remedies in personalization-based systems," *User Modeling and User-Adapted Interaction*, vol. 22, no. 1-2, pp. 203–220, 2012.

[44] M. Hildebrandt, "Defining profiling: A new type of knowledge?" *Profiling the European Citizen: Cross-Disciplinary Perspectives*, pp. 17–45, 2008.

[45] Patrick. Thibodeau, Online Profiling, [Online], https://www.computerworld.com/article/2597220/retail-it/online-profiling.html.

[46] A. Odlyzko, "Privacy, economics, and price discrimination on the Internet," in *Proceedings of the the 5th international conference*, pp. 355–366, Pittsburgh, Pennsylvania, September 2003.

[47] *Profiling and Targeting - Behavioral Advertisers Beware!*, [Online], URL https://www.ecommercetimes.com/story/73966.html.

[48] G. L. Orgill, G. W. Romney, M. G. Bailey, and P. M. Orgill, "The urgency for effective user privacy-education to counter social engineering attacks on secure computer systems," in *Proceedings of the the 5th conference*, p. 177, Salt Lake City, UT, USA, October 2004.

[49] G. T. Marx, "The surveillance society: the threat of 1984-style techniques. in. The Futurist," in *June 21-6*, p. 21, The surveillance society, the threat of 1984-style techniques. in. The Futurist, 1985.

[50] A. Vedder, "KDD: The challenge to individualism," *Ethics and Information Technology*, vol. 1, no. 4, pp. 275–281, 1999.

[51] D. Lyon, *Surveillance as Social Sorting*, Routledge, 2005.

[52] B. Custers, "Effects of unreliable group profiling by means of data mining," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 2843, pp. 291–296, 2003.

[53] M. Hildebrandt and S. Gutwirth, *Profiling the European Citizen*, Springer Netherlands, Dordrecht, 2008.

[54] J. Menn, "Social networks scan for sexual predators, with uneven results," in *Reuters. [Online. Last accessed*, pp. 2013-02, 2012, [Online. Last accessed 2013-02-07] http://reut.rs/Nnejb7.

[55] S. John Walker, "Big Data: A Revolution That Will Transform How We Live, Work, and Think," *International Journal of Advertising*, vol. 33, no. 1, pp. 181–183, 2015.

[56] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): a vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[57] M. Yun and B. Yuxin, "Research on the architecture and key technology of Internet of Things (IoT) applied on smart grid," in *Proceedings of the International Conference on Advances in Energy Engineering (ICAEE '10)*, pp. 69–72, June 2010.

[58] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," *Computer Networks*, vol. 51, no. 4, pp. 921–960, 2007.

[59] L. Zhao, S. Yin, L. Liu, Z. Zhang, and S. Wei, "A crop monitoring system based on wireless sensor network," in *Proceedings of the 2011 2nd International Conference on Challenges in Environmental Science and Computer Engineering, CESCE 2011*, pp. 558–565, China, December 2011.

[60] Smart. Samsung's, TV privacy policy sounds like an Orwellian nightmare [online], https://www.theverge.com//2/8//samsungs-smart-tv-privacy-policy-george-orwell.

[61] https://epic.org/privacy/internet/ftc/EPIC-Letter-FTC-AG-Always-On.pdf.

[62] Y.-A. de Montjoye, *Computational PRIvacy: Towards PRIvacy-Conscientious Uses of Metadata*, ProQuest LLC, Ann Arbor, MI, 2015.

[63] J. Reagle and L. F. Cranor, "The platform for privacy preferences," *Communications of the ACM*, vol. 42, no. 2, pp. 48–55.

[64] J.-Y. Lee, W.-C. Lin, and Y.-H. Huang, "A lightweight authentication protocol for Internet of Things," in *Proceedings of the 3rd International Symposium on Next-Generation Electronics, ISNE 2014*, Taiwan, May 2014.

[65] P. Porambage, C. Schmitt, P. Kumar, A. Gurtov, and M. Ylianttila, "Pauthkey: a pervasive authentication protocol and key establishment scheme for wireless sensor networks in distributed IoT applications," *International Journal of Distributed Sensor Networks*, vol. 2014, Article ID 357430, 14 pages, 2014.

[66] Y. Sharaf-Dabbagh and W. Saad, "On the authentication of devices in the Internet of things," in *Proceedings of the 17th International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2016*, Portugal, June 2016.

[67] I. Bouij-Pasquier, A. Ait Ouahman, A. Abou El Kalam, and M. Ouabiba De Montfort, "SmartOrBAC security and privacy in the Internet of Things," in *Proceedings of the 12th IEEE/ACS International Conference of Computer Systems and Applications, AICCSA 2015*, November 2015.

[68] O. Salman, S. Abdallah, I. H. Elhajj, A. Chehab, and A. Kayssi, "Identity-based authentication scheme for the Internet of Things," in *Proceedings of the 2016 IEEE Symposium on Computers and Communication, ISCC 2016*, pp. 1109–1111, Italy, July 2016.

[69] N. Davies, N. Taft, M. Satyanarayanan, S. Clinch, and B. Amos, "Privacy mediators: Helping IoT cross the chasm," in *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications, HotMobile 2016*, pp. 39–44, USA, February 2016.

[70] M. Langheinrich, "A Privacy Awareness System for Ubiquitous Computing Environments," in *UbiComp 2002: Ubiquitous Computing*, vol. 2498 of *Lecture Notes in Computer Science*, pp. 237–245, Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[71] M. Langheinrich, "Privacy by design-principles of privacy-aware ubiquitous systems," in *Proceedings of the Ubicomp 2001: Ubiquitous Computing*, Lecture Notes in Computer Science, pp. 273–291, Springer, Berlin, Germany, 2001.

[72] S. A. Bagüés, A. Zeidler, F. Valdivielso, and I. R. Matias, "Sentry@Home - Leveraging the smart home for privacy in pervasive computing," *International Journal of Smart Home*, vol. 1, no. 2, pp. 129–146, 2007.

[73] S.-W. Seong, J. Seo, M. Nasielski et al., "PrPl: A decentralized social networking infrastructure," in *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing and Services: Social Networks and Beyond, MCS'10, Co-located with ACM MobiSys 2010*, USA, June 2010.

[74] J. Vilk, D. Molnar, B. Livshits et al., "SurroundWeb: Mitigating Privacy Concerns in a 3D Web Browser," in *Proceedings of the 2015 IEEE Symposium on Security and Privacy (SP)*, pp. 431–446, San Jose, CA, May 2015.

[75] B. Zhou, J. Pei, and W. Luk, "A brief survey on anonymization techniques for privacy preserving publishing of social network data," *ACM SIGKDD Explorations Newsletter*, vol. 10, no. 2, p. 12, 2008.

[76] D. Koukis, S. Antonatos, D. Antoniades, E. P. Markatos, and P. Trimintzios, "A generic anonymization framework for network traffic," in *Proceedings of the 2006 IEEE International Conference on Communications, ICC 2006*, pp. 2302–2309, Turkey, July 2006.

[77] M. K. Reiter and A. D. Rubin, "Crowds: anonymity for Web transactions," *ACM Transactions on Information and System Security*, vol. 1, no. 1, pp. 66–92.

[78] *Benezit Dictionary of Artists*, Oxford University Press, 2011.

[79] J. Sliwa, "A generalized framework for multi-party data exchange for IoT systems," in *Proceedings of the 30th IEEE International Conference on Advanced Information Networking and Applications Workshops, WAINA 2016*, pp. 193–198, Switzerland, March 2016.

[80] F. Z. Berrehili and A. Belmekki, "Privacy Preservation in the Internet of Things," in *Advances in Ubiquitous Networking 2*, vol. 397 of *Lecture Notes in Electrical Engineering*, pp. 163–175, Springer Singapore, Singapore, 2017.

[81] T. Shinzaki, I. Morikawa, Y. Yamaoka, and Y. Sakemi, "IoT security for utilization of big data: Mutual authentication technology and anonymization technology for positional data," *Fujitsu scientific & technical journal*, vol. 52, no. 4, pp. 52–60, 2016.

[82] A. Otgonbayar, Z. Pervez, and K. Dahal, "Toward Anonymizing IoT Data Streams via Partitioning," in *Proceedings of the 13th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS 2016*, pp. 331–336, Brazil, October 2016.

[83] J. Wang, B. Amos, A. Das, P. Pillai, N. Sadeh, and M. Satyanarayanan, "A scalable and privacy-aware IoT service for live video analytics," in *Proceedings of the 8th ACM Multimedia Systems Conference, MMSys 2017*, pp. 38–49, Taiwan, June 2017.

[84] I. D. Addo, P. Madiraju, S. I. Ahamed, and W. C. Chu, "Privacy Preservation in Affect-Driven Personalization," in *Proceedings of the 2016 IEEE 40th Annual Computer Software and Applications Conference, COMPSAC 2016*, pp. 400–405, USA, June 2016.

[85] C. Thompson, "25 Ideas for 2010: Digital Forgetting," *25 Ideas for 2010: Digital Forgetting*, November 2009.

[86] D. Boneh and R. J. Lipton, "A revocable backup system," *in USENIX Security*, pp. 91–96, 1996.

[87] S. Diesburg, C. Meyers, M. Stanovich et al., "TrueErase," in *Proceedings of the the 28th Annual Computer Security Applications Conference*, p. 439, Orlando, Florida, December 2012.

[88] R. G. Baraniuk, "More is less: Signal processing and the data deluge," *Science*, vol. 331, no. 6018, pp. 717–719, 2011.

[89] V. Cantoni, L. Lombardi, and P. Lombardi, "Challenges for data mining in distributed sensor networks," in *Proceedings of the 18th International Conference on Pattern Recognition, ICPR 2006*, pp. 1000–1007, China, August 2006.

[90] T. Keller, F. Thiesse, J. Kungl, and E. Fleisch, "Using low-level reader data to detect false-positive RFID tag reads," in *Proceedings of the Internet of Things (IOT '10)*, vol. 7, pp. 1–8, IEEE, Tokyo, Japan, December 2010.

[91] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[92] W. Shi and S. Dustdar, "The Promise of Edge Computing," *The Computer Journal*, vol. 49, no. 5, pp. 78–81, 2016.

[93] G. Moore, *Crossing the Chasm*, Harpercollins, 1991.

[94] M. Satyanarayanan, P. Bahl, R. Cáceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[95] T. Baltrusaitis, P. Robinson, and L.-P. Morency, "OpenFace: An open source facial behavior analysis toolkit," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision, WACV 2016*, pp. 1–10, March 2016.

[96] Z. N. Peterson, R. C. Burns, J. Herring, A. Stubblefield, A. D. Rubin, and A. Stubblefield, "Secure deletion for a versioning file system," *in File and Storage Technologies (FAST*, vol. 5, p. 11, 2005.

[97] N. Proferes, "Delete: The Virtue of Forgetting in the Digital Age. Viktor Mayer-Schönberger. Princeton, NJ: Princeton University Press, 2009.," *The Journal of Popular Culture*, vol. 45, no. 1, pp. 226–228, 2012.

[98] D. Reed, J. Larus, and D. Gannon, "Imagining the future: thoughts on computing," *The Computer Journal*, vol. 45, no. 1, pp. 25–30, 2012.

[99] K. Zhao and L. Ge, "A survey on the internet of things security," in *Proceedings of the 9th International Conference on Computational Intelligence and Security, CIS 2013*, pp. 663–667, December 2013.

[100] A. Juels, R. L. Rivest, and M. Szydlo, "The blocker tag: Selective blocking of RFID tags for consumer privacy," in *Proceedings of the 10th ACM Conference on Computer and Communications Security, CCS 2003*, pp. 103–111, USA, October 2003.

[101] X. Yi, Y. Liang, E. Huerta-Sanchez et al., "Sequencing of 50 human exomes reveals adaptation to high altitude," *Science*, vol. 329, no. 5987, pp. 75–78, 2010.

[102] S. Kent and K. Seo, "Security Architecture for the Internet Protocol," RFC Editor RFC4301, 2005.

[103] S. Raza, S. Duquennoy, J. Höglund, U. Roedig, and T. Voigt, "Secure communication for the Internet of Things-a comparison of link-layer security and IPsec for 6LoWPAN," *Security and Communication Networks*, vol. 7, no. 12, pp. 2654–2668, 2014.

[104] H. Zimmermann, "OSI reference model—the ISO model of architecture for open systems interconnection," *IEEE Transactions on Communications*, vol. 28, no. 4, pp. 425–432, 1980.

[105] S. Liang, Y. Zhang, and G. Jian, *Development Trend of IPv6-based Information Security Products in Network Layer of IOT [J]. Netinfo Security 8*, 018, 2012.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 2051693, 12 pages
https://doi.org/10.1155/2018/2051693

Research Article

Towards Secure Network Computing Services for Lightweight Clients Using Blockchain

Yang Xu,[1] **Guojun Wang**,[2] **Jidian Yang**,[1] **Ju Ren**,[1] **Yaoxue Zhang**,[1] **and Cheng Zhang**[1]

[1]*School of Information Science and Engineering, Central South University, Changsha 410083, China*
[2]*School of Computer Science and Educational Software, Guangzhou University, Guangzhou 510006, China*

Correspondence should be addressed to Yang Xu; xuyangcsu@csu.edu.cn

Received 27 July 2018; Revised 18 October 2018; Accepted 1 November 2018; Published 13 November 2018

Guest Editor: Constantinos Kolias

The emerging network computing technologies have significantly extended the abilities of the resource-constrained IoT devices through the network-based service sharing techniques. However, such a flexible and scalable service provisioning paradigm brings increased security risks to terminals due to the untrustworthy exogenous service codes loading from the open network. Many existing security approaches are unsuitable for IoT environments due to the high difficulty of maintenance or the dependencies upon extra resources like specific hardware. Fortunately, the rise of blockchain technology has facilitated the development of service sharing methods and, at the same time, it appears a viable solution to numerous security problems. In this paper, we propose a novel blockchain-based secure service provisioning mechanism for protecting lightweight clients from insecure services in network computing scenarios. We introduce the blockchain to maintain all the validity states of the off-chain services and edge service providers for the IoT terminals to help them get rid of untrusted or discarded services through provider identification and service verification. In addition, we take advantage of smart contracts which can be triggered by the lightweight clients to help them check the validities of service providers and service codes according to the on-chain transactions, thereby reducing the direct overhead on the IoT devices. Moreover, the adoptions of the consortium blockchain and the proof of authority consensus mechanism also help to achieve a high throughput. The theoretical security analysis and evaluation results show that our approach helps the lightweight clients get rid of untrusted edge service providers and insecure services effectively with acceptable latency and affordable costs.

1. Introduction

The Internet of Things (IoT) industry has evolved remarkably in the last decade. Currently, there exist more than 13 billion connected IoT devices and this number would increase to 30 billion in the near future [1]. Meanwhile, the emerging network computing technologies, typically, fog/edge computing [2, 3] and transparent computing [4, 5], have significantly extended the abilities of the existing resource-constrained IoT devices, through the network-based service provisioning and sharing mechanisms. For example, in the IoT-oriented edge transparent computing scenario [6, 7], with the aid of block-stream code loading and execution techniques [8], the resource-constrained wearable devices (e.g., wristbands and smartwatches) are enabled to alternately run numerous applications obtained from either the cloud servers or close edge servers (e.g., personal computers), which goes beyond the original capabilities of these local devices (see Figure 1) [9].

However, such flexible and scalable service provisioning paradigms bring increased security risks to terminal devices unintentionally. Comparing to the traditional closed architectures, the attack surfaces of network computing systems have inevitably increased due to the opening service sharing over the network [10, 11]. The frequent-changing exogenous service codes loading on the clients from the remote servers via the network can be unreliable, fragile, and even harmful to the host terminals in absence of adequate security mechanisms [12]. To make things worse, the various edge servers intermingled with vulnerable and malicious ones certainly heighten the risks.

Some early studies have already been done for protecting the terminals from illegal services in network computing

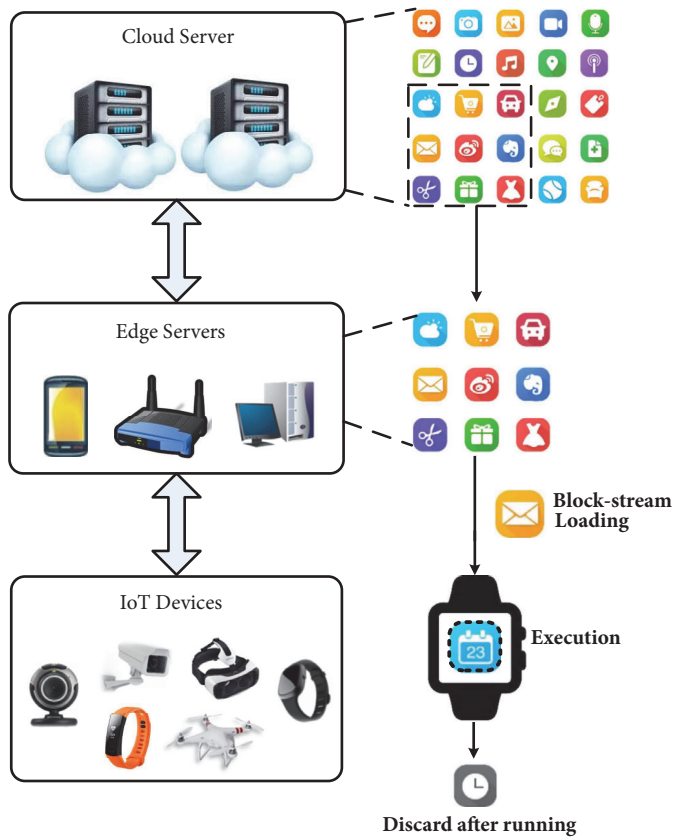


Figure 1: The IoT-oriented edge transparent computing scenario.

scenarios [12–16]. Based on integrity verification techniques, these works equip the terminals with the abilities to check the validities of the acquired service programs before executions with the help of static information (e.g., hash checksum) prestored in local trusted firmware [12–15] or trusted platform module (TPM) [16]. However, when it comes to IoT scenarios [17] in which the vulnerable IoT devices are threatened by distributed cyberattacks, the rigid prestored information is technically less maintainable for updating, while the spare firmware space or specific hardware is usually unavailable.

Recently, the rise of blockchain technologies [18, 19] inspires researchers for brand new solutions. With the excellent features of openness, decentralization, and tamper resistance, the blockchain techniques have been used as the underlying security fabric for a bunch of emerging service provisioning and sharing systems [20–29]. These approaches leverage the blockchain to release service information so as to ensure that the clients can obtain services correctly. Unfortunately, these blockchain-based schemes usually have low throughput and high service latency problems and take little consideration of necessary information updating as well as the legality validation of the numerous service providers. Even worse, few of them are designed for IoT scenarios and thus bring about unaffordable computing and storage costs to most existing IoT devices.

Motivated by the situations mentioned above, in this paper, we proposed a novel blockchain-based secure service provisioning mechanism to protect the lightweight clients from insecure exogenous service codes from untrustworthy edge servers in the edge transparent computing scenario. We leverage the blockchain to maintain all the validity states of the off-chain services and edge servers dynamically updated by the arbitration cloud merchants, to help the lightweight clients get rid of untrusted or discarded services through the provider identification and service verification. Besides, the specific smart contracts [30] are introduced and can be triggered to verify the validities of the edge servers and service codes on behalf of the lightweight clients according to the on-chain transactions, thereby reducing the direct costs of these IoT devices. Furthermore, a consortium blockchain with the proof of authority consensus mechanism [31, 32] is employed to achieve a high throughput and low latency further. Finally, we demonstrate the security of our approach and then test it comprehensively. The evaluation results show that our approach protects the lightweight clients from untrusted edge service providers and undependable service codes effectively with acceptable latency and affordable costs.

Summarily, the major contributions of our work are threefold:

(1) We design a blockchain system to maintain the appendable and tamper-resistant validity states of the off-chain services and edge servers dynamically declared by the arbitration cloud merchants, to help the lightweight clients get rid of insecure or deprecated services by the means of provider identification or service verification.

(2) We not only introduce smart contracts, which can be triggered by the lightweight clients to help them check the validities of the acquired services and edge servers according to the transactions on chain for reducing the costs of these IoT devices, but also employe the efficient consortium blockchain with the proof of authority consensus engine for ensuring the high throughput and low latency of the entire system.

(3) We demonstrate the security of the proposed approach, implement a prototype based on the Ethereum project [33], and evaluate its effectiveness and efficiency in the IoT-oriented edge transparent computing environment.

The rest of this paper is organized as follows. Section 2 gives an introduction to some related work and shores up our choice of blockchain technique for protecting lightweight clients from insecure service in network computing scenarios. In Section 3, we propose a blockchain-based secure service provisioning mechanism for lightweight clients in network computing scenarios. And then, we discuss the security of the proposed approach and evaluate it in experiments in Section 4. Finally, Section 5 concludes this paper and describes possible enhancement.

2. Related Work

In this section, we introduce some existing approaches about secure service sharing mechanisms which can be applied to the network computing environments.

To defend against the threatening service codes loading from the remote servers via the open network, Kuang et al. [12] proposed a security-enhanced service sharing approach for local terminals in network computing by using the integrity checking technique. This approach deploys the checking procedures together with static hash results of services on the local firmware and checks the integrity of acquired service codes from the Internet. Therefore, the terminals are secured as any unmatched suspicious service code would be discarded without execution. Furthermore, the software engineers of Intel Cooperation [13–16] proposed a series of integrity-checking-based secure methods on the UEFI (Unified Extensible Firmware Interface) firmware collaborating with the dedicated TPM hardware. However, these security approaches become unpractical in the IoT scenarios. For one thing, the static information prestored in local device is rigid and technically less updatable, especially in IoT scenarios. For another, the requirements of extra supports of firmware or specific hardware are usually unavailable for lightweight IoT devices.

In recent years, the emergence and fast growth of blockchain technologies [18, 19] also contribute to the development of service sharing techniques and meanwhile indicate a new way to secure the local terminals from the threats of untrusted extraneous services. As a decentralized ledger built upon peer-to-peer (P2P) structure, blockchain eliminates the need of trusted third parties and has the features of decentralization, trustworthiness, and anonymity. According to the permissions of blockchain nodes, current blockchains can be divided into three types: the public blockchain (which is an open public system that can be partaken by any entities), the private blockchain (which is totally controlled by a single entity), and the consortium blockchain (which is maintained by several privileged entities with limited permissions to normal participants). As the soul of blockchain techniques, there exist several consensus algorithms; typically, the Proof of Work (PoW) is a very fair but costly hashrate-based algorithm fitting for public blockchains, while the Proof of Stake (PoS) is a stake-based algorithm, and the Proof of Authority (PoA) is an efficient and economical authority-determined algorithm often used in consortium blockchains. Besides, the smart contract is another important part of the blockchain. It is a set of promise codes that may be triggered for automatic execution when deployed on the blockchain. And the transactions of execution results will be generated and verified by all blockchain miners so that they will be appended on the blockchain trustworthily. Obviously, the booming of smart contracts makes the blockchain a functionally rich technology. Based on the outstanding features of blockchain technologies, some people started to use blockchain for content sharing. Kishigami et al. [20] proposed a digital content distribution system based on the blockchain. The content owner named licensor shares the data content with licensees over the Internet and all the transactions are recorded on the blockchain. Using blockchain for content distribution can guarantee certain security. With the support of such blockchain-based platforms, users can obtain rich services, while service developers can also control the deliveries of content-sharing services. This is the purpose of most schemes using blockchains for content distribution because a decentralized platform always gives users more freedom. Fotiou and Polyzos [21] presented a decentralized name-based security mechanism that aims to secure content distribution on the blockchain architecture. They leveraged Hierarchical Identity Based Encryption (HIBE) to solve the problem of content storage and verification. And the data content was divided into many small parts for flexible management. Similarly, Decent [22] also uses a similar method for managing data in chunks. It splits the data into multiple pieces before sending it to consumers. However, this system does not take the rationality of data into account, such as whether the data is tampered or not. These approaches that utilize the decentralization characteristics of the blockchain for service sharing ignored the importance of service reliability. Apparently, due to the lack of appropriate security mechanism, the clients are exposed to risks as they may receive unexpected malicious services. Some relevant solutions based on blockchain techniques were proposed to improve the reliability of service sharing. Xu et al. [23] proposed an integrity-checking-based blockchain approach to improve the security of data sharing. They transmitted the personal data in an off-chain manner and stored the corresponding hash value on the blockchain. However, it depends on users' own subjective judgment to decide whether the obtained services are secure or not, thereby causing many subjective controversies. Zhou et al. [24] proposed a protocol named CSSP (cleanroom security service protocol) based on the consortium blockchain to provide network software services. Instead of using the PoW algorithm, it uses an arbitration node to mediate and record transactions which saves a lot in mining. However, these approaches are

only imperfect mitigations for security problems. When it comes to our blockchain-based service provisioning scheme, it implemented an off-chain service delivery, dynamic on-chain verification mechanism, to help the lightweight IoT clients get rid of insecure services and service providers, without the participation of traditional trusted third party.

The approaches mentioned above are mainly designed for the desktop environment. When it comes to the IoT scenarios, although blockchain and smart contracts have been introduced for improving the security of IoT systems [34–36], there are few practices using blockchains for achieving secure service sharing due to the limitations on the hardware and software in IoT environments. Boudguiga et al. [25] used blockchain as a platform to provide service updates for IoT devices. There are three entities in this system: manufacturer node, user node, and innocuousness checking node. Before manufacturers providing update service for clients, the innocuousness checking nodes will download the updates from the blockchain to check the innocuousness. And then they will respond with a message indicating whether the update is problematic or innocuous. The clients will not be allowed to download the update until more than half of the checking points prove that the current update is innocuous. This approach also makes use of an arbitrator node to ensure the reliability of services. Usually, these authority-determined consensus algorithms (i.e., the PoA algorithm) are used in the consortium blockchain which is maintained by several privileged entities. There also exist some typical studies which used the PoA-based consortium blockchain in the IoT scenarios [26]. Undoubtedly, the success of the PoA-based consortium blockchain is quite inspiring.

Except for security, efficiency is another important issue for service sharing on the IoT platform. Due to the limitations of hardware resources, IoT devices are not capable of performing too many service tasks. We can refer to some effective desktop methods in the IoT scenarios. The works [21, 22] reduced the pressure on a single data transfer by delivering content in chunks. Herbaut and Negru [27] divided regions on the blockchain by smart contracts; each contract manages a part of the edge users and content providers. This approach reduces the burden of content transfer on a single service node. Sharma et al. [28] proposed an edge-cloud architecture implemented as the blockchain system for service sharing in IoT environment. In this approach, the close fog nodes are responsible for service delivery for IoT devices. And all the services are stored in the blockchain cloud, thereby achieving the low-cost service access control. A similar approach is also proposed by Dorri et al. [29]. However, these architectures did not improve the performance from the perspectives of blockchain itself as well as the consensus mechanism. On the contrary, our platform took advantage of the PoA-based blockchain which achieves the high throughput and low latency of the entire system.

In conclusion, comparing with the existing solutions, our blockchain-based service provisioning scheme implements an off-chain service delivery and dynamic on-chain verification mechanism to help the lightweight IoT clients get rid of insecure services and service providers, without the participation of traditional trusted third party. Our approach uses smart contracts to help the lightweight IoT clients check the validities of the acquired services and corresponding edge servers which significantly reduces the costs on the side of IoT devices. Besides, our system employs the efficient consortium blockchain with the PoA consensus engine which achieves the high throughput and low latency of the entire system.

3. Blockchain-Based Secure Service Provisioning System

In this section, we provide an overview of the secure service provisioning framework and then detail it in terms of its validity management and verification businesses.

3.1. Overview of the Model. The blockchain-based secure service provisioning framework builds on the edge transparent computing model and is working in on-chain and off-chain collaboration mode, as shown in Figure 2.

It consists of both the legacy entities of edge transparent computing and several new entities of blockchain system.

The Legacy Entities of Edge Transparent Computing

(i) *Cloud Service Provider* (CSP): The CSP is the powerful cloud-tier service provider which provides the trusted service codes to ESPs in an off-chain manner. There exist several CSPs which belong to different organizations in the system and each CSP may consist of several cloud servers.

(ii) *Edge Service Provider* (ESP): The ESP is the off-chain weak service provider close to the LCs. It is able to cache the service programs from the CSPs and deliver them to the LCs when requested. The ESPs and their services are not always dependable. Devices such as laptops and routers are usually acting as the ESPs in practice.

(iii) *Lightweight Client* (LC): The LC is the terminal which is eager to request and execute the service codes from the service providers. The LCs are abstractions of the physical IoT devices.

The Entities of Blockchain System

(i) *Arbitration Node* (AN): The ANs are privileged nodes in the consortium blockchain and maintain a distributed ledger together which records smart contracts and transactions of the validities of ESPs and service codes. The ANs are responsible for initiating transactions of validities, verifying the candidate block, and executing smart contracts. All the ANs work in the PoA consensus mode in which each AN packages and broadcasts new block in turn while the others vote to reach a consensus according to the plurality (more than 50%) rule. In our approach, each CSP acts as an AN in the blockchain network (the *AN*s are deployed on legacy cloud servers in practice).

(ii) *Lightweight Node* (LN): The LN is the less privileged entity which is only allowed to read the information on the blockchain and trigger the smart contract

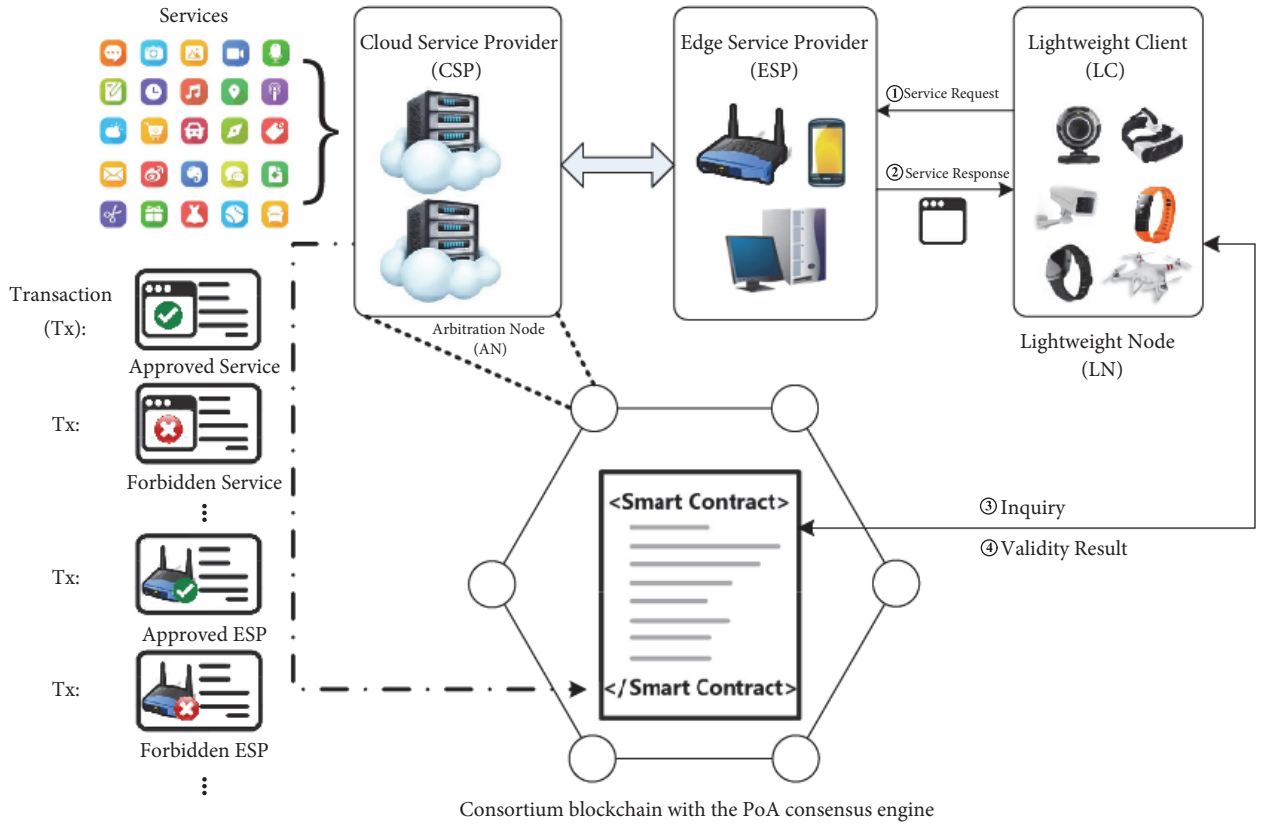


Figure 2: The secure service provisioning in edge transparent computing.

to query the validity states about the ESPs or the acquired service codes. Each LC is also an LN in our blockchain network.

In this framework, the LCs mainly request and obtain service codes from close ESPs. For protecting the LCs from untrusted ESPs and undependable services, they are allowed to trigger the smart contract (oracle smart contract, SC_O) deployed on the blockchain to figure out the validity states of the current serving ESPs or the acquired service codes according to existing validity transactions, so that the LCs can determine whether to execute the service programs or not. Besides, to keep the validity states of the ESPs and service codes updated, the ANs would continually append the new transactions of validity into the blockchain.

In our approach, the service business is off-chain while the security business is on-chain, which helps to achieve the security with low-performance overheads. Additionally, the blockchain is implemented as a consortium blockchain with the PoA consensus engine for performance reasons as well.

Next, we describe the major businesses of our approach, namely, the validity maintenance and verification in detail.

3.2. Validity Maintenance Business. The ANs keep the validity states of the ESPs and service codes updated by continually appending the new transactions of validity into the blockchain. The transaction structures are given as follows:

(i) The validity transaction of ESP: $Tx_E = \langle ESP_{ID}; V; C; T\rangle$,

where ESP_{ID} is the MAC address of the ESP, V is the validity state, C is the comments, and T is the timestamp.

(ii) The validity transaction of service: $Tx_S = \langle S_{NAME}; S_{HASH}; V; C; T\rangle$,

where S_{NAME} is the service name, $S_{HASH} = hash_{Keccak\text{-}256}$ (service codes), V is the validity state, C is the comments, and T is the timestamp.

By appending new Tx_E and Tx_S with corresponding validity states to the blockchain, the ANs can declare new legal ESPs and service programs, discard the existing ESPs and service programs when necessary (e.g., bugs discovered), update the service version, or even declare malicious ESPs and service programs. In addition, all the transactions and corresponding addresses will also be stored into the public database of a maintainer smart contract (SC_M) synchronously to make the transactions efficiently searchable for the smart contract SC_O (the on-chain address of SC_M is embedded in SC_O).

Notice that we reasonably assume that all the *ANs* can obtain service codes and necessary information about *ESPs* which are engaged in the transactions; meanwhile, the *ESP* authentication and service security testing are out of the scope of this work. Besides, the MAC address-based identification used in our case is an exemplary method

which can be replaced or combined with other identification mechanisms. And defending against identification spoofing attacks such as MAC spoofing are complementary to our work.

3.3. Validity Verification Business. For security purpose, when LCs request and obtain service codes from ESPs, they can trigger the oracle smart contract (SC_O) with the corresponding indices of the current ESPs or the service codes, to query the corresponding validity states. The workflow of the secure service provisioning is as follows:

(1) *LC* initiates an off-chain service request $\langle S_{NAME} \rangle$ to a close ESP.

(2) The ESP returns service codes to the *LC* in an off-chain manner.

(3) *LC* (i.e., *LN*) calculates the hash value of the service codes by the Keccak-256 algorithm and then triggers SC_O with a vector $\langle S_{NAME};\ S_{HASH};\ ESP_{ID} \rangle$.

(4) SC_O invokes SC_M for corresponding on-chain records about $\langle S_{NAME} \rangle$ and $\langle ESP_{ID} \rangle$ and then compares them with the received ones. Matching a valid record means the corresponding *ESP* or service codes are secure while matching an invalid record means the opposite. Note that an invalid result will also be given if there is no record related to $\langle S_{NAME} \rangle$ or $\langle ESP_{ID} \rangle$. Finally, SC_O outputs the result in the form of the on-chain transaction so that *LC* can make decisions accordingly.

The process of validity verification is shown in Figure 3.

3.4. Case Study. This section demonstrates an example of our approach to help people understand how it works concretely.

Assume there exists a blockchain-based secure service provisioning system which includes the following validity transactions in the blockchain:

$$\mathrm{Tx}_{E1} = \langle \textit{00-50-56-C0-00-08};\ 1;\ \text{Legitimate node};\ \textit{1539450834} \rangle,$$
$$\mathrm{Tx}_{S1} = \langle \text{Servie\_1.3};\ \text{7d7b084c0e330d734986a3a5884ad2c2af23a72e90ea06e8691849c64bbc64f9};\ 0;\ \text{Legitimate service};\ 1539454312 \rangle. \quad (1)$$

Then, we assume that *LC* initiates an off-chain service request <Servie_1.3> to a close ESP. Then, the ESP returns service codes to *LC* in an off-chain manner. After receiving the service codes from the ESP, the *LC* calculates the hash value of the service codes by Keccak-256 algorithm and then triggers SC_O with a vector before execution:

$$\langle \text{Servie\_1.3};\ \text{7d7b084c0e330d734986a3a5884ad2c2af23a72e90ea06e8691849c64bbc64f9};\ \textit{00-50-56-C0-00-08} \rangle. \quad (2)$$

SC_O invokes SC_M for the latest on-chain records about <Servie_1.3> and <*00-50-56-C0-00-08*> and then compares them with the received vector from the *LC*. Since the valid records Tx_{E1} and Tx_{S1} are found in the blockchain, SC_O outputs the valid result in the form of on-chain transaction (*cf*. Figures 4(a) and 5(a)). Finally, the *LC* finds the result transaction and believes that corresponding *ESP* and service codes are secure.

On the contrary, if the vector submitted to SC_O from the *LC* is <Servie_1.3; 56608f2ed0cdcf51ba6a99b2718aab4d2-e74ff78acdfa64ee8290 37be50b2cef; *E0-94-67-D4-1C-7D*>, SC_O outputs the invalid result in the form of on-chain transaction (*cf*. Figures 4(c) and 5(b)) because no valid record can be found in the blockchain. Therefore, the *LC* finds this result transaction and then denies the service codes from the unreliable *ESP*.

4. Analysis and Evaluation

This section demonstrates the security of our approach and then analyzes the experimental results in terms of effectiveness and efficiency.

4.1. Security Analysis

4.1.1. Threat Model. We assume that the adversary can set illegal ESPs to provide malicious or vulnerable service codes for attacking clients. Besides, the benign ESPs may also provide illegal services, e.g., outdated unpatched codes, due to the improper maintenance, thereby putting clients at risks. However, the adversary can neither compromise the majority of arbitration nodes to tamper the blockchain system nor forge digital signatures without corresponding private keys, which is the basic security assumption of general blockchain network commonly accepted. Note that defending against identification spoofing attacks such as MAC spoofing attacks on *ESP*s are out of the scope of this work as the MAC-based identification used in this approach is only an exemplary method which can be replaced or combined with other advanced mechanisms.

4.1.2. Analysis. Since the *ESP*s and service codes are not always reliable, our security mechanism makes use of the smart contract to check the latest validity states of edge

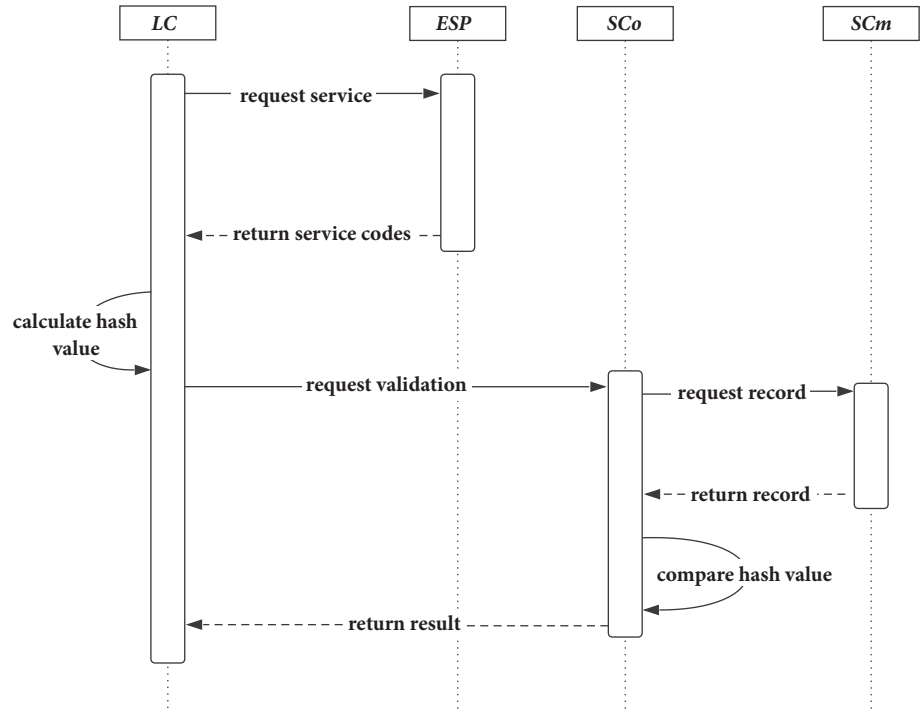
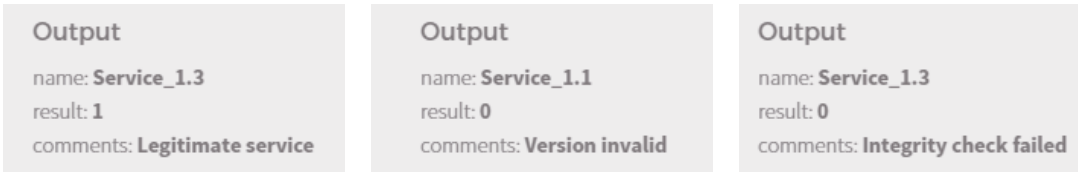


Figure 3: The workflow of validity verification.



(a) Legal service (b) Invalid version (c) Illegal service

Figure 4: The checking results of service given by the smart contract.

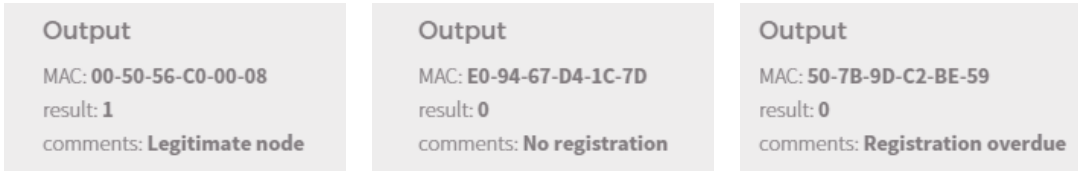
servers and service codes recorded in the form of transactions on the blockchain, so as to help the lightweight clients get rid of illegal service providers and avoid running the unknown or discarded service codes, thereby mitigating the risks. Obviously, the security of our approach mainly depends on the correctness of validity transactions and the proper executions of smart contracts.

For the validity transactions, since every transaction is publicly checked and maintained by all the distributed arbitration nodes, according to the basic security assumption of blockchain network, it is almost impossible to tamper existing transactions in blocks or package incorrect transactions into new blocks because, in the PoA consensus mechanism, the adversary can hardly compromise the majority of arbitrators (more than 50% *AN*s), which are deployed on well-maintained cloud servers. Besides, with the aid of the digital signature technique integrated into the blockchain, the adversary is unable to add malicious transactions with forged digital signatures of legal arbitration nodes because the adversary does not have corresponding private keys. Therefore, the validity transactions are trustworthy in our approach.

When it comes to the smart contracts, just like the ordinary transactions in the blockchain, they are also publicly verified and will be executed by all the arbitration nodes. Since the codes of the smart contracts are designed to be immutable, they cannot be modified after deployment even by the creators. Besides, all the execution results given by smart contracts are verified and packaged as transactions within the blocks by all arbitration nodes; therefore, these results are tamper-proof as well.

Consequently, according to the analysis above, the validity transactions of service codes and *ESP*s are trusted, and the smart contracts would be executed correctly. Therefore, the IoT clients can obtain the trusted results for security decisions, thereby getting rid of illegal service providers and insecure service codes effectively.

Additionally, as a blockchain-based approach, our security facilitates work in a decentralized P2P manner without relying on a single trusted third party and thus is more robust



(a) Legal *ESP* (b) Illegal *ESP* (c) Revoked *ESP*

Figure 5: The checking results of the *ESP*s given by the smart contract.

Table 1: The specifications of the testing devices.

Parameter	Cloud (arbitration) node (Virtual cloud server)	Edge node (Virtual laptop)	IoT node (Virtual IoT client)
CPU frequency	3.4 GHz	2.6 GHz	512 MHz
CPU core	Quad-core	Dual-core	Single-core
Network	100 Mbps	100 Mbps	100 Mbps
RAM	16 G	8 G	256 M
ROM	1 T	512 G	4 G
OS	CentOS 6.0	Fedora 12	Ubuntu Mate

against security problems like the single point of failure which can be caused by distributed denial of service attacks that often happen in IoT scenarios [17].

4.2. Experimental Evaluations. In this section, we conducted experiments to evaluate the effectiveness and efficiency of our system. We simulated cloud nodes on a single physical machine; each of them acts as an arbitration node in the consortium blockchain, and they have the highest power as miners. Also, we simulated several edge nodes to serve client nodes. There also exists virtual IoT client to request service. The details of the arbitration node, edge service node, and client node are listed in Table 1.

We use the *Ethereum Geth_1.8.11* which supports the PoA consensus mechanism to implement the consortium blockchain-based approach.

4.2.1. Effectiveness. To test the effectiveness, we simulated 20 edge nodes and 6 of them are set to provide the wrong service to the clients. Besides, we simulated 10 arbitration nodes on the consortium blockchain. When receiving the service codes, the client will calculate the corresponding hash value and then submit the result together with the identification of the edge node to the oracle smart contract SC_O for checking. Then, SC_O will query the corresponding service hash in SC_M. Finally, it puts the result on the block which can be referred by the *LC*. Therefore, the IoT device can decide to abandon the service or start to use it. For visualization purpose, we use Ethereum-Wallet's graphical interface to show the feedback from the smart contract, and Figure 4 consists of the screenshots of corresponding information of the service. As we can see, SC_O returns the query result with the help of SC_M, which includes the service name (version), the check result, and the comments. Figure 4(a) shows that the service is a legitimate service, because the hash value of the service is consistent with what the IoT device provides. And the contract will return "1" to confirm the legitimacy of the service. Figure 4(b) shows that the version of the service is invalid, and the IoT device finds "0" as the result. The edge node may not be malicious, but it has not updated the service so that the contract will identify this service as an expired service. Figure 4(c) indicates that the service provided by the edge node is completely unreliable because the integrity checking failed, and the data being transmitted is likely to be malicious and must be deprecated by the IoT node.

The contract will also check the information of the edge node. Figure 5(a) shows that the edge node is legitimate, and the result is "1." And we can see that the MAC address is also recorded. Figure 5(b) shows an illegal edge node that has not been registered on the blockchain. If the registration of an edge node is past due, it cannot be accepted and Figure 5(c) shows such information in this case.

According to the results of the experiments, our smart contracts correctly record and send back the details of the *ESP*s and service codes. The system is considered effective and the security is assured.

4.2.2. Performance. To measure the performance of our system, we conducted comparative experiments on the blockchain using the consensus algorithm of the PoA and the PoW (with the mining difficulty 0x131072), respectively. We simulated 100 IoT devices to request services from edge servers. To be more practical, we enforced the IoT clients following the Poisson probability distribution ($\lambda = 0.2 \times n$) to initiate requests. There are 20 edge nodes to provide 87 kinds of services with corresponding information recorded in the blockchain. We also simulated 10 cloud servers (i.e., arbitration nodes) whose major tasks are mining blocks and updating the validity states of various services. Through the experiment, we recorded experimental results to show the

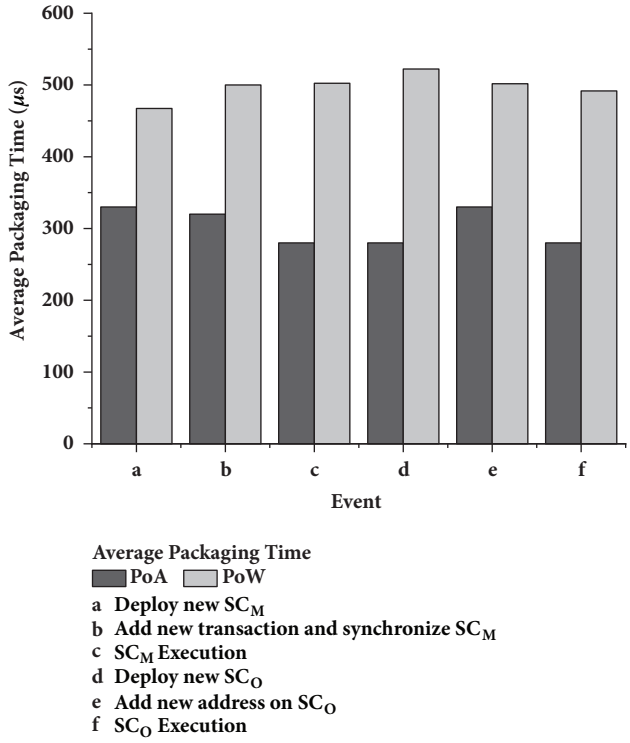


Figure 6: The average packaging time of each transaction.

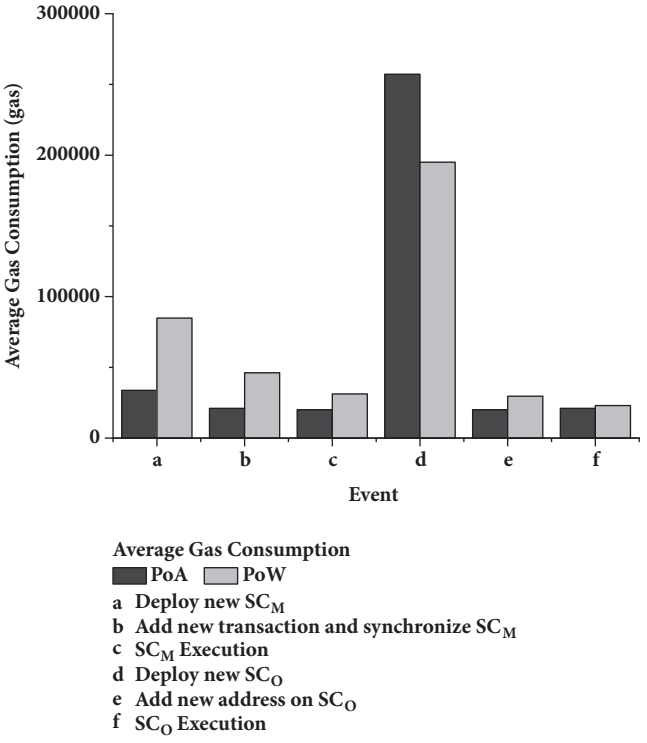


Figure 8: The average gas cost of each process.

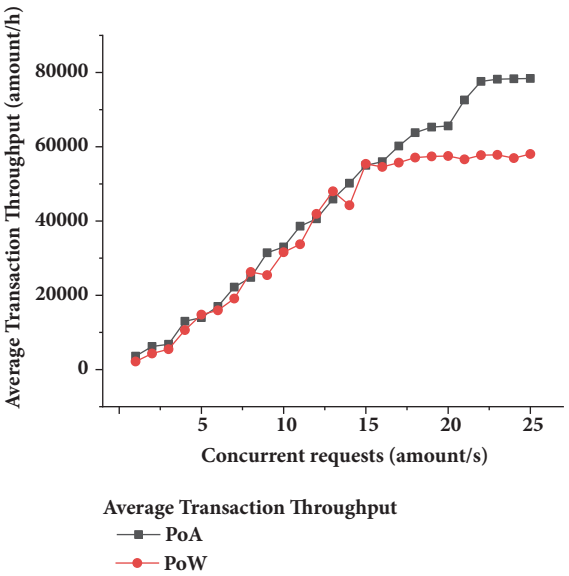


Figure 7: The throughput of transactions.

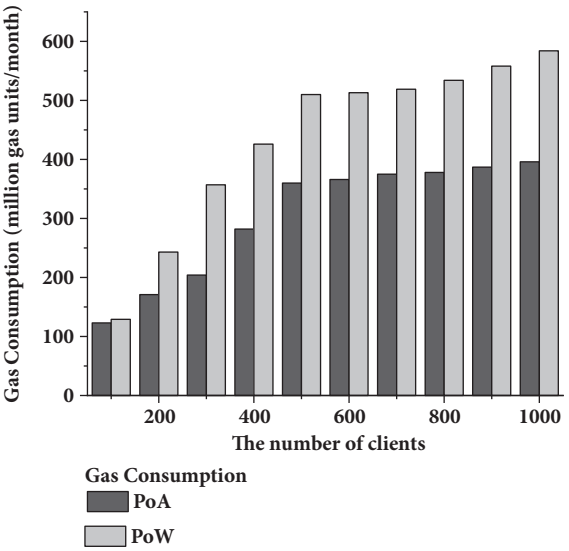


Figure 9: The gas cost under the different numbers of *LC*s in one month.

performance of the system in terms of system delays (see Figure 6), throughput (see Figure 7), and gas consumption (see Figures 8 and 9).

The packaging time is a measure of the speed of block output; to some extent, it determines how fast the system can complete transactions. As shown in Figure 6, the packaging time delay for each event is almost the same (about 500us when using the PoW and 300us when using the PoA). We can see that the packaging time under the PoA mechanism is only about 60% of that under the PoW, which shows the benefit of the adoption of the PoA in our approach. In the case of large scale throughout, the platform based on the PoA mechanism will have better performance. The experiment result shows that, in general, the packaging time for each event is small and acceptable. In particular, it is negligible to the clients in the service provisioning process.

We also conducted an experiment to measure system throughput by studying the relationship between concurrent service requests and transaction output speed. As shown in Figure 7, below a certain amount of concurrent service

requests, given a fixed period of time, the output speed of transactions increases at a certain rate along with the increase of concurrent service requests. But when the amount of concurrent requests is over 22 per second, the curve starts to converge, and the output speed of the transaction gradually tends to a stable value. The system throughput cannot increase indefinitely because it is limited by the speed of blocks creations and the capability of each block. And our maximum transaction throughput is approximately 80000 per hour. At its best, the platform can complete about 80000 transactions per hour, i.e., about 22 transactions per second, which is a relatively high and stable throughput large enough for service business. When it comes to the PoW-based approach, the transaction throughout curve converges earlier (since the concurrent request amount is 16 per second) and the maximum throughput is about 55000 per hour, which is obviously inferior to our PoA-based approach.

In the Ethereum-based blockchain, each mining node (i.e., *AN*s in our case) participating in the network will perform the blockchain protocol. With the creation of a transaction, a certain amount of gas will be charged. The gas price is the unit price of gas (e.g., 1 ether ≈ 210 USD) set by the initiator of the transaction, and the total cost of the transaction is cost (ether) = gas × gas price. Therefore, we also logged the average gas consumption of all the events in our approach, seen as shown in Figure 8. When using the PoA mechanism, event (a) costs about 33800 gas units. Events (b), (c), (e), and (f) cost about 21000 gas units per transaction. Event (d) costs around 257000 gas units. Similar results were observed from the experiment using the PoW mechanism. As we can see, the gas usage of each execution event is almost the same. But the deployment of SC_O costs much gas units than other events. We believe the reason is that the codes of SC_O are more complex than those of the others. Besides, as for the setup events, events (a) and (d) only happen once in the initialization while events (b) and (e) occur when new validity state of service codes or an *ESP* is appended. The gas price in this experiment is 0.02 ether per million gas units; therefore, it costs about 0.0058 ethers (≈ 1.2 USD) to deploy the smart contracts which are necessary to make our system functional. And as for normal transactions, we can record 1000 transactions with the cost of only 0.042 ethers (≈ 8.8 USD). We can see that the cost of our platform is relatively small and acceptable.

Besides, we also simulated n ($n = 100, 200, 300, \ldots, 1000$) IoT devices to request services from edge servers within a month and recorded the gas consumption. We also enforced the IoT devices following the Poisson probability distribution ($\lambda = 0.2 \times n$) to initiate requests. In Figure 9, the x-coordinate represents the number of *LC*s increasing from 100 to 1000 with the increment of 100. The y-coordinate represents the monthly gas consumption in the certain number of *LC*s. As shown in Figure 9, when the number of clients is 100, the monthly gas consumption is about 125 million gas units (≈ 5.25 USD per client) under the PoA, compared with 130 million gas units (≈ 5.46 USD per client) under the PoW. We can see that the gas consumption maintains a slow and stable growth along with the increase of clients. When the number of clients is 1000, the monthly gas consumption of the system under the PoA consensus mechanism is about 400 million gas units (≈ 1.68 USD per client), compared with 580 million gas units (≈ 2.436 USD per client) in the PoW-based one. Besides, the gas consumption of both the PoA and PoW relies on a stable growth, which means adding IoT devices does not impose a great overhead; therefore, our system has a low consumption and sufficient scalability.

5. Conclusions

In this paper, we proposed a novel blockchain-based secure service provisioning mechanism to protect the lightweight clients from insecure exogenous service codes from untrustworthy edge servers in the edge transparent computing scenario. We introduce the blockchain to keep all the validity states of the off-chain services and edge service providers for helping the IoT terminals get rid of undependable services through edge servers' identification and service verification. Besides, we develop and deploy the smart contracts that can be triggered by the lightweight clients to check the validities of both the service codes and edge servers according to the transactions on chain, thereby reducing the direct overheads of these resource-constrained IoT devices. Additionally, to ensure the high throughput and low latency, we adopt the efficient permissioned blockchain together with the PoA consensus engine. The security analysis and the evaluation results show that our approach protects the lightweight clients from untrusted edge service providers and undependable service codes effectively, and the validation latency is acceptable while the overheads are affordable to IoT devices.

Next, we would like to establish a blockchain-based reputation system for the service providers according to the feedbacks from IoT terminals, so as to achieve a better trade-off among flexibility, availability, and security of service provisioning. Besides, service auditing and charging are also interesting issues that can be further studied.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China [Grant nos. 61632009, 61702561, 61702562, and 61472451], the Hunan Provincial Innovation Foundation for Postgraduate [Grant no. CX2015B047], and the Guangdong Provincial Natural Science Foundation [Grant no. 2017A030308006].

References

[1] C. MacGillivray and P. Gorman, "Connecting the IoT: The Road to Success," International Data Corporation (IDC) Report, 2018.

[2] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog Computing: A Taxonomy, Survey and Future Directions," in *Internet of Everything*, Internet of Things, pp. 103–130, Springer Singapore, Singapore, 2018.

[3] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[4] Y. Zhang, K. Guo, J. Ren et al., "Transparent Computing: A Promising Network Computing Paradigm," *Computing in Science & Engineering*, vol. 19, no. 1, pp. 7–20, 2017.

[5] J. He, Y. Zhang, J. Lu, M. Wu, and F. Huang, "Block-Stream as a Service: A More Secure, Nimble, and Dynamically Balanced Cloud Service Model for Ambient Computing," *IEEE Network*, vol. 32, no. 1, pp. 126–132, 2018.

[6] J. Ren, H. Guo, C. Xu, and Y. Zhang, "Serving at the Edge: A Scalable IoT Architecture Based on Transparent Computing," *IEEE Network*, vol. 31, no. 5, pp. 96–105, 2017.

[7] H. Guo, J. Ren, D. Zhang, Y. Zhang, and J. Hu, "A scalable and manageable IoT architecture based on transparent computing," *Journal of Parallel and Distributed Computing*, vol. 118, no. 1, pp. 5–13, 2017.

[8] X. Peng, J. Ren, L. She, D. Zhang, J. Li, and Y. Zhang, "BOAT: A Block-Streaming App Execution Scheme for Lightweight IoT Devices," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1816–1829, 2018.

[9] W. Li, B. Wang, J. Sheng, K. Dong, Z. Li, and Y. Hu, "A Resource Service Model in the Industrial IoT System Based on Transparent Computing," *Sensors*, vol. 18, no. 4, pp. 981–1022, 2018.

[10] Y. Zhang, L. T. Yang, Y. Zhou, and W. Kuang, "Information security underlying transparent computing: Impacts, visions and challenges," *Web Intelligence and Agent Systems*, vol. 8, no. 2, pp. 203–217, 2010.

[11] G. Wang, Q. Liu, Y. Xiang, and J. Chen, "Security from the transparent computing aspect," in *Proceedings of the 2014 International Conference on Computing, Networking and Communications, ICNC 2014*, pp. 216–220, USA, February 2014.

[12] W. Kuang, Y. Zhang, Y. Zhou, and H. Yang, "RBIS: Security Enhancement for MRBP and MRBP2 Using Integrity Check," *Journal of Chinese Computer Systems*, vol. 28, no. 02, pp. 251–254, 2007.

[13] M. Wu, "Analysis and a Case Study of Transparent Computing Implementation with UEFI," *International Journal of Cloud Computing*, vol. 1, no. 4, pp. 312–328, 2012.

[14] M. Wu, "How to Make Transparent Computing Secure – Several Security Considerations in Transparent Computing Design and Implementation," in *Proceedings of the Workshop on Trusted Computing (Guangzhou) Presentation*, 2018.

[15] V. J. Zimmer and D. Wei, "UEFI Technical Updates and Platform Innovations," in *Proceedings of the Transparent Computing Summit (Shanghai) Presentation*, 2010.

[16] V. J. Zimmer, "Platform trust beyond BIOS using the Unified Extensible Firmware Interface," in *Proceedings of the 2007 International Conference on Security and Management, SAM'07*, pp. 400–405, USA, June 2007.

[17] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: mirai and other botnets," *IEEE Computer Society*, vol. 50, no. 7, pp. 80–84, 2017.

[18] S. Nakamoto, *Bitcoin: A peer-to-peer electronic cash system*, 2008.

[19] Z. Zheng, S. Xie, H. Dai et al., "Blockchain Challenges and Opportunities: A Survey," *International Journal of Web & Grid Services*, 2016.

[20] J. Kishigami, S. Fujimura, H. Watanabe, A. Nakadaira, and A. Akutsu, "The Blockchain-Based Digital Content Distribution System," in *Proceedings of the 5th IEEE International Conference on Big Data and Cloud Computing, BDCloud 2015*, pp. 187–190, China, August 2015.

[21] N. Fotiou and G. C. Polyzos, "Decentralized name-based security for content distribution using blockchains," in *Proceedings of the 35th IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS 2016*, pp. 415–420, USA, April 2016.

[22] M. Michalko and J. Sevcik, *DECENT Whitepaper*, DECENT Foundation Documentation, 2015.

[23] X. Xu, C. Pautasso, L. Zhu et al., "The blockchain as a software connector," in *Proceedings of the 13th Working IEEE/IFIP Conference on Software Architecture, WICSA 2016*, pp. 182–191, Italy, April 2016.

[24] L. Zhou, G. Wang, T. Cui, and X. Xing, "Cssp: The Consortium Blockchain Model for Improving the Trustworthiness of Network Software Services," in *Proceedings of the 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC)*, pp. 101–107, Guangzhou, December 2017.

[25] A. Boudguiga, N. Bouzerna, L. Granboulan et al., "Towards better availability and accountability for IoT updates by means of a blockchain," in *Proceedings of the 2nd IEEE European Symposium on Security and Privacy Workshops, EuroS and PW 2017*, pp. 50–58, France, April 2017.

[26] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, "Consortium Blockchain for Secure Energy Trading in Industrial Internet of Things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3690–3700, 2018.

[27] N. Herbaut and N. Negru, "A Model for Collaborative Blockchain-Based Video Delivery Relying on Advanced Network Services Chains," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 70–76, 2017.

[28] P. K. Sharma, M.-Y. Chen, and J. H. Park, "A Software Defined Fog Node Based Distributed Blockchain Cloud Architecture for IoT," *IEEE Access*, vol. 6, pp. 115–124, 2018.

[29] A. Dorri, S. S. Kanhere, and R. Jurdak, "Blockchain in Internet of Things: Challenges and Solutions," https://arxiv.org/abs/1608.05187.

[30] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proceedings of the 23rd ACM Conference on Computer and Communications Security, CCS 2016*, pp. 254–269, Austria, October 2016.

[31] X. Li, P. Jiang, T. Chen, X. Luo, and Q. Wen, "A survey on the security of blockchain systems," *Future Generation Computer Systems*, In press, corrected proof, Available online 23 August 2017.

[32] S. D. Angelis, L. Aniello, R. Baldoni et al., "PBFT vs Proof-of-authority: Applying the CAP Theorem to Permissioned Blockchai," in *Proceedings of the Italian Conference on Cybersecurity*, pp. 1–11, 2018.

[33] https://www.ethereum.org/.

[34] E. F. Jesus, V. R. L. Chicarino, C. V. N. de Albuquerque, and A. A. Rocha, "A Survey of How to Use Blockchain to Secure Internet of Things and the Stalker Attack," *Security and Communication Networks*, vol. 2018, Article ID 9675050, 27 pages, 2018.

[35] H. Shafagh, L. Burkhalter, A. Hithnawi, and S. Duquennoy, "Towards blockchain-based auditable storage and sharing of iot data," in *Proceedings of the 8th ACM Cloud Computing Security Workshop, CCSW 2017*, pp. 45–50, ACM, TX, USA.

[36] K. Christidis and M. Devetsikiotis, "Blockchains and Smart Contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 8646973, 11 pages
https://doi.org/10.1155/2018/8646973

Research Article

On the RCCA Security of Hybrid Signcryption for Internet of Things

Honglong Dai,[1] Ding Wang ,[1,2] Jinyong Chang ,[1] and Maozhi Xu[1]

[1]*Peking University, Beijing 100871, China*
[2]*State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China*

Correspondence should be addressed to Ding Wang; wangdingg@pku.edu.cn

Received 31 August 2018; Accepted 28 October 2018; Published 12 November 2018

Guest Editor: Weizhi Meng

With the rapid development of the Internet of Things (IoT), a lot of sensitive information in our daily lives are now digitalized and open to remote access. The provision of security and privacy of such data would incur comprehensive cryptographic services and has raised wide concern. Hybrid signcryption schemes could achieve various kinds of cryptographic services (e.g., confidentiality, authenticity, and integrity) with much lower cost than the combination of separate traditional cryptographic schemes with each providing a single cryptographic service. Thus, hybrid signcryption schemes are very suitable for IoT environments where resources are generally very constrained (e.g., lightweight sensors and mobile phones). To ensure that the overall hybrid signcryption scheme provides adequate cryptographic service (e.g., confidentiality, integrity, and authentication), its parts of KEM (key encryption mechanism) and DEM (data encryption mechanism) must satisfy some security requirements. Chosen-ciphertext attack (CCA) security has been widely accepted as the golden standard requirement for general encryption schemes. However, CCA security appears too strong in some conditions. Accordingly, Canetti et al. (CRYPTO 2003) proposed the notion of replayable CCA security (RCCA) for encryption schemes, which is a strictly weaker security notion than CCA security and naturally more efficient. This new security notion has proved to be sufficient for most existing applications of CCA security, e.g., encrypted password authentication. This is particularly promising for IoT environments, where security is demanding, yet resources are constrained. In this paper, we examine the RCCA security of the well-known SKEM+DEM style hybrid signcryption scheme by Dent at ISC 2005. Meanwhile, we also examine the RCCA security of the Tag-SKEM+DEM style hybrid signcryption scheme by Bjorstad and Dent at PKC 2006. We rigorously prove that a hybrid signcryption scheme can achieve RCCA security if both its SKEM part and its DEM part satisfy some security assumptions.

1. Introduction

With the booming development of wireless technology, Internet of Things (IoT) has seen its proliferation in various applications such as personal health, government work, and battle surveillance. How to ensure security and privacy of the sensitive data in these security-critical applications is a challenging issue, because it would generally incur comprehensive cryptographic services. Hybrid signcryption schemes could achieve various kinds of cryptographic services (e.g., confidentiality, authenticity, and integrity) with much lower cost than the combination of separate traditional cryptographic schemes with each providing a single cryptographic service [1]. Thus, hybrid signcryption schemes are very suitable for IoT environments where resources are generally very constrained (e.g., lightweight sensors and mobile phones).

The first signcryption scheme was proposed by Zheng [2] at CRYPTO'97. The notion of confidentiality for a signcryption scheme is analogous to an original encryption scheme, while the nonrepudiation service is analogous to a digital signature one [3]. Since then, various kinds of signcryption schemes have been suggested. At 2002, Lee [4] proposed identity-based signcryption; At AsiaCCS'08, Barbosa et al. [5] proposed certificateless signcryption. At IMACC'13, Nakano et al. [6] presented two generic constructions of signcryption in the standard model. At 2017, Li et al. [7] proposed a signcryption for cloud computing. At PQCrypto'18 Sato et

al. [8] proposed lattice-based signcryption without random oracles. At the same time, Datta et al. [9] proposed the functional signcryption.

In addition, a number of signcryption schemes have been proposed for the IoT environments (e.g., key establishment over ATM networks [10], defense against fragment duplication attack in 6LoWPAN networks [11], short signcryption scheme for IoT [12], and provably secure signcryption for IoT [13]). Belguith et al. [14] proposed privacy preserving attribute based signcryption for IoT.

However, in the traditional signcryption schemes, the keyed encapsulation encryption is generally not made full use of, and the length of messages is always related to the signcryption scheme. Further, the major weakness of asymmetric encryption schemes is that the computational efficiency is worse than these symmetric ones [15]. Accordingly, the notion of hybrid signcryption is proposed. Hybrid signcryption uses a symmetric encryption scheme to improve the overall performance and flexibility of asymmetric signcryption. Hybrid signcryption can simultaneously combine the main advantages of a public key encryption and a digital signature scheme with much lower cost when compared with traditional schemes [1, 16]. As sensor nodes in IoT are resource-constrained (e.g., limited battery power) and deployed to run for years, hybrid cryptography is particularly suitable for data storage and transmission to achieve secure and efficient communication [17]. At 2004, Dent [15] proposed a formal composition model for hybrid signcryption, and this model covers Zheng's scheme [2]. Later, Bjorstad et al. [18] proposed an improve signcryption scheme with tag-KEMs, Li et al. [19] proposed a certificateless hybrid signcryption scheme, and Zhou [20] proposed an improved certificateless hybrid signcryption scheme. Due to the usage of a symmetric encryption scheme to overcome the weakness and restricted message space of traditional asymmetric encryption schemes, these hybrid signcryption schemes can make the length of message independent of the security of the overall signcryption scheme.

Secure encryption is one of the most fundamental tasks in cryptographic schemes, while CCA security has been widely accepted as the golden standard requirement for encryption schemes [21, 22]. However, chosen-ciphertext attack security appears to be too strong in many conditions; there exist many encryption schemes that are not CCA secure but still have practical applications [23]. Here we take a CCA secure public key encryption scheme PKE as example. We change it into a public key encryption scheme PKE', which is equal to public key encryption scheme PKE except that this encryption oracle algorithm appends a bit 0 to each ciphertext and the decryption oracle algorithm of PKE' discards this bit 0 of a ciphertext. Then, one naturally obtains a different ciphertext decrypted to the same message as the original one. However, this change takes no real consequence in most situation, because the modified scheme PKE′ appears to be as secure as the scheme PKE in most situations. This example is also used in [23].

Accordingly, Canetti et al. [23] proposed the RCCA security notion at CRYPTO 2003. RCCA security is a strictly weaker security notion than CCA security, which has proved to be abundant for most cryptographic primitives, e.g., encrypted password authentication [24]. There are some studies (e.g., [23, 25]) about the RCCA security of hybrid cryptography, and there are also several studies (e.g., [2, 3, 15]) about the CCA security of hybrid signcryption. As far as we know, there is no work about examining the RCCA security of hybrid signcryption. To fill the gap, in this paper we consider the RCCA security of hybrid signcryption and show that hybrid signcryption can achieve RCCA security (rather than only CCA security) based on certain conditions.

1.1. Main Contributions. In this paper, we examine the RCCA security of the hybrid signcryption scheme Tag-SKEM+DEM [18] and the hybrid signcryption scheme SKEM+DEM [3]. We will show the following: (1) The hybrid signcryption scheme (SKEM+DEM) can be RCCA-secure if the scheme Tag-SKEM is RCCA-secure and the scheme DEM is RCCA-secure. (2) The hybrid encryption scheme (Tag-SKEM+DEM) can be RCCA-secure if the signcryption scheme Tag-SKEM is RCCA-secure and the scheme DEM is RCCA-secure. Although our results might be expected and somewhat straightforward, we concretely confirm such expectations with a formal proof. When giving our proof, we mainly use the hybrid game-based reduction technique presented in [26–28].

1.2. Related Works and Discussions. It is obvious that if the hybrid signcryption scheme is going to provide an integrity and authentication service, then its KEM part and DEM part must satisfy some kind of security criterion. Dent et al. [15] examined the CCA security of hybrid signcryption schemes (SKEM+DEM and Tag-SKEM+DEM). Chen et al. [27] examined the RCCA security for hybrid encryption scheme KEM+DEM. Cui et al. [29] gave two kinds of RKA-secure signcryption schemes. In 2017, Dai et al. [30] considered the ECCA security for hybrid encryptions Tag-KEM+DEM and KEM+Tag-DEM. Abe et al. [26] provided a hybrid encryption scheme Tag-KEM+DEM, and they presented a useful way to get CCA secure hybrid encryptions. Cramer et al. [31] have shown that the hybrid encryption scheme Tag-KEM+DEM is CCA secure if its KEM part is CCA secure and its DEM part is one-time secure.

As, for the scheme Tag-SKEM+DEM, the ciphertext of scheme DEM is a tag of the scheme Tag-SKEM, one may think that the security assumption of scheme SKEM could be weakened to chosen plaintext attack (CPA) security when considering the RCCA security of signcryption. As it is impossible to make a simulation for the decryption oracle query for an adversary when the adversary attacks the hybrid signcryption, we leave it as an open problem that the security of scheme SKEM and DEM could be relaxed to a weaker security (e.g., CPA). One may also think that, with the RCCA security of Tag-KEM and one-time security of DEM, one can get the RCCA security of hybrid signcryption scheme Tag-KEM+DEM. However, the adversary cannot generate useful challenge ciphers if the adversary does not change the tag used for the scheme Tag-DEM. In this paper, when proving our results, we make a perfect simulation for the adversary, who initiates a IND-RCCA experiment to hybrid

Table 1: The hybrid cryptology and their security notion*.

Hybrid cryptology	Security notion	Reference
KEM+DEM	RCCA+RCCA⟹RCCA	[27]
KEM+DEM	CCA+CCA⟹CCA	[31]
SKEM+DEM	RCCA+RCCA⟹RCCA	Section 3.2
SKEM+DEM	CCA+CCA⟹CCA	[15]
Tag-KEM+DEM	CCA+one time security ⟹CCA	[32]
Tag-KEM+DEM	RCCA+RCCA⟹RCCA	[27]
Tag-SKEM+DEM	CCA+CCA⟹CCA	[15]
Tag-SKEM+DEM	RCCA+RCCA⟹RCCA	Section 3.4

*KEM: key encapsulation mechanism, DEM: data encapsulation mechanism.

signcryption. We summarise the hybrid cryptology and their security in Table 1.

Organizations of the Paper. In Section 2, we review some basic notations and definitions. In Section 3 we review the definition of general hybrid signcryption scheme, SKEM + DEM and Tag–SKEM + DEM, and then we prove its RCCA security. In Section 4, we review our main conclusions.

2. Preliminaries

In this section, we will review some useful notations and cryptographic primitives that will be used throughout this paper.

Notations. We denote by 1^λ the security parameter and write $m \xleftarrow{R} M$ to denote the algorithm that picks an m randomly from the set M. PPT denotes probabilistic polynomial time. we write $z \longleftarrow A(x, y, \cdots)$ to denote the algorithm that runs algorithm $\mathcal{A}$ with inputs $(x, y, \cdots)$ and then outputs z. We define a function negl(λ) as *negligible*: if for any constant $c > 0$, there exits a $k_0 \in \mathbb{Z}$, such that for all $\lambda > k_0$, negl(λ) $< \lambda^{-c}$.

2.1. RCCA Security Definition. PKE = (Gen, Enc, Dec) is a public key encryption (PKE) scheme that consists of three polynomial-time algorithms:

(i) Gen is key generation algorithm that inputs the security parameter λ and outputs a pair of public/private keys (pk, sk).

(ii) Enc is PPT encryption algorithm that encrypts a message m into a ciphertext c.

(iii) Dec is a deterministic decryption algorithm that decrypts a ciphertext c and outputs either message m or a reject symbol $\perp$.

Now, we define its RCCA security by describing the attack experiment between a challenger and an PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ with the following experiment:

(i) **Setup:** The adversary $\mathcal{A}$ queries Gen algorithm: $(pk, sk) \longleftarrow \mathrm{Gen}(\lambda)$.

(ii) **Stage 1:** The adversary $\mathcal{A}_1$ queries a ciphertext c to Dec: $m \longleftarrow \mathrm{Dec}(sk, c)$, and adversary $\mathcal{A}_1$ responds with m.

(iii) **Challenge stage:** The adversary $\mathcal{A}_1$ queries a pair message (m_0, m_1) to Enc, where $|m_0| = |m_1|$, and then the challenger chooses a bit $b \xleftarrow{R} \{0, 1\}$, computes the challenge cipher $\mathrm{Enc}(pk, m_b) = c^*$, and, finally, sends the challenge c^* to $\mathcal{A}_1$.

(iv) **Stage 2:** The adversary $\mathcal{A}_2$ makes continuous queries c to Dec; here, we require that the cipher c is not identical to the challenge cipher c^*. The decryption algorithm runs $m \longleftarrow \mathrm{Dec}(sk, c)$. Finally, if $m \in \{m_0, m_1\}$, adversary $\mathcal{A}_2$ responds with *text*, or else adversary $\mathcal{A}_2$ responds with m or reject symbol $\perp$.

(v) **Guess stage:** The adversary $\mathcal{A}$ outputs $b' \in \{0, 1\}$.

We let $\mathrm{Adv}^{\mathrm{IND-RCCA}}_{\mathrm{PKE},\mathcal{A}}(\lambda) = |\Pr[b = b'] - 1/2|$ in the above experiment.

If for any PPT adversary $\mathcal{A}$, the function $\mathrm{Adv}^{\mathrm{IND-RCCA}}_{\mathrm{PKE},\mathcal{A}}(\lambda)$ is negligible, we believe that PKE = (Gen, Enc, Dec) is RCCA-secure.

2.2. Signcryption Key Encryption Mechanism (SKEM) and Its RCCA Security Notions. A signcryption key encryption mechanism SKEM = ($\mathrm{KEM.Gen}_S$, $\mathrm{KEM.Gen}_R$, KEM.Enc, KEM.Dec) is a asymmetric encryption scheme [3], which consists of the four algorithms with the following:

(i) $\mathrm{SKEM.Gen}_S$ is a PPT algorithm that inputs a security parameter 1^λ and outputs the sender's public/private key (sk_S, pk_s).

(ii) $\mathrm{SKEM.Gen}_R$ is a PPT algorithm that inputs a security parameter 1^λ and outputs the receiver's public/private key (sk_R, pk_R).

(iii) SKEM.Enc is a PPT encryption algorithm that inputs the send's private key sk_S and the receiver's public key pk_R and outputs (K, C); here, K is a symmetric key and C is the key encapsulation of K.

(iv) SKEM.Dec is a deterministic, polynomial-time decryption algorithm that inputs the sender's public key pk_S, a key encapsulation c, and the receiver's private key sk_R and outputs either a key K or the error symbol $\perp$.

We now define its RCCA security by describing the attack experiment; this experiment is played by an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ and the challenger:

(i) **Setup:** The challenger queries a key generation oracle $\mathtt{SKEM.Gen}_S(\lambda)$ and $\mathtt{SKEM.Gen}_R(\lambda)$. The key generation oracle $\mathtt{SKEM.Gen}_S$ runs $(pk_S, sk_S) \longleftarrow \mathtt{SKEM.Gen}_S(\lambda)$ and the key generation oracle $\mathtt{SKEM.Gen}_S$ runs $(pk_R, sk_R) \longleftarrow \mathtt{SKEM.Gen}_R(\lambda)$. Finally, the key generation oracle $\mathtt{SKEM.Gen}_S$ and $\mathtt{SKEM.Gen}_R$ sends (pk_R, pk_S) to adversary $\mathscr{A}$.

(ii) **Stage 1:** The adversary $\mathscr{A}_1$ inputs (pk_R, pk_S) and makes queries to encapsulation oracle and decapsulation oracle. For every decapsulation oracle algorithm query, the adversary $\mathscr{A}_1$ submits a ciphertext ψ to decryption algorithm Dec: $\mathtt{K} \longleftarrow \mathtt{Dec}(pk_S, sk_R, \psi)$. Finally, responds the adversary $\mathscr{A}$ with K or $\perp$.

(iii) **Challenge stage:** The challenger computes $\psi^* \longleftarrow \mathtt{SKEM.Enc}(sk_S, pk_R, \mathtt{K}_1)$, chooses $\mathtt{K}_0 \overset{R}{\longleftarrow} \mathtt{K}_K$, $\sigma \overset{R}{\longleftarrow} \{0,1\}$, where $\mathtt{K}_K$ is the key space, $|\mathtt{K}_0| = |\mathtt{K}_1|$, and sends $(\mathtt{K}_\sigma, \psi^*)$ to adversary $\mathscr{A}_1$.

(iv) **Stage 2:** The adversary $\mathscr{A}_2$ inputs (pk_R, pk_S) and makes continuous queries ψ to SKEM.Dec. Here, we require that adversary $\mathscr{A}_2$ is not allowed to query (pk_S, c^*) to SKEM.Dec. However, we admit that adversary A_2 can query SKEM.Dec on $(pk_{S'}, C)$ for any $pk_{S'} \neq pk_{S_i}$ and on (pk_{S_i}, C) for any $C \neq C^*$. The decryption oracle algorithm responds with $\mathtt{K} \longleftarrow \mathtt{Dec}(sk_R, pk_S, \psi)$. Finally, if $\mathtt{K} \in \{\mathtt{K}_0, \mathtt{K}_1\}$, $\mathscr{A}_2$ is responded with *text*, or else $\mathscr{A}_2$ is responded with K.

(v) **Guess stage:** In the end, $\mathscr{A}$ outputs a bit $\sigma' \in \{0,1\}$.

In the attack experiment, we let $\mathrm{Adv}^{\mathtt{IND-RCCA}}_{\mathtt{SKEM},\mathscr{A}}(\lambda) = |\Pr[\sigma = \sigma'] - 1/2|$. If for any PPT adversary $\mathscr{A}$, the function $\mathrm{Adv}^{\mathtt{IND-RCCA}}_{\mathtt{SKEM},\mathscr{A}}(\lambda)$ is negligible, we say the signcryption scheme $\mathtt{SKEM} = (\mathtt{SKEM.Gen}_R, \mathtt{SKEM.Gen}_R, \mathtt{SKEM.Enc}, \mathtt{SKEM.Dec})$ is RCCA-secure.

2.3. Data Encryption Mechanism and Its IND-RCCA Security. A signcryption data encryption mechanism DEM is a symmetric encryption scheme, which consists of the following two algorithms: DEM.ENC, DEM.Dec.

(i) $\mathtt{DEM.Enc} : c \longleftarrow \mathtt{DEM.Enc}(\mathtt{K}, m)$; DEM.Enc is a polynomial-time encryption algorithm; DEM.Enc encrypts m by using a key K and outputs the corresponding ciphertext c.

(ii) $\mathtt{DEM.Dnc} : m \longleftarrow \mathtt{DEM.Dec}(\mathtt{K}, c)$; DEM.Dec is a polynomial-time decryption algorithm; it inputs ciphertext χ and decrypts the cipher c by using the same key K.

We define its IND–RCCA security by describing the attack experiment; this experiment is played by an adversary $\mathscr{A} = (\mathscr{A}_1, \mathscr{A}_2)$ and the challenger:

(i) **Setup 1:** The challenger chooses a key symmetric $K \overset{R}{\longleftarrow} K_D$.

(ii) **Challenge stage:** The adversary $\mathscr{A}$ queries (m_0, m_1) to DEM.Enc, $|m_0| = |m_1|$. The challenger chooses $b \overset{R}{\longleftarrow} \{0,1\}$, computes the challenge cipher $c^* \longleftarrow \mathtt{Enc}(\mathtt{K}, m_b)$, and then sends the challenge cipher c^* to adversary $\mathscr{A}$.

(iii) **Setup 2:** The adversary $\mathscr{A}_2$ continues to make queries cipher c to Dec: $m \longleftarrow \mathtt{Dec}(c, \mathtt{K})$; here, c is not equal to the challenge cipher c^*. If $m \in \{m_0, m_1\}$, Dec responds to adversary $\mathscr{A}$ with *text*, or else Dec responds to adversary $\mathscr{A}$ with m.

(iv) **Guess stage:** In the end, the adversary $\mathscr{A}$ outputs $b' \in \{0,1\}$.

We define $\mathrm{Adv}^{\mathtt{IND-RCCA}}_{\mathtt{DEM},\mathscr{A}}(\lambda) = |\Pr[b = b'] - 1/2|$ in the above experiment.

If for any PPT adversary $\mathscr{A}$, the function $\mathrm{Adv}^{\mathtt{IND-RCCA}}_{\mathtt{DEM},\mathscr{A}_2}(\lambda)$ is negligible, the scheme DEM is IND–RCCA secure.

3. The RCCA Security of Hybrid Signcryption Schemes

In this section, we will recall the definition of hybrid signcryption which is adapted by Dent and An [15, 33]. Some definitions include the verification algorithm, whose aim is to provide nonrepudiation. However, in their view, nonrepudiation is unnecessary for most cryptography applications and hence will not be discussed further. Next, we examine the RCCA security for hybrid signcryption and consider the outsider security (the adversary is third party, neither sender nor receiver) of hybrid signcryption, which is proposed by Dent in [3].

3.1. SKEM+DEM Hybrid Signcryption Scheme and Its Relaxing Chosen Cipher Attack Security. $\mathtt{SKEM} = (\mathtt{SKEM.Gen}_S, \mathtt{SKEM.Gen}_R, \mathtt{SKEM.Enc}, \mathtt{SKEM.Dec})$ is signcryption key encapsulation mechanism, $\mathtt{DEM} = (\mathtt{DEM.Enc}, \mathtt{DEM.Dec})$ is data encapsulation mechanism, and hybrid signcryption scheme $\mathtt{SKEM+DEM} = (\mathtt{signcrypt.Gen}, \mathtt{signcrypt.Enc}, \mathtt{signcrypt.Dec})$ can be constructed from SKEM and DEM as follows:

(i) $\mathtt{signcrypt.Gen}(1^\lambda)$: It runs receiver's key generation algorithm $(pk_R, sk_R) \longleftarrow \mathtt{SKEM.Gen}_R(1^\lambda)$ and runs sender's key generation algorithm $(pk_S, sk_S) \longleftarrow \mathtt{SKEM.Gen}_S(1^\lambda)$, Finally, it outputs (pk_S, sk_S) and (pk_R, sk_R).

(ii) signcrypt.Enc(pk, m): signcrypt.Enc is a PPT algorithm that inputs the sender's private key sk_S, a message m, and the receiver's public key pk_R. It chooses $\mathtt{K} \longleftarrow \mathtt{K}_D$ and computes $\chi \longleftarrow \mathtt{SKEM.Enc}(sk_S, pk_R, \mathtt{K})$; here $\mathtt{K}_D$ is the signcryption scheme DEM's key space. Then it computes $\psi \longleftarrow \mathtt{DEM.Enc}_K(m)$, and the resulting signcryption is $c \coloneqq (\chi, \psi)$.

(iii) signcrypt.Dec(sk, c): the signcrypt.Dec algorithm inputs the sender's public key pk_S, a cipher c, and the receiver's private key sk_R. It then parses cipher c as $\psi \parallel \chi$ and runs $\mathtt{K} \longleftarrow \mathtt{TKEM.Dec}(\psi, pk_S, sk_R), m \longleftarrow \mathtt{DEM.Dec}_K(\chi)$. In the end, it outputs m or "reject" symbol $\perp$.

3.2. The RCCA Security of Hybrid Signcryption Schemes

Theorem 1. *The hybrid signcryption scheme* (SKEM + DEM) *can be constructed from a signcryption scheme* SKEM *and a scheme* DEM. *If the signcryption scheme* SKEM *is* IND–RCCA *secure and the signcryption scheme* DEM *is* IND–RCCA *secure, then hybrid signcryption scheme* (SKEM + DEM) *can achieve* IND–RCCA *security. For every given* PPT *adversary* $\mathscr{A}$, *there exist probabilistic adversary* $\mathscr{A}_1$ *and adversary* $\mathscr{A}_2$, *such that the following conclusion holds:*

$$\mathrm{Adv}^{\mathrm{RCCA}}_{\mathrm{SKEM+DEM},\mathscr{A}}(\lambda) \le 2\mathrm{Adv}^{\mathrm{RCCA}}_{\mathrm{SKEM},\mathscr{A}_1}(\lambda) + \mathrm{Adv}^{\mathrm{RCCA}}_{\mathrm{DEM},\mathscr{A}_2}(\lambda) + \frac{q_s}{|K_D|}. \tag{1}$$

Here, we assume the adversary $\mathscr{A}_1$ at most makes the q_s queries to the encryption-decryption oracle, the running times of $\mathscr{A}_1$ and $\mathscr{A}_2$ are equal to that of adversary $\mathscr{A}$, and K_D is the signcryption scheme DEM's key space.

Proof. Fix adversary $\mathscr{A}$ and λ; $\mathscr{A}$ is a PPT IND–RCCA adversary, which attacks the hybrid signcryption scheme SKEM + DEM; then we proved the theorem by the following experiments.

Experiment$_0$: This is an IND–RCCA experiment on the signcryption scheme SKEM + DEM, which is played by an adversary $\mathscr{A} = (\mathscr{A}_1, \mathscr{A}_2)$ and the challenger. (We denote by T_0 the event of adversary $\mathscr{A}$ succeeding in this experiment.)

(i) **Setup:** The adversary $\mathscr{A}$ makes queries to key generation algorithm $(pk_R, sk_R) \longleftarrow \mathtt{SKEM.Gen}_R(1^\lambda)$ and makes queries to key generation algorithm $(pk_S, sk_S) \longleftarrow \mathtt{SKEM.Gen}_S(1^\lambda)$. Finally, it sends (pk_R, pk_S) to adversary $\mathscr{A}$.

(ii) **Stage 1:** The adversary $\mathscr{A}$ inputs a public key pair (pk_R, pk_S) and makes continuous queries to decryption oracle algorithm. For adversary $\mathscr{A}$'s decryption algorithm query $c = (\psi, \chi)$, the adversary $\mathscr{A}_1$ sends a cipher $c = (\psi, \chi)$ to the challenger $\mathscr{C}$, and the challenger $\mathscr{C}$ runs decryption algorithm $\mathtt{K} \longleftarrow \mathtt{SKEM.Dec}(\psi, pk_s, sk_R)m \longleftarrow \mathtt{DEM.Dec}_K(\chi)$. In the end, the challenger responds to $\mathscr{A}_1$ with m.

(iii) **Challenge stage:** The adversary $\mathscr{A}_1$ inputs (pk_R, pk_S) and queries (m_0, m_1) to an encryption oracle, and the challenger chooses $\mathtt{K}_1 \longleftarrow \mathtt{K}_D$ and chooses $b \in \{0, 1\}$. Then the challenger computes $\chi \longleftarrow \mathtt{SKEM.Enc}(sk_S, pk_R, \mathtt{K}_1)$ and computes $\psi \longleftarrow \mathtt{DEM.Enc}_\mathtt{K}(m_b)$ the signcryption $c := (\chi, \psi)$.

(iv) **Stage 2:** The adversary $\mathscr{A}_2$ inputs (pk_{R_i}, pk_{S_i}) and makes continuous queries $c = (\psi, \chi)$ to the challenger. Here, the adversary $\mathscr{A}_2$ is not admitted to query (pk_S, c^*) to the decryption oracle algorithm. But we admit that adversary A_2 can make a query to the decryption oracle algorithm on $(pk_{S'}, C)$ for any $pk_{S'} \neq pk_{S_i}$ and on (pk_{S_i}, C) for any cipher $C \neq C^*$. The challenger runs decryption oracle $\mathtt{K} \longleftarrow \mathtt{SKEM.Dec}(pk_{S_i}, sk_{R_i}, \psi, \chi)$ and $m \longleftarrow \mathtt{DEM.Dec}_{K_i}(\chi)$. If $m \in \{m_0, m_1\}$, the challenger responds to $\mathscr{A}_2$ with *text* or else responds to $\mathscr{A}_2$ with m.

(v) **Guess stage:** In the end, the adversary $\mathscr{A}$ outputs a guessing bit $b' \in \{0, 1\}$.

The following conclusion holds:

$$\mathrm{Adv}^{\mathrm{RCCA}}_{\mathrm{SKEM+DEM},\mathscr{A}}(\lambda) = \left|\Pr\left[b = b'\right] - \frac{1}{2}\right| = \left|\Pr\left[T_0\right] - \frac{1}{2}\right|. \tag{2}$$

Experiment$_1$: We now modify experiment G_0 to obtain a new experiment G_1. These two experiments are identical except that we use a uniformly random key $\mathtt{K}_0 \overset{R}{\longleftarrow} \mathtt{K}_D$ to compute the challenge cipher $C^* = (\chi^*, \psi^*)$ in step 3 of $\mathtt{Game}_0$; the challenge cipher $C^* = (\chi^*, \psi^*)$ is computed by the encryption algorithm $\mathtt{SKEM.Enc}(sk_S, pk_R, \mathtt{K}_0)$ and $\psi \longleftarrow \mathtt{DEM.Enc}_K(m)$. To maintain consistency, the challenger should use the symmetric key $\mathtt{K}_0$ to answer the decryption oracle algorithm query $(pk_S, \chi, .)$. Hence, the distinction between experiment G_0 and experiment G_1 mainly lies in how the scheme SKEM runs. (Denote by T_1 the sign of the adversary $\mathscr{A}$ succeeding in this experiment.) We have the following conclusion.

Lemma 2. *There is an adversary* $\mathscr{A}_1$, *and its running time is equal to the running time of adversary* $\mathscr{A}$; *the following conclusion holds:*

$$\left|\Pr\left[T_1\right] - \Pr\left[T_0\right]\right| \le 2\mathrm{Adv}^{\mathrm{RCCA}}_{\mathrm{SKEM},\mathscr{A}_1}(\lambda) + \frac{q_s}{|\mathtt{K}_D|}. \tag{3}$$

Proof. We prove the lemma by constructing an adversary $\mathscr{A}_1$ who attacks the signcryption scheme SKEM. The adversary $\mathscr{A}_1$ simulates the environment for $\mathscr{A}$, their interactions can be described as follows:

(i) **Setup:** The adversary $\mathscr{A}_1$ was given $(pk_S, pk_R, \mathtt{K}_\sigma)$, and the adversary $\mathscr{A}_1$ sent (pk_S, pk_R) to $\mathscr{A}$.

(ii) **Stage 1:** The adversary $\mathscr{A}$ inputs (pk_S, pk_R) and makes some queries c to the decryption oracle: $m \longleftarrow \mathtt{Dec}(sk_R, pk_S, c)$. Finally, $\mathscr{A}$ is reponded with m or reject symbol $\perp$.

(iii) **Challenge stage:** The adversary $\mathscr{A}$ inputs (pk_S, pk_R) and queries (m_0, m_1) to an encryption oracle, $|m_0| = |m_1|$. The adversary $\mathscr{A}_1$ computes $\chi^* \longleftarrow \mathtt{SKEM.Enc}(sk_S, pk_R, \mathtt{K}_1)$ and computes $\psi^* \longleftarrow \mathtt{DEM.Enc}_\mathtt{K}(m_b)$, and finally the adversary $\mathscr{A}_1$ sends the challenge cipher $c^* = (\psi^*, \chi^*)$ to the adversary $\mathscr{A}$.

(iv) **Stage 2:** The adversary $\mathscr{A}$ inputs (pk_{S_i}, pk_{R_i}) and makes queries $c = (\psi_i, \chi_i)$ to decryption oracle algorithm. Here, we require that $\mathscr{A}_2$ cannot query (pk_S, c^*) to the decryption oracle algorithm. However, we admit that adversary A_2 can query the decryption oracle algorithm on $(pk_{S'}, C)$ for any $pk_{S'} \neq pk_{S_i}$ and on (pk_{S_i}, C) for $C \neq C^*$. The adversary $\mathscr{A}_1$ runs

$$\begin{aligned} &\mathtt{K}_i \longleftarrow \mathtt{SKEM.Dec}\left(pk_{S_i}, sk_{R_i}, \chi_i, \psi_i\right), \\ &m \longleftarrow \mathtt{DEM.Dec}_{K_i}\left(\psi_i\right). \end{aligned} \tag{4}$$

Finally, if $m \in \{m_0, m_1\}$, $\mathscr{A}_2$ responds with *text*, or else $\mathscr{A}_2$ is responded with m.

(v) **Guess stage:** The adversary $\mathscr{A}$ outputs a guessing bit $b' \in \{0, 1\}$ and $\mathscr{A}_1$ outputs $\sigma' = b'$ in the end.

This has completed the construction of $\mathscr{A}_1$. By description, we can see that the adversary $\mathscr{A}_1$ played a perfectly simulated decryption for adversary $\mathscr{A}$ unless the cipher ψ^* is decrypted to $\mathtt{K}_1$ and test is returned by the correct answer from the decryption oracle SKEM.Dec for every query. However, the probability of this event is $1/|\mathtt{K}_D|$ since in that case the key $\mathtt{K}_1$ is uniformly random and independent of the opinion of the adversary $\mathscr{A}_1$ for each such query.

(i) If $\sigma = 0$, we can obtain that cipher χ is computed by a random key $\mathtt{K}_0$; meanwhile, the opinion of the adversary $\mathscr{A}$ is equal to that in $\mathtt{Experiment}_0$.

(ii) If $\sigma = 1$, we can obtain that $\mathtt{K}_1$ is corresponding correct key embedded in the cipher ψ; meanwhile, the opinion of the adversary $\mathscr{A}$ is equal to that in $\mathtt{Experiment}_1$.

Thus,

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{RCCA}}_{\mathtt{SKEM},\mathscr{A}_1}(\lambda) &= \left|\Pr\left[b = b'\right] - \frac{1}{2}\right| = \left|\Pr\left[T_0\right] - \frac{1}{2}\right| \\
&= \frac{1}{2}\left|\Pr\left[\sigma' = 1 \mid \sigma = 1\right] - \Pr\left[\sigma' = 1 \mid \sigma = 0\right]\right| \\
&= \frac{1}{2}\left|\Pr\left[b = b' \mid \sigma = 1\right] - \Pr\left[b = b' \mid \sigma = 0\right]\right| \\
&\geq \frac{1}{2}\left|\Pr\left[T_0\right] - \Pr\left[T_1\right] - \frac{q_s}{|\mathtt{K}_D|}\right|.
\end{aligned} \tag{5}$$

We can get the following conclusion:

$$\left|\Pr\left[T_1\right] - \Pr\left[T_0\right]\right| \leq 2\mathrm{Adv}^{\mathrm{RCCA}}_{\mathtt{SKEM},\mathscr{A}_1}(\lambda) + \frac{q_s}{|\mathtt{K}_D|}. \tag{6}$$

Lemma 2 is proved. □

In the stage of experiment G_1's encryption and decryption oracle algorithms, we use a uniformly random key $\mathtt{K}_0$, so the challenger cipher ψ^* is not be decrypted. From this point, we notice that the challenge cipher χ^* is generated by using a random symmetric key $\mathtt{K}_0$ in experiment G_1. Meanwhile, the other cipher $\chi = \chi^*$ is decrypted by using random key $\mathtt{K}_0$, which has no other role in experiment G_1. Hence, in experiment G_1, the adversary $\mathscr{A}$ plays an adaptive replayable chosen ciphertext attack against (RCCA) the signcryption scheme DEM in substance, so the following conclusion holds.

Lemma 3. *There is a probabilistic adversary $\mathscr{A}_2$, and its running time is equal to the running time of the adversary $\mathscr{A}$, such that the following conclusion holds:*

$$\left|\Pr\left[T_1\right] - \frac{1}{2}\right| \leq \mathrm{Adv}^{\mathrm{RCCA}}_{\mathtt{DEM},\mathscr{A}_2}(\lambda). \tag{7}$$

Proof. The symmetric key $\mathtt{K}_0$ was chosen uniformly, randomly, and independently, so the challenge cipher χ does not reveal related information about which message was encrypted. Hence, to gain success in experiment 2, the adversary must learn some information from the challenger cipher χ. We prove Lemma 3 by constructing a probabilistic adversary $\mathscr{A}_2$, who attacks the signcryption scheme DEM, and $\mathscr{A}_2$ provides an environment for the adversary $\mathscr{A}$. Now, we describe their interactions:

(i) **Setup:** The adversary $\mathscr{A}_2$ runs receiver key generation algorithm $(pk_R, sk_R) \longleftarrow \mathtt{SKEM.Gen}_R(1^\lambda)$, runs sender key generation algorithm $(pk_S, sk_S) \longleftarrow \mathtt{SKEM.Gen}_S(1^\lambda)$, and sends (pk_R, pk_S) to $\mathscr{A}$.

(ii) **Stage 1:** The adversary $\mathscr{A}$ inputs (pk_R, pk_S) and makes queries ciphertext c to a decryption oracle algorithm: $\mathtt{K} \longleftarrow \mathtt{SKEM.Dec}(pk_S, sk_R, \chi)$, $m \longleftarrow \mathtt{DEM.Dec}(\mathtt{K}, \chi)$. If $m = \perp$, the decryption oracle algorithm responds to adversary $\mathscr{A}$ with m or reject symbol $\perp$.

(iii) **Challenge Stage:** The adversary $\mathscr{A}$ inputs a public key pair (pk_R, pk_S) and sends (m_0, m_1) to the adversary $\mathscr{A}_2$. the adversary $\mathscr{A}_2$ chooses $\mathtt{K}_1 \overset{R}{\longleftarrow} \mathtt{K}_D$, runs $\psi^* \longleftarrow \mathtt{SKEM.Enc}(pk_S, sk_R, \mathtt{K}_1)$, and sends the challenge $c^* = (\psi^*, \chi^*)$ to $\mathscr{A}$. We notice that the symmetric key $\mathtt{K}_1$ was chosen as the encryption key of scheme DEM and embedded in cipher ψ^*, which is uniformly random and independent of each other.

(iv) **Stage 2:** The adversary inputs (pk_R, pk_S) and makes continuous queries $c = (\psi, \chi)$ to decryption oracle algorithm. Here, we require that adversary $\mathscr{A}_2$ cannot query (pk_S, c^*) to the decryption oracle algorithm. However, we admit adversary A_2 can make a query to the decryption oracle algorithm on $(pk_{S'}, C)$ for any cipher $pk_{S'} \neq pk_{S_i}$ and on (pk_{S_i}, C) for any $C \neq C^*$. The adversary $\mathscr{A}_2$ uses the secret key sk_R to run the decryption oracle algorithm and answer the decryption query $c = (\psi, \chi)$ of adversary $\mathscr{A}$ with the following:

(a) If $\psi_i = \psi^*$, hence $\chi_i \neq \chi^*$. Then The adversary $\mathscr{A}_2$ runs the decryption oracle $\mathtt{K} \longleftarrow \mathtt{SKEM.Dec}(pk_S, sk_R, \chi)$. If $\mathtt{K} = \perp$, the adversary $\mathscr{A}_2$ responds to $\mathscr{A}$ with $\perp$ or else $m \longleftarrow \mathtt{DEM.Dec}(\mathtt{K}, \chi)$. If $m \in \{m_0, m_1\}$, the adversary $\mathscr{A}_2$ responds to $\mathscr{A}$ with *text*, or else $\mathscr{A}_2$ responds to $\mathscr{A}$ with m.

(v) **Guess Stage:** Finally, adversary $\mathscr{A}$ outputs a bit $b' \in \{0, 1\}$ and $\mathscr{A}_2$ also outputs a bit b'.

This has completed the description of the adversary $\mathscr{A}_2$. By our construction, it is obvious that the adversary $\mathscr{A}_2$ plays a perfectly simulated decryption for $\mathscr{A}$, and whenever $\mathscr{A}$ gets success, so does $\mathscr{A}_2$. We have the following conclusion:

$$\left|\Pr\left[T_1\right] - \frac{1}{2}\right| \leq \mathrm{Adv}^{\mathrm{RCCA}}_{\mathtt{DEM},\mathscr{A}_2}(\lambda). \tag{8}$$

□

We can know that the advantage of $\mathscr{A}$ in experiment$_0$ is

$$\begin{aligned}\mathrm{Adv}^{\mathrm{RCCA}}_{\mathrm{SKEM+DEM},\mathscr{A}}(\lambda) &= \left|\Pr[T_0] - \frac{1}{2}\right| \\ &\leq 2\mathrm{Adv}^{\mathrm{RCCA}}_{\mathrm{SKEM},\mathscr{A}_1}(\lambda) + \mathrm{Adv}^{\mathrm{RCCA}}_{\mathrm{DEM},\mathscr{A}_2}(\lambda),\end{aligned} \tag{9}$$

which is negligible; we have proved Theorem 1.

3.3. The Hybrid Signcryption Scheme Tag-SKEM+DEM and Its RCCA Security

Definition 4 (signcryption scheme Tag-SKEM). A signcryption scheme $\mathtt{Tag\text{–}KEM} = (\mathtt{Gen}_R, \mathtt{Gen}_S, \mathtt{Tag\text{–}KEM.Enc}, \mathtt{Tag\text{–}KEM.Dec})$ consists of the following three algorithms:

(i) $\mathtt{Tag\text{–}KEM.Gen}_S(1^\lambda)$:$\mathtt{Tag\text{–}KEM.Gen}_S(1^\lambda)$ is a PPT algorithm that inputs a security parameter 1^λ and outputs a pair of public/private keys (sk_S, pk_S).

(ii) $\mathtt{Tag\text{–}KEM.Gen}_R(1^\lambda)$:$\mathtt{TKEM.Gen}_S(1^\lambda)$ is a PPT algorithm that inputs a security parameter 1^λ and outputs a pair of public/private keys (sk_R, pk_R).

(iii) An encryption algorithm $\mathtt{Tag\text{–}KEM.Enc}$: It runs $(\omega, \mathtt{K}) \longleftarrow \mathtt{Tag\text{–}KEM.Key}(pk_R, sk_S)$. $\mathtt{Tag\text{–}KEM.Key}(\cdot)$ is a PPT algorithm that inputs the private key of sender sk_S and public key of receiver pk_R and outputs one-time key $\mathtt{K}$ and Intermediate state information ω. Choose $r \overset{R}{\leftarrow} \{0,1\}^\lambda$ and compute $\psi \longleftarrow \mathtt{TKEM.Enc}(\omega, r, pk_R, sk_S, \tau)$. $\mathtt{Tag\text{–}KEM.Enc}$ is a PPT algorithm that encrypts the key $\mathtt{K}$ (embedded in ω) into cipher ψ along with a tag $\tau \in T$ and returns a cipher ψ; here, τ is called a tag.

(iv) An decryption algorithm $\mathtt{Tag\text{–}KEM.Dec} : \mathtt{K} \longleftarrow \mathtt{Tag\text{–}KEM.Dec}(pk_S, sk_R, \psi, \tau)$. $\mathtt{TKEM.Dec}$ is a deterministic decryption verification algorithm for a signcryption cipher, which inputs the receiver's private key sk_R, the cipher c, the sender's public key pk_S, and a tag τ; the decryption oracle $\mathtt{Tag\text{–}KEM.Dec}$ returns a key $\mathtt{K}$ or reject symbol $\perp$.

Definition 5 (hybrid signcryption scheme Tag-SKEM+DEM). The signcryption scheme

$$\mathtt{Tag\text{–}SKEM} = \left(\mathtt{Gen}_R, \mathtt{Gen}_S, \mathtt{TKEM.Enc}, \mathtt{TKEM.Dec}\right) \tag{10}$$

is a asymmetric encryption scheme and the signcryption scheme $\mathtt{DEM} = (\mathtt{DEM.Enc}, \mathtt{DEM.Dec})$ is a corresponding symmetric encryption scheme [18].

Then the hybrid signcryption scheme

$$\begin{aligned}&\mathtt{Tag\text{–}SKEM + DEM} \\ &\quad = (\mathtt{Gen}, \mathtt{signcrypt}, \mathtt{unsigncrypt})\end{aligned} \tag{11}$$

can be constructed as follows:

(i) Key generation algorithm $\mathtt{Gen}(1^\lambda)$: $\mathtt{Gen}_R(1^\lambda)$ is a probabilistic receiver's key generation algorithm that inputs a 1^λ and outputs the receiver's public/private key pair (pk_R, sk_R); we write this as $(pk_R, sk_R) \longleftarrow \mathtt{Gen}_R(1^\lambda)$. $\mathtt{Gen}_S(1^\lambda)$ is a probabilistic receiver key generation algorithm, which takes a as input a security parameter 1^λ and as output a receiver's public/private key pair (pk_S, sk_S); we write this as $(pk_S, sk_S) \longleftarrow \mathtt{Gen}_S(1^\lambda)$.

(ii) An encryption algorithm $\mathtt{signcrypt} : \mathtt{Tag\text{–}SKEM.Key}(\cdot)$ is a probabilistic algorithm that inputs the receiver's public key pk_S and outputs a symmetric key $\mathtt{K} \in \mathtt{K}_D$ and the internal state information ω, $(\omega, \mathtt{K}) \longleftarrow \mathtt{Tag\text{–}SKEM.Key}(pk_S)$. Here $\mathtt{K}_D$ is the scheme $\mathtt{DEM}$'s key space. Then choose $r \overset{R}{\leftarrow} \{0,1\}^\lambda$ and compute $\chi \longleftarrow \mathtt{DEM.Enc}_K(m)$, $\psi \longleftarrow \mathtt{Tag\text{–}SKEM.Enc}(pk_S, sk_R, \omega, r, \chi)$. Finally, output the signcrypt cipher $c = (\psi, \chi)$.

(iii) A decryption algorithm $\mathtt{unsigncrypt}$: First, it parses the cipher c to obtain $\psi \parallel \chi$. Next, it computes $\mathtt{K} \longleftarrow \mathtt{Tag\text{–}SKEM.Dec}(sk_R, pk_R, \psi, \chi)$ to obtain a symmetric key $\mathtt{K}$ and computes $m \longleftarrow \mathtt{DEM.Dec}_K(\chi)$. Finally, it outputs the message m or "reject" symbol $\perp$.

3.4. The RCCA Security of Hybrid Signcryption Scheme Tag-SKEM+DEM

Theorem 6. *The hybrid signcryption scheme* ($\mathtt{Tag\text{–}SKEM + DEM}$) *is constructed from a scheme* $\mathtt{Tag\text{–}SKEM}$ *and a scheme* $\mathtt{DEM}$. *If the signcryption scheme* $\mathtt{Tag\text{–}SKEM}$ *is* $\mathtt{IND\text{–}RCCA}$ *secure and the signcryption scheme* $\mathtt{DEM}$ *is* $\mathtt{IND\text{–}RCCA}$ *secure, then the hybrid signcryption scheme* ($\mathtt{Tag\text{–}SKEM + DEM}$) *is also* $\mathtt{IND\text{–}RCCA}$ *secure. For every* PPT *adversary* $\mathscr{A}$, *there are probabilistic adversary* $\mathscr{A}_1$ *and adversary* $\mathscr{A}_2$, *whose running times are essentially equal to that of adversary* $\mathscr{A}$, *such that for all* $\lambda \geq 0$, *the following holds.*

$$\begin{aligned}\mathrm{Adv}^{\mathrm{RCCA}}_{\mathtt{Tag\text{–}SKEM+DEM},\mathscr{A}}(\lambda) &\leq 2\mathrm{Adv}^{\mathrm{RCCA}}_{\mathtt{Tag\text{–}KEM},\mathscr{A}_1}(\lambda) \\ &\quad + \mathrm{Adv}^{\mathrm{RCCA}}_{\mathrm{DEM},\mathscr{A}_2}(\lambda) + \frac{q_s}{|\mathtt{K}_D|}.\end{aligned} \tag{12}$$

Here, we assume the adversary at most makes the q_s queries to the encryption-decryption oracle algorithm and $\mathtt{K}_D$ is the scheme $\mathtt{DEM}$'s key space.

Proof. We prove the theorem by constructing a PPT adversary $\mathscr{A}$ who attacks the hybrid signcryption scheme $\mathtt{Tag\text{–}SKEM + DEM}$ with the following experiments. (We denote by T_i the event of the adversary $\mathscr{A}$ succeeding in the i-th game.)

Experiment$_0$: This is the $\mathtt{IND\text{–}RCCA}$ experiment on the signcryption scheme $\mathtt{Tag\text{-}SKEM+DEM}$, and this experiment is played between an adversary $\mathscr{A}$ and the challenger as follows:

(i) **Setup:** The adversary queries a key generation oracle. The challenger runs receiver key generation algorithm $(pk_R, sk_R) \longleftarrow \mathtt{Tag\text{–}SKEM.Gen}_R(1^\lambda)$, runs sender key generation algorithm $(pk_S, sk_S) \longleftarrow \mathtt{Tag\text{–}SKEM.Gen}_S(1^\lambda)$, and responds to the adversary $\mathscr{A}$ with (pk_R, pk_S).

(ii) **Stage 1:** The adversary $\mathcal{A}$ inputs (pk_R, pk_S) and makes continuous queries to decryption oracle algorithm. The adversary $\mathcal{A}$ sends a cipher c to the decryption oracle algorithm, and the decryption oracle algorithm runs $\mathtt{K} \longleftarrow \mathtt{Tag\text{–}SKEM.Dec}(pk_S, sk_R, \chi, \psi)$, $m \longleftarrow \mathtt{DEM.Dec}(\mathtt{K}, \chi)$. If $m = \perp$, the decryption oracle algorithm responds to $\mathcal{A}_1$ with $\perp$ or else responds to $\mathcal{A}_1$ with m.

(iii) **Challenge stage:** The adversary $\mathcal{A}_1$ inputs a public key pair (pk_R, pk_S) and queries (m_0, m_1) to an encryption oracle algorithm, and then the challenger runs $(\omega, \mathtt{K}) \longleftarrow \mathtt{Tag\text{–}SKEM.Key}(pk_R sk_S)$, $\mathtt{K} \in \mathtt{K}_D$. Then the challenger computes $\mathtt{DEM.Enc}_K(m_0) = \chi^*$, $\mathtt{Tag\text{–}SKEM.Enc}(pk_R, sk_S, \omega, \chi^*) = \psi^*$ and sends the challenge cipher $c^* = (\psi^*, \chi^*)$ to the adversary $\mathcal{A}_1$.

(iv) **Stage 2:** The adversary $\mathcal{A}_2$ inputs a public key pair (pk_{R_i}, pk_{S_i}) and makes continuous queries $c = (\psi, \chi)$ to the challenger. Here, we require that adversary $\mathcal{A}_2$ is not admitted to query (pk_S, c^*) to the decryption oracle. However, we admit that adversary A_2 can make a query to the decryption oracle on $(pk_{S'}, C)$ for any public key $pk_{S'} \neq pk_{S_i}$ and on (pk_{S_i}, C) for any cipher $C \neq C^*$. The challenger runs decryption oracle.

$$\begin{aligned} &\mathtt{K} \longleftarrow \mathtt{Tag\text{–}SKEM.Dec}\left(sk_{R_i}, pk_{S_i}, \psi, \chi\right), \\ &m \longleftarrow \mathtt{DEM.Dec}_K\left(\chi\right) \end{aligned} \tag{13}$$

Finally, if $m \in \{m_0, m_1\}$, $\mathcal{A}_2$ responds with *text*, or else $\mathcal{A}_2$ responds with m.

(v) **Guess stage:** In the end, the adversary $\mathcal{A}$ outputs a guess bit $b' \in \{0, 1\}$.

Naturally, the following holds:

$$\begin{aligned} \mathrm{Adv}^{\mathtt{RCCA}}_{\mathtt{Tag-SKEM+DEM},\mathcal{A}}(\lambda) &= \left|\Pr\left[T_0\right] - \frac{1}{2}\right| \\ &= \left|\Pr\left[b = b'\right] - \frac{1}{2}\right|. \end{aligned} \tag{14}$$

Experiment$_1$: We now modify $\mathtt{Experiment}_0$ to obtain a new $\mathtt{Experiment}_1$; this experiment is equal to the above experiment except that we just use a random key $\mathtt{K}_0 \overset{R}{\longleftarrow} \mathtt{K}_D$ to encrypt the message m_0 in step 3 of $\mathtt{experiment}_0$; hence, we get the following conclusion.

Lemma 7. *There exists a probabilistic adversary $\mathcal{A}_1$, and its running time is equal to that of adversary $\mathcal{A}$, such that the following conclusion holds:*

$$\left|\Pr\left[T_1\right] - \Pr\left[T_0\right]\right| \leq \mathrm{Adv}^{\mathtt{RCCA}}_{\mathtt{Tag-SKEM},\mathcal{A}_1}(\lambda) + \frac{q_s}{\left|\mathtt{K}_D\right|}. \tag{15}$$

Here, we assume the adversary at most makes the q_s queries to the encryption-decryption oracle algorithm.

Proof. We prove the lemma by constructing an adversary $\mathcal{A}_1$ who attacks signcryption scheme Tag-SKEM. The adversary $\mathcal{A}_1$ simulates an environment for adversary $\mathcal{A}$; their interactions can be described as follows:

(i) **Stage 1:** The adversary $\mathcal{A}_2$ was given $(pk_R, pk_S, \mathtt{K}_\sigma)$, and at the same time, (pk_R, pk_S) was sent to adversary $\mathcal{A}$.

(ii) **Stage 2:** The adversary $\mathcal{A}$ inputs a public key pair (pk_R, pk_S) and makes continuous queries c to a decryption oracle algorithm Dec. The decryption oracle algorithm runs $m \longleftarrow \mathtt{Dec}(pk_S, sk_R, c)$. Finally, if $m = \perp$, $\mathcal{A}$ responds with m or reject symbol $\perp$.

(iii) **Stage 3:** The adversary $\mathcal{A}$ inputs a pair public key (pk_R, pk_S) and queries (m_0, m_1) to the encryption oracle, $|m_0| = |m_1|$. The adversary $\mathcal{A}_1$ requires the encryptions oracle of scheme Tag-SKEM to obtain $(\mathtt{K}_\sigma, \psi^*)$. The adversary $\mathcal{A}_1$ chooses $b \in \{0, 1\}$ and computes $\mathtt{DEM.EncS}(K_\sigma, m_b) = \chi^*$. Finally, the adversary $\mathcal{A}_1$ sends challenge cipher $c^* = (\psi^*, \chi^*)$ to the adversary $\mathcal{A}$.

(iv) **Stage 4:** The adversary $\mathcal{A}$ inputs (pk_{R_i}, pk_{S_i}) and makes continuous calls $c = (\psi_i, \chi_i)$ to decryption oracle query. Here, we require that adversary $\mathcal{A}_2$ is not admitted to query (pk_S, c^*) to the decryption oracle algorithm. However, we admit that adversary A_2 can make a query to the decryption oracle on $(pk_{S'}, C)$ for any $pk_{S'} \neq pk_{S_i}$ and on (pk_{S_i}, C) for any $C \neq C^*$. The adversary $\mathcal{A}_1$ runs its own decryption oracle $\mathtt{Tag\text{–}SKEM.Dec}(pk_S, sk_R, .)$ to answer the adversary $\mathcal{A}$'s decryption query as follows:

(a) If $\perp$ is returned, then the adversary $\mathcal{A}_1$ responds to $\mathcal{A}$ with $\perp$.

(b) If $\perp$ is returned and $\chi_i \neq \chi^*$, then $\mathcal{A}_1$ uses K_σ to decrypt the cipher χ.

(1) If m_0 or m_0 is returned, then the adversary $\mathcal{A}_1$ responds to $\mathcal{A}$ with *text*.
(2) Otherwise, $\mathcal{A}_1$ responds to $\mathcal{A}$ with the result.

(c) If *test* is returned and cipher $\chi_i = \chi^*$, then the adversary $\mathcal{A}_1$ responds to $\mathcal{A}$ with *text*.

(d) If $\mathtt{K}_1$ is returned, then the adversary uses $\mathtt{K}_1$ to decrypt the cipher χ.

(1) If m_0 or m_0 is returned, then the adversary $\mathcal{A}_1$ responds to adversary $\mathcal{A}$ with *text*.
(2) Otherwise, $\mathcal{A}_1$ responds to adversary $\mathcal{A}$ with the result.

(v) **Stage 5:** In the end, the adversary $\mathcal{A}$ outputs a guess bit $b' \in \{0, 1\}$, and $\mathcal{A}_1$ outputs a bit $\sigma' = b'$.

This has completed the description of $\mathcal{A}_1$; it is clear that the adversary $\mathcal{A}_1$ plays a perfectly simulated decryption for $\mathcal{A}$ unless the cipher ψ^* is decrypted to $\mathtt{K}_1$ and test is returned by the correct answer from the decryption oracle $\mathtt{Tag\text{–}SKEM.Dec}$ for every query. However, the probability of this event is $1/|\mathtt{K}_D|$ since in that case the key $\mathtt{K}_1$ is random and independent of the opinion of the adversary $\mathcal{A}_1$ for each such query.

(i) If $\sigma = 0$, we can know that random key $\mathtt{K}_0$ is used for computing the cipher χ and the view of $\mathscr{A}$ is identical to that in $\mathtt{Experiment}_0$. Accordingly, $\Pr[b' = b \mid \sigma = 0] = \Pr[T_2]$.

(ii) If $\sigma = 1$, we can know that the key $\mathtt{K}_1$ is the correct key embedded in the cipher ψ and the view of $\mathscr{A}$ is equal to that in $\mathtt{Experiment}_1$. Accordingly, $|\Pr[b' = b \mid \sigma = 1] - \Pr[T_1]| \leq q_D/|\mathtt{K}_D|$.

Thus,

$$\begin{aligned}
\mathrm{Adv}^{\mathtt{RCCA}}_{\mathtt{Tag-SKEM},\mathscr{A}_1}(\lambda) &= \left|\Pr[T_0] - \frac{1}{2}\right| \\
&= \left|\Pr[b = b'] - \frac{1}{2}\right| \\
&= \frac{1}{2}\left|\Pr[\sigma' = 1 \mid \sigma = 1] - \Pr[\sigma' = 1 \mid \sigma = 0]\right| \\
&= \frac{1}{2}\left|\Pr[b = b' \mid \sigma = 1] - \Pr[b = b' \mid \sigma = 0]\right| \\
&\geq \frac{1}{2}\left|\Pr[T_0] - \Pr[T_1] - \frac{q_d}{|\mathtt{K}_D|}\right|.
\end{aligned} \tag{16}$$

Hence,

$$\left|\Pr[T_1] - \Pr[T_0]\right| \leq 2\mathrm{Adv}^{\mathtt{RCCA}}_{\mathtt{Tag-SKEM},\mathscr{A}_1}(\lambda) + \frac{q_s}{|\mathtt{K}_D|}. \tag{17}$$

□

Lemma 2 is proved. Next, we show that the adversary $\mathscr{A}$ playing $\mathtt{Experiment}_1$ essentially conducts an IND-RCCA attack on the signcryption scheme DEM; we claim the following.

Lemma 8. *There is a probabilistic adversary $\mathscr{A}_2$, and its running time is equal to that of $\mathscr{A}$, and the following conclusion holds:*

$$\left|\Pr[T_2] - \frac{1}{2}\right| \leq \mathrm{Adv}^{\mathtt{RCCA}}_{\mathtt{DEM},\mathscr{A}_2}(\lambda). \tag{18}$$

Proof. This can be shown by constructing an adversary $\mathscr{A}_2$ who attacks the signcryption scheme DEM. The adversary $\mathscr{A}_2$ simulates the environment for adversary $\mathscr{A}$; their interactions can be described as follows:

(i) **Stage 1:** The adversary $\mathscr{A}_2$ queries receiver's key generation algorithm $(pk_R, sk_R) \longleftarrow \mathtt{Tag\text{-}SKEM.Gen}_R(1^\lambda)$, queries sender's key generation algorithm $(pk_S, sk_S) \longleftarrow \mathtt{Tag\text{-}SKEM.Gen}_S(1^\lambda)$, and sends a public key pair (pk_R, pk_S) to $\mathscr{A}$.

(ii) **Stage 2:** The adversary $\mathscr{A}$ inputs a public key pair (pk_R, pk_S) and makes continuous queries c to a decryption oracle algorithm: $m \longleftarrow \mathtt{Dec}(sk_R, pk_S, c)$. In the end, adversary $\mathscr{A}$ is responded with m or reject symbol $\perp$.

(iii) **Stage 3:** The adversary $\mathscr{A}$ inputs a pair public key (pk_R, pk_S) and sends (m_0, m_1) to the adversary $\mathscr{A}_2$; $\mathscr{A}_2$ queries (m_0, m_1) to the encryption oracle algorithm and then receives a challenge ciphertext χ^*. $\mathscr{A}_2$ runs $(\omega, K_1) \longleftarrow \mathtt{Tag\text{-}SKEM.Key}(pk_R, sk_S)$ and then computes the following.

$$\mathtt{Tag\text{-}SKEM.Enc}_{pk}(\omega, \chi^*) = \psi^* \tag{19}$$

In the end, the adversary $\mathscr{A}_2$ submits the challenger cipher $c^* = (\psi^*, \chi^*)$ to adversary $\mathscr{A}$.

(iv) **Stage 4:** The adversary $\mathscr{A}$ inputs a public key pair (pk_{R_i}, pk_{S_i}) and makes continuous queries $c = (\psi_i, \chi_i)$ to decryption oracle algorithm. Here, we require that adversary $\mathscr{A}_2$ is not admitted to query (pk_S, c^*) to the decryption oracle. However, we admit that the adversary A_2 is admitted to query the decryption oracle algorithm on $(pk_{S'}, C)$ for any $pk_{S'} \neq pk_{S_i}$ and on (pk_{S_i}, C) for any cipher $C \neq C^*$. The adversary $\mathscr{A}_2$ uses (pk_{S_i}, sk_{R_i}) to decrypt the cipher $c = (\psi_i, \chi_i)$. The adversary $\mathscr{A}_2$ runs the decryption oracle.

$$\begin{aligned}
&\mathtt{K}_i \longleftarrow \mathtt{Tag\text{-}SKEM.Dec}(pk_S, sk_R, \psi_i, \chi_i), \\
&m \longleftarrow \mathtt{DEM.Dec}_{K_i}(\chi_i)
\end{aligned} \tag{20}$$

The adversary $\mathscr{A}_2$ answers $\mathscr{A}$'s decryption query $c = (\psi_i, \chi_i)$ with the following:

(a) If $\mathtt{K}_i = \perp$, then $\mathscr{A}_2$ responds to $\mathscr{A}$ with $\perp$.

(b) If $\mathtt{K}_i = \mathtt{K}_1$ and $\chi = \chi^*$, then $\mathscr{A}_2$ responds to $\mathscr{A}$ with $'text'$.

(c) If $\mathtt{K}_i = \mathtt{K}_1$ and $\chi \neq \chi^*$, then $\mathscr{A}_2$ responds to $\mathscr{A}$ with m.

(d) If $\mathtt{K}_i \neq \mathtt{K}_1$, then $\mathscr{A}_2$ uses $\mathtt{K}_i$ to decrypt the cipher χ.

(1) If $m \in \{m_0, m_1\}$, then adversary $\mathscr{A}_2$ sends $'text'$ to adversary $\mathscr{A}$.

(2) Otherwise, adversary $\mathscr{A}_2$ sends m to adversary $\mathscr{A}$.

Here, we notice that the key $\mathtt{K}_0$ chosen by the signcryption scheme DEM's encryption oracle and embedded in the cipher χ^* is randomly chosen and independent.

(v) **Stage 5:** In the end, the adversary $\mathscr{A}$ outputs $b' \in \{0, 1\}$, and the adversary $\mathscr{A}$ also outputs $\sigma' = b'$.

We have described the construction of the adversary $\mathscr{A}_2$. $\mathscr{A}_2$ plays a perfect simulation $\mathtt{Experiment}$ for $\mathscr{A}$; the view of $\mathscr{A}$ is equal to that in $\mathtt{Experiment}_0$ and $\mathtt{Experiment}_1$; hence, we have

$$\left|\Pr[T_1] - \frac{1}{2}\right| \leq \mathrm{Adv}^{\mathtt{RCCA}}_{\mathtt{DEM},\mathscr{A}_2}(\lambda). \tag{21}$$

Putting all the facts together, we have the following conclusion:

$$\begin{aligned}
&\left|\left(\Pr[T_0] - \frac{1}{2}\right) - \left(\Pr[T_1] - \frac{1}{2}\right)\right| \\
&\leq \mathrm{Adv}^{\mathtt{RCCA}}_{\mathtt{Tag-SKEM+DEM},\mathscr{A}}(\lambda) + \frac{q_s}{|\mathtt{K}_D|}
\end{aligned}$$

$$\mathrm{Adv}^{\mathrm{RCCA}}_{\mathrm{Tag-SKEM+DEM},\mathscr{A}}(\lambda) \leq 2\mathrm{Adv}^{\mathrm{RCCA}}_{\mathrm{Tag-SKEM},\mathscr{A}_1}(\lambda) + \frac{q_s}{|K_D|} + \mathrm{Adv}^{\mathrm{CCA}}_{\mathrm{Tag-DEM},\mathscr{A}_2}(\lambda). \quad (22)$$

□

We have proved Theorem 6.

4. Conclusion

We have examined the RCCA security of two representative hybrid signcryption schemes, i.e., SKEM + DEM [3] and Tag−SKEM+DEM [18], in this paper. We proved that the hybrid signcryption scheme SKEM+DEM is RCCA-secure if the signcryption scheme SKEM is RCCA-secure and the signcryption scheme DEM is RCCA-secure. Meanwhile, we showed that the hybrid encryption scheme Tag−SKEM + DEM can be RCCA-secure if the signcryption scheme Tag−SKEM is RCCA-secure and the scheme DEM is RCCA-secure.

Data Availability

Data sharing is not applicable to this article as no new data was created or analyzed in this study.

Conflicts of Interest

The authors declare that no conflicts of interest exist.

Acknowledgments

This paper is supported by the National Key Research and Development Plan of China under Grant No. 2016YFB0800600 and the National Natural Science Foundation of China (No. 61802006; No. 61602061; No. 61672059; No. 61472016).

References

[1] A. W. Dent and Y. Zheng, *Practical Signcryption, a volume in Information Security and Cryptography*, Springer, 2010.

[2] Y. Zheng, "Digital signcryption or how to achieve cost(signature ∝ encryption) << cost(signature) + cost(encryption)," in *Proceedings of the CRYPTO 1997*, pp. 165–179, 1997.

[3] A. W. Dent, "Hybrid signcryption schemes with outsider security (extended abstract)," in *Proceedings of the ISC*, vol. 3650, pp. 203–217.

[4] J. Lee, "Identity-Based Signcryption, IACR Cryptology ePrint Archive, 2002/098," availabel online https://eprint.iacr.org/2002/098.pdf.

[5] M. Barbosa and P. Farshim, "Certificateless signcryption," in *Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS '08)*, pp. 369–372, ACM, March 2008.

[6] R. Nakano and J. Shikata, "Constructions of Signcryption in the Multi-user Setting from Identity-Based Encryption," in *Proceedings of the IMACC 2013: Cryptography and Coding*, pp. 324–343, 2013.

[7] F. Li, B. Liu, and J. Hong, "An efficient signcryption for data access control in cloud computing," *Computing*, vol. 99, no. 5, pp. 465–479, 2017.

[8] S. Sato and J. Shikata, "Lattice-Based Signcryption Without Random Oracles," in *Proceedings of the PQCrypto 2018*, pp. 331–351, 2018.

[9] P. Datta, R. Dutta, and S. Mukhopadhyay, "Functional Signcryption," *Journal of Information Security and Applications*, vol. 42, pp. 118–134, 2018.

[10] Y. Zheng and H. Imai, "Compact and unforgeable key establishment over an ATM network," in *Proceedings of the 1998 17th Annual IEEE Conference on Computer Communications, INFOCOM. Part 1 (of 3)*, pp. 411–418, April 1998.

[11] M. Nikravan, A. Movaghar, and M. Hosseinzadeh, "Correction to: A lightweight signcryption scheme for defense against fragment duplication attack in the 6LoWPAN networks," *Peer-to-Peer Networking and Applications*, pp. 1–18, 2018.

[12] X. Zhou, Z. Jin, Y. Fu, H. Zhou, and L. Qin, "Short signcryption scheme for the Internet of Things," *Informatica. An International Journal of Computing and Informatics*, vol. 35, no. 4, pp. 521–530, 2011.

[13] K. T. Nguyen, N. Oualha, and M. Laurent, "Lightweight certificateless and provably-secure signcryptosystem for the internet of things," in *Proceedings of the 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2015*, pp. 467–474, Finland, August 2015.

[14] S. Belguith, N. Kaaniche, M. Mohamed, and G. Russello, "C-ABSC: Cooperative Attribute Based SignCryption Scheme for Internet of Things Applications," in *Proceedings of the 2018 IEEE International Conference on Services Computing (SCC)*, pp. 245–248, San Francisco, CA, USA, July 2018.

[15] A. Dent, *Hybrid cryptography*, 2004, Available from http://eprint.iacr.org/2004/210/.

[16] Y. Cui and G. Hanaoka, *Applications of Signcryption*, Springer, Germany, 2010.

[17] S. Prakash and A. Rajput, "Hybrid cryptography for secure data communication in wireless sensor networks," *Advances in Intelligent Systems and Computing*, vol. 696, pp. 589–599, 2018.

[18] T. Bjorstad and A. Dent, "Building better signcryption schemes with tag-KEMs," in *Proceedings of the PKC 2006*, pp. 491–507, 2006.

[19] F. Li, M. Shirase, and T. Takagi, "Certificateless Hybrid Signcryption," in *Proceedings of the ISPEC 2009*, pp. 112–123, 2009.

[20] C. Zhou, "Improved certificateless hybrid signcryption scheme," *Application Research of Computers*, vol. 30, no. 1, pp. 273-272, 2013.

[21] R. Cramer and V. Shoup, "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack," in *Proceedings of the CRYPTO 1998*, pp. 13–25, 1998.

[22] R. Canetti, S. Halevi, and J. Katz, "Chosen-ciphertext security from identity-based encryption," in *Proceedings of the Eurocrypt 2008*, pp. 207–222, 2008.

[23] R. Canetti, H. Krawczyk, and J. B. Nielsen, "Relaxing chosen-ciphertext security," in *Proceedings of the CRYPTO*, pp. 565–582, 2003.

[24] R. Canetti, H. Krawczyk, and J. Nielsen, *Relaxing Chosen Ciphertext Security*, 2003, available online at http://eprint.iacr.org.

[25] M. Yoshida and T. Fujiwara, "On the Security of Tag-KEM for Signcryption," *Electronic Notes in Theoretical Computer Science*, vol. 171, pp. 83–91, 2007.

[26] M. Abe, R. Gennaro, and K. Kurosawa, "Tag-KEM/DEM: a new framework for hybrid encryption," *Journal of Cryptology. The Journal of the International Association for Cryptologic Research*, vol. 21, no. 1, pp. 97–130, 2008.

[27] Y. Chen and Q. Dong, "RCCA Security for KEM+DEM Style Hybrid Encryptions," in *Proceedings of the Inscrypt*, pp. 102–121, 2012.

[28] V. Shoup, "On formal models for secure key exchange, IACR Cryptology ePrint Archive,Report 1999/012," Available online http://eprint.iacr.org/1999/012.pdf.

[29] H. Cui, Y. Mu, and M. H. Au, "Signcryption secure against linear related-key attacks," *The Computer Journal*, vol. 57, no. 10, pp. 1472–1483, 2014.

[30] H. Dai, J. Chang, Z. Hou, and M. Xu, "The ECCA security of hybrid encryptions," in *Proceedings of the ISPEC*, pp. 847–859, 2017.

[31] R. Cramer and V. Shoup, "Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack," *SIAM Journal on Computing*, vol. 33, no. 1, pp. 167–226, 2003.

[32] M. Abe, R. Gennaro, K. Kurosawa, and V. Shoup, "Tag-KEM/DEM: A new frame-work for hybrid encryption and a new analysis of Kurosawa-Desmedt KEM," in *Proceedings of the EUROCRYPT*, pp. 128–146, 2005.

[33] J. H. An, *Authenticated encryption in the public-key setting: Security notions and analyses*, 2001, Available from http://eprint.iacr.org/2001/079.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 2143897, 23 pages
https://doi.org/10.1155/2018/2143897

Review Article

Towards Smart Healthcare: Patient Data Privacy and Security in Sensor-Cloud Infrastructure

Isma Masood,[1] **Yongli Wang**,[1] **Ali Daud**,[2] **Naif Radi Aljohani**,[3] **and Hassan Dawood**[4]

[1]*School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210000, China*
[2]*Department of Computer Science and Software Engineering, International Islamic University, Islamabad 44000, Pakistan*
[3]*Faculty of Computing and Information Technology, King Abdulaziz University Jeddah, 21432, Saudi Arabia*
[4]*Department of Software Engineering, University of Engineering and Technology, Taxila 47070, Pakistan*

Correspondence should be addressed to Isma Masood; isma_masood@njust.edu.cn

Received 16 July 2018; Revised 13 October 2018; Accepted 22 October 2018; Published 4 November 2018

Guest Editor: Georgios Kambourakis

Nowadays, wireless body area networks (WBANs) systems have adopted cloud computing (CC) technology to overcome limitations such as power, storage, scalability, management, and computing. This amalgamation of WBANs systems and CC technology, as sensor-cloud infrastructure (S-CI), is aiding the healthcare domain through real-time monitoring of patients and the early diagnosis of diseases. Hence, the distributed environment of S-CI presents new threats to patient data privacy and security. In this paper, we review the techniques for patient data privacy and security in S-CI. Existing techniques are classified as multibiometric key generation, pairwise key establishment, hash function, attribute-based encryption, chaotic maps, hybrid encryption, Number Theory Research Unit, Tri-Mode Algorithm, Dynamic Probability Packet Marking, and Priority-Based Data Forwarding techniques, according to their application areas. Their pros and cons are presented in chronological order. We also provide our six-step generic framework for patient physiological parameters (PPPs) privacy and security in S-CI: (1) selecting the preliminaries; (2) selecting the system entities; (3) selecting the technique; (4) accessing PPPs; (5) analysing the security; and (6) estimating performance. Meanwhile, we identify and discuss PPPs utilized as datasets and provide the performance evolution of this research area. Finally, we conclude with the open challenges and future directions for this flourishing research area.

1. Introduction

The advancement and application of Wireless Body Area Networks (WBANs) are considered key research areas for improving healthcare quality [1]. Pervasive healthcare monitoring provides rich contextual information to handle the odd conditions of chronically ill patients. Constant monitoring and an early medical response not only increase the life quality of elderly and chronically ill people but also help families and parents by providing high-quality healthcare to their young babies and paralyzed children [1–6]. The importance of the WBANs cannot be very promising, as many applications and prototypes are already in progress. For example, some WBANs are dedicated to continuous observation of cognitive diseases such as Alzheimer's, epilepsy, and Parkinson's disease. Another significant advancement in WBANs is the formation of tiny sensors implanted in the human body or integrated into fabric.

While the importance of WBANs in healthcare is indubitable, the amount of data generated by these sensors is huge and demands more resources in terms of computation, memory, communication power, massive storage infrastructure, energy-efficient performance for processing, real-time monitoring, and data analysis [5, 7–18]. Cloud computing shows very promising progress in hosting the aforementioned resources as services over the Internet [10, 19, 20]. At present, IT professionals extend cloud computing to reduce the complexity and utilization of WBANs' resources. This extension is called S-CI [8, 21–24]. Figure 1 shows a typical S-CI for PPPs monitoring and access.

In S-CI, a large amount of patient data are collected from WBANs and transmitted to cloud servers for scalability, real-time accessibility, storage, and processing capability. Therefore, patient data privacy and security are more challenging due to the distributed environment [25–27]. The motivation

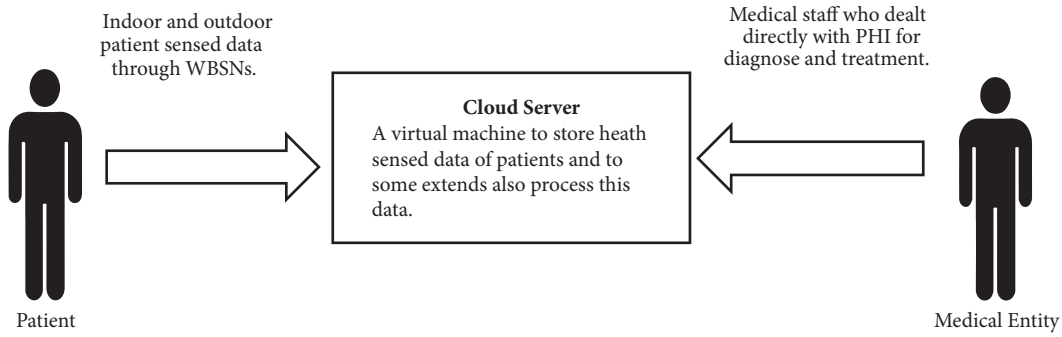


Figure 1: S-CI for PPPs monitoring.

for this study is to investigate and organize the existing techniques of S-CI so that the research community can address the vulnerabilities and limitations of PDPS and to identify the need for further work in this domain. Significantly, a number of studies have utilized patient physical parameters (PPPs) as their dataset. However, other studies refer to their dataset as medical data, personal health information (PHI), or electronic health records (EHRs), not clearly mentioning which are PPPs. In this review, we will identify and organize existing solutions to patient data privacy and security in S-CI.

Nowadays, the use and significance of the S-CI in the healthcare domain cannot be denied [28]. At a commercial level, large numbers of applications [8, 29] are already in service: ubiquitous healthcare, Google health, Microsoft Health Vault, and so on. However, the distributed environment of S-CI opens new challenges for patient data privacy and security: data integrity, confidentiality, patient participation in data control, data purpose specification for use limitation, audit control, availability, scalability, data transmission security, network security, source authentication, and so on [30–35]. In 2010, Almedar et al. [1] evaluated the literature to show the state of the art regarding how wireless sensor technology is improving healthcare conditions for patients at home and highlighted issues to bear in mind for future development. Similarly, in 2012, two studies were published. Kumar et al. [32] reviewed the literature to identify security and privacy issues in medical sensor-based applications, while Ameen et al. [36] reviewed the literature on wireless sensor networks and raised major concerns relating to social implications such as security and privacy. Furthermore, in 2013, Alamri et al. [8] investigated sensor-cloud architecture in several applications and discussed the emerging opportunities to handle more complex scenarios in the real world through S-CI. Presently, no comprehensive and organized study is available to address patient data privacy and security in S-CI. The literature shows some significant solutions in different application areas such as mobile healthcare [37–41], electronic healthcare [42–45], health data management [2], and health data aggregation [46]. Hence, the following are the major contributions summarized in this study:

(1) Detailing the state-of-the-art existing techniques, which gives a roadmap for this innovative research area.

(2) Providing a classification scheme for existing techniques in order to identify in-depth investigation and limitations of each application domain for better future extension.

(3) Providing a generic six-step framework for privacy and security of PPPs in S-CI.

(4) Highlighting future directions, with useful recommendations.

The rest of the paper is organized as follows: Section 2 presents the basic concepts and terminologies, Section 3 explained the method, Section 4 presents the results, Section 5 gives the performance estimation of the techniques, finally, Section 6 highlights the future directions and useful recommendations for this area of interest, and Section 7 concludes this study.

Table 1 provides a list of the abbreviations and notations used in the study.

2. Basic Concepts and Terminologies

In this section, we will discuss some important concepts and terminologies relating to patient data privacy and security in S-CI. The concepts and terminologies have evolved according to their level of complexity.

2.1. UBUNTU Enterprise Cloud. The general concept of cloud computing is that it is an Internet-based service provided by a third party. This is true for a public cloud, yet there is another type of cloud computing known as private cloud computing whereby an enterprise or an organization hosts its own private cloud. The UBUNTU enterprise cloud is a cloud computing technology that allows an enterprise to build a private cloud on their environment. UBUNTU allows a centrally managed resource pool behind a firewall on a local network. The chief benefits of this technology are as follows: (1) better use of server resources; (2) provision of new cloud images in a short period of time; (3) allowing bursting to public cloud (e.g., Amazon EC2), giving an added level of flexibility and also driving down building and maintenance costs [56].

2.2. Amazon EC2 IaaS Platform. The Amazon Elastic Compute Cloud EC2 is an Amazon web service used to access software, servers, and storage resources across the Internet

Table 1: List of Abbreviations and Notations.

Sr. no	Abbreviations	Description
1	CC	Cloud Computing
2	WBANs	Wireless Body Area Networks
3	S-CI	Sensor Cloud Infrastructure
4	PPPs	Patient Physiological Parameters
5	PDPS	Patient Data Privacy and Security
6	PHI	Personal Health Information
7	EHRs	Electronic Health Records
8	AWS	Amazon Web Services
9	Eucalyptus	Elastic Utility Computing Architecture Linking Your Programs to Useful Systems
10	SP	Social Point
11	CS	Cloud Server
12	WBSs	Wireless Body Sensors
13	TA	Trusted Authority
14	HA	Healthcare Authority
15	SK_p	Secret Key
16	AR_p	Access Structure
17	CSP	Cloud Service Provider
18	AES	Advanced Encryption Standard
19	DES	Data Encryption Standard
20	IDEA	International Data Encryption Algorithm
21	MD5	Message Digest 5
22	SHA	Secure Hashing Algorithm
23	ECG	Electrocardiograms
24	PR	Pulse Rate
25	RR	Respiratory Rate
26	BT	Body Temperature
27	SpO_2	Oxygen Saturation
28	GL	Hyperglycemia
29	BP	Blood Pressure
30	PS	Personal Server

on a self-service basis. Amazon EC2 provides scalability, pay-per-use computing capacity, and an elastic scale in both directions [11].

2.3. Eucalyptus System. "Eucalyptus" stands for "Elastic Utility Computing Architecture Linking Your Programs to Useful Systems." Eucalyptus is free and open-source software for developing Amazon web services (AWS) compatible with the hybrid and private cloud-computing environment. Eucalyptus facilitates storage, pooling the computing and network resources dynamically. The Eucalyptus system announced a formal agreement with AWS in March, 2012. The main objectives are to provide (1) a vehicle to extend the utility model of cloud computing; (2) an experimentation vehicle for development and a debugging platform for public clouds before buying original software; (3) a homogenized IT environment for public clouds; and (4) a basic platform for the open-source community (e.g., Linux) [57].

2.4. SNIA's Cloud Data Management Interface. SNIA's [58] cloud data management interface is a standard for cloud data storage. This standard proposed an interface for managing and accessing data cloud storage. The "cloud data management interface' is broadly acceptable architecture that specifies a framework for data access, data management operations, data object definitions, access control, and logging specifications for cloud environment. However, this standard lacks specifications for security and privacy [59].

2.5. Social Spot. According to Zhang et al. [46], a social spot (SP) is a predeployed local gateway that is fully equipped for high storage and powerful communication. The PHDA [46] scheme proposed for cloud-assisted WBANs used these social spots for the collection of outdoor PPPs. The total L numbers of SP are located at intersections or 'spots' where patients frequently visit. These spots are located according to their behaviour. SP is responsible for collecting PPPs directly sensed data from each patient via a cloud-assisted WBAN. Finally, SPs upload this aggregated data at cloud servers.

2.6. Cloud Server. A cloud server (CS) is a virtual machine that stores large amounts of health-sensed data from patients

and, to some extent, processes that data. For example, this could be ECG data to produce useful information that can be accessed by doctors or other medical staff, through query, for diagnosis [7].

2.7. Outdoor and Indoor Patient. The term 'outdoor' refers to those patients who are equipped with wireless body sensors (WBSs) for healthcare monitoring and to transmit PPPs to CS through social spots or social networks (explained below). Similarly, 'indoor' patients are those who are equipped with WBSs and monitored in their home, hospital, and so on. PPPs are transmitted to CS by personal handheld devices or laptops [60].

2.8. Trusted Authority. A trusted authority (TA) is a trusted, powerful, and rich storage entity. A TA bootstraps the whole system in the initialization phase. According to Zhang et al. [46], a TA can be a certified hospital in the real world that is responsible for the management of health data. In the PHDA [46] scheme, initially a TA generates a secret key for legitimate users and certificates for further authorization. After authorization of legitimate users and health data aggregation, a TA can decrypt data for diagnosis. In addition, a TA repels malicious user attacks in PHDA. In ESPAC [42], a TA generates public and secret key parameters. A TA is responsible for issuing keys and revoking, updating, and granting authorization rights to individuals based on their roles and attributes. For storage, a TA maintains an index table to store the location of the distributed storage server. Lounis et al. [2] introduced a healthcare authority (HA) as a TA in their scheme for healthcare data management. An HA generates a secret key SK_p and builds an access structure AR_p that patients use for health data encryption.

2.9. Medical Entity. Medical entities cover those staff who dealt directly with PPPs and PHI for patient diagnosis and treatment, for example, doctors, nurses, and medical assistants. These entities access PHI primarily to perform some operation or transfer to third party for secondary use [48, 61].

2.10. Encryption Technology in Cloud Computing. The hassle-free management and encouragement attract a large number of users towards untrusted servers. A CS may leak information to unauthorized parties. Therefore, all data needs to be transmitted in ciphertext mode to ensure data confidentiality and integrity against untrusted cloud service providers (CSP) [62]. The transmitted data are encrypted so that authorized bodies understand it. The three main encryption technologies that are utilized and used in cloud computing [63] are set out as follows.

Symmetric encryption, also known as "private key cryptography" [63], is a basic and the most trustworthy method to secure online transmission. A private key preserves arbitrarily created words or mix of letters connected as a secret key to change the message particularly. For example, let the password be ABC and for the encryption, algorithm advances this password by five places; then, the new password will be EFG, which is obviously simple, like the ABC password, but difficult to hack. This encryption technique can be used as a "stream cipher" [63] or "block cipher" [63], directly proportional to the quantity of data encrypted or decrypted over time. A "stream cipher" [63] performed encryption character by character at a time, while a "block cipher" [63] processed a fixed amount of information. Traditional algorithms for symmetric encryption are "Advanced Encryption Standard (AES)" [63], "Data Encryption Standard (DES)" [63], and "International Data Encryption Algorithm (IDEA)" [63].

The asymmetric method, or simply "public key cryptography" [63], is that two paired keys are used together to encrypt and decrypt messages to keep them secure during transmission. When talking about data transfer for large businesses or organizations, this method is considered to be more enhanced than symmetric encryption. According to Microsoft, "you do not have to worry about passing public keys over the Internet (the keys are supposed to be public). However, asymmetric encryption is slower than symmetric encryption. It requires far more processing power to both encrypt and decrypt the content of the message" [63].

The generation of special fixed-length passwords for a message, signature, or set of data is called hashing encryption. In this type of encryption, hash functions are used to protect information. The main advantage of this method is that the slightest change in information makes a completely new hash function that is incredibly difficult to hack and, once the message is secured, it cannot be read or altered by any process: "This means that even if a potential attacker were able to obtain a hash, he or she would not be able to use a decryption method to discover the contents of the original message. Some common hashing algorithms are Message Digest 5 (MD5) and Secure Hashing Algorithm (SHA) [64]."

2.11. Pairing-Based Cryptography. The basic concept of pairing-based cryptography is pairing between elements of two cryptographic groups and mapping this pairing to a third group $\boldsymbol{e: G_1 \; x \; G_2 \text{ -> } G_T}$, for the construction or analysis of cryptographic systems. According to academic research [65], the common definition used for pairing-based cryptography is as follows: Let $\boldsymbol{G_1, G_2}$ be additive cyclic groups of prime order q and $\boldsymbol{G_T}$ another order of prime q for multiplicativity. The pairing map of $\boldsymbol{e: G_1 \; x \; G_2 \text{ -> } G_T}$ satisfies the following properties:

Bilinearity:

$$\forall \mathscr{P}, \mathscr{Q} \in \mathscr{G}1, \; \forall \alpha, \mathscr{b} \in \mathscr{Z} \; * \; q, \quad e\left(\mathfrak{a}\mathscr{P}, \mathscr{b}\mathscr{Q}\right)^{ab} \tag{1}$$

Nondegeneracy:

$$P \in G1, P \neq 0 \Longrightarrow < e\left(P, P\right) i = G2\left(e\left(P, P\right) \text{ generates } G2\right) \tag{2}$$

$$\mathscr{P} \neq 0 \; \mathfrak{e}\left(\mathscr{P}, \mathscr{P}\right) 6 \neq 1$$

Computability:

$$\text{e is efficiently computable.} \tag{3}$$

If the first two groups use the same group (**i.e.,** $\mathbf{G_1}=\mathbf{G_2}$), then this type of pairing is known as symmetric. This classification of pairing can be further divided into three types: (1) $\mathbf{G_1}=\mathbf{G_2}$; (2) $\mathbf{G_1}\neq\mathbf{G_2}$, with efficient computable homomorphism ϕ: $\mathbf{G_2}=\mathbf{G_1}$; and (3) $\mathbf{G_1}\neq\mathbf{G_2}$ nonefficient computable homeomorphisms between $\mathbf{G_1}$ and $\mathbf{G_2}$ [66].

3. Method

In this study, we have conducted a literature review to find techniques proposed for PPPs privacy and security in S-CI. We categorized and organized these techniques according to the applications of the healthcare domain. The outcome of the study will be beneficial for the research community who are involved for the betterment of patient data privacy and security in S-CI.

3.1. Inclusion and Exclusion Criteria. Literature addressed large number of studies on privacy and security of images, cloud storage-based patient data, sensor networks, wireless communication, cloud-assisted wireless body area network, and in-home patient monitoring. This study only included those empirical published studies, which have been peer-reviewed in journals, conferences, and workshops published up to two quarters of 2018. This inclusion criterion is based upon the evidence provided by the pilot study. Those studies not explicitly providing techniques for the privacy and security of PPPs or supporting any other area of wireless body sensors rather than cloud-assisted wireless body area network were excluded. We also excluded books, technical reports, and project thesis studies based on expert and physiological opinions.

3.2. Search String. Literature addressed large number of studies on privacy and security of images, cloud storage-based patient data, sensor networks, wireless communication, and in-home patient monitoring. We used these results in finalizing the pilot study. Initial search by applying general string at selected databases for pilot study was as follows:

> *Patient AND Medical AND Wireless Body Sensors AND Cloud Computing (Privacy OR Security)*

There are many diversified terms used to address patient data privacy, security, and body sensors for patients in literature. It was a challenge to generate a valid string for targeting relevant studies. Therefore, we used the major terms of our selected primary studies search from the aforementioned string to formalize a search string for our final study. As a result, the following search string was produced:

> *((Healthcare" OR "Patient" OR "Medical" OR " eHealth" OR "mHealth" OR "Health data" OR "Mobile Computing" OR "Mobile Device" OR "Medical Care System" OR "Mobile Cloud" OR "E-Healthcare System") AND ("Wireless Body Area Network" OR "WSN" OR "Wireless Sensor Network") AND ("Cloud" OR " Cloud Computing" OR " Cloud-assisted" OR "Private Cloud" OR "Sensor Cloud" OR "Cloud Storage") AND ("Privacy" OR "Security"))*

3.3. Data Extraction and Analysis. At this stage of conducting phase, data of selected primary studies from previous phase was extracted. To carry out data extraction more efficiently, forms were designed in MS word. These forms also help in consistency of data extraction. These data extraction forms were evaluated in our pilot study. It is difficult to set values of all properties prior to data extraction. These properties are totally dependent on the papers and their contents. However, the extracted properties with relevant questions are mentioned. Data synthesis involves collecting and summarizing the results of the included primary studies. Synthesis can be descriptive (nonquantitative). However, it is sometimes possible to complement a descriptive synthesis with a quantitative summary. The extracted data from data extraction forms were recorded on Excel sheets. This really helped us to find trends, consistency, and relevant similarities for analysis of data.

4. Results

4.1. S-CI Process for Patient Data Privacy and Security. In this section, we outline our six-step generic framework for S-CI to achieve PPPs privacy and security. This framework does not follow any particular research method of a study. We give the basic steps that we adopted to ensure patient data security and privacy in S-CI. The main purpose of our framework is to help readers to understand the process more clearly and easily. Figure 2 is a block diagram showing patient data privacy and security in S-CI. Firstly, particular techniques identify relevant system entities before defining method. Meanwhile, PPPs were accessed as dataset and utilized to validate the technique. Finally, security and performance analysis of the selected parameters was performed for evaluation of PDPS.

4.1.1. Selecting the Preliminaries. Almost all studies define a set of preliminaries before proposing a technique. These preliminaries are the basic concepts of the proposed solution. Preliminaries serve as the baseline, and the entire technique for PPPs security and privacy stems from them. For example, bilinear pairing [42], pairing-based cryptography [55], hash function [38], attribute-based encryption [38], and access tree [39] are some important preliminaries in S-CI.

4.1.2. Identify the System Entities. The majority of studies have identified system entities before proposing a solution or technique. The system entities are those such as trusted authority, cloud service provider, registered user, data-access requester, health cloud, social cloud, data owner, user, healthcare provider, healthcare analyzer, hospital, key generation centre, IoT medical sensor, mobile device, emergency family contacts, key management centre, doctor, medical staff, body sensor, patient, and social spot, which are some significant entities identified for different techniques. One should identify the relevant set of system entities based on the relevant application area and technique.

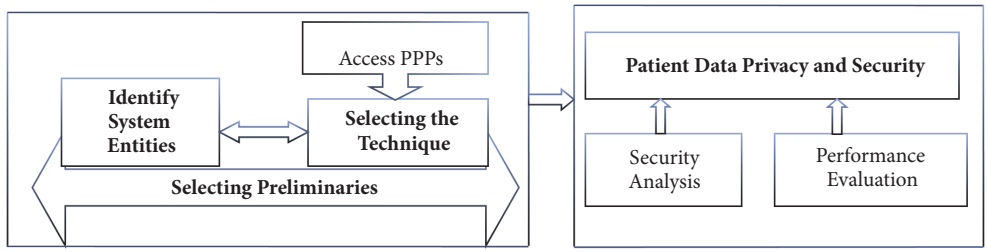


Figure 2: Patient data privacy and security in S-CI.

4.1.3. Selecting the Technique. In this study, we categorized S-CI-based technique for patient data privacy and security into 10 types: (1) multibiometric key generation; (2) pairwise key establishment; (3) hash function; (4) attribute-based encryption; (5) chaotic maps; (6) hybrid encryption; (7) Number Theory Research Unit; (8) Tri-Mode Algorithm; (9) Dynamic Probability Packet Marking; and (10) Priority-Based Data Forwarding. All techniques were proposed for specific application areas such as m-healthcare, e-healthcare, health data aggregation, and health data management. The primary purpose of all the techniques is to ensure PPPs privacy and security for S-CI. Every technique has its pros and cons, and before selecting one, all possible alternatives should be borne in mind. However, attribute-based encryption is the most widely adopted technique for this area of interest.

4.1.4. Access PPPs. The core of this study is to organize the techniques available for PPPs privacy and security in S-CI. Therefore, every study concerns PPPs through WBANs or medical sensors. Common PPPs, accessed for real-time monitoring and early diagnosis, are Electrocardiograms (ECG), pulse rate (PR), respiratory rate (RR), body temperature (BT), oxygen saturation (SpO_2), hyperglycemia (GL), blood pressure rate (BP), and so on. One should access PPPs according to the needs of the solution and based on the condition and type of the patient (indoor or outdoor).

4.1.5. Security Analysis. Almost every study had performed security analysis to show the strength of the techniques against security attacks. For example, analyses of some common security requirements include data confidentiality, fine-grained access control, collusion resistance, patient-centered access control, message integrity, denial of service (DoS) attack, prevention of ciphertext-only attack, patient privacy, patient control, source authentication, dynamic data operation, audit control, attribute revocation, cloud reciprocity problem, availability, scalability, identity privacy, impersonation attack, resistance to forgery attack, replay attack, man-in-the middle attack, nonrepudiation, known-key security, signature unforgeability and anonymity, transmission continuity, authorization, and network security.

4.1.6. Performance Evaluation. Many different ways have been adopted to evaluate the performance of the techniques. The most common parameters for evaluation are communication cost, computation cost, storage cost, encryption/decryption time, and key generation time.

4.2. Patient Data Privacy and Security in the Sensor-Cloud Infrastructure. In this section, we discuss the various application areas of healthcare in which patient data privacy and security for S-CI have been addressed. Next, we list the pros and cons of existing techniques to ensure patient data privacy and security in S-CI. Lastly, we follow the taxonomical details of these techniques to provide a comprehensive summary of each.

Figure 3 shows the evolution of the types of techniques used to handle patient data privacy and security in S-CI in chronological order. We can see that attribute-based encryption (ABE) is the most used technique in three main application areas: mobile healthcare, e-healthcare, and health data management [6].

The three chief application areas of S-CI in which patient data privacy and security are addressed are set out as shown in Figure 3.

4.2.1. Mobile Healthcare. Mobile healthcare, or m-health, technology [67, 68] is a rapidly emerging factor facilitating healthcare for better and more efficient services. M-health with cloud computing includes offloading benefits such as reliability improvement, performance improvement, energy savings, ease of software development, and better exploitation of contextual information [24, 69]. For instance, Figure 4 shows tremendous achievements by m-health to aid healthcare services through technology in bidirectional perspective (customers and providers): mobile-enabled EHRs, patient portals, secure text messaging, patient monitoring devices, and telemedicine. While adopting S-CI in mobile computing, new vulnerabilities affect patient data privacy and security. The following nine techniques are proposed for mobile healthcare to solve issues for patient data privacy and security in S-CI.

Multibiometric Key Generation (M-BKG). A secure cloud-based framework is proposed for mobile healthcare using WBANs [37]. In this framework, the author presented a two-fold solution: (1) intersensor communication secured by a multibiometric key generation scheme; (2) secure storage of EMRs on a hospital community cloud to preserve patient privacy. The framework adopted dynamic reconstruction of metadata [70] to secure patient privacy. It is claimed to not

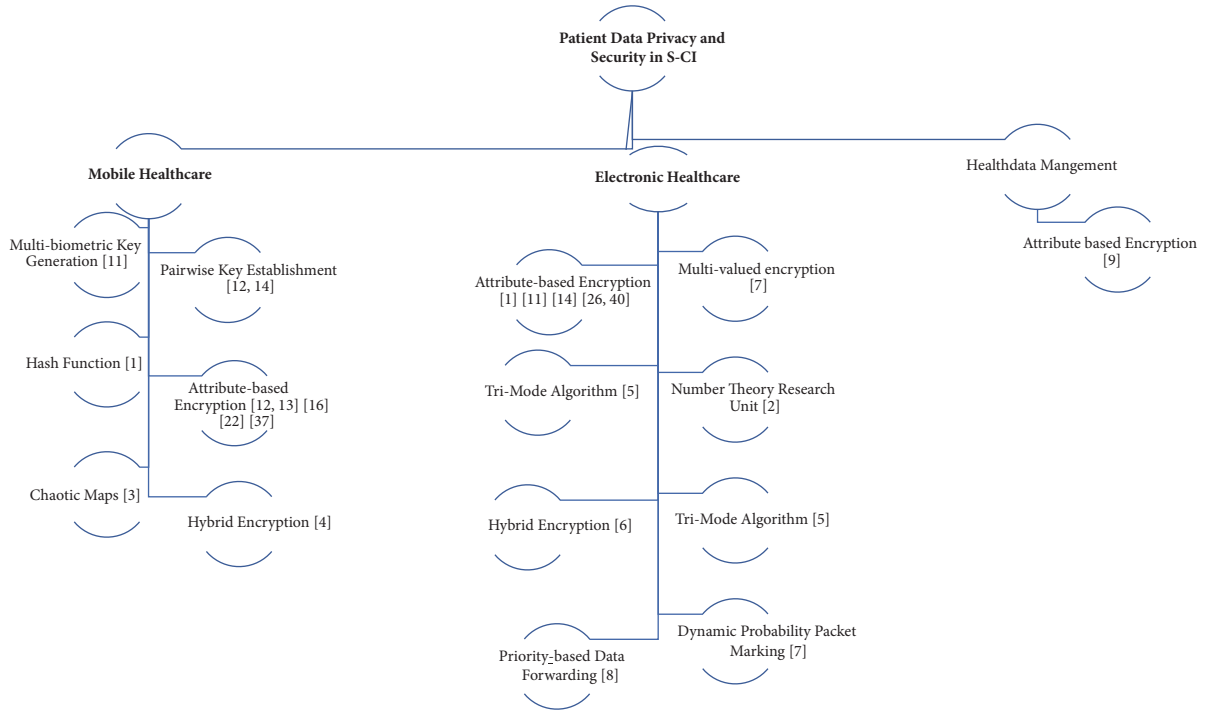


Figure 3: Taxonomy of the study.

Figure 4: M-healthcare services to aid healthcare.

only serve as guidance on privacy and security specifications for SNIA but also assist designers in developing privacy-preserving database schemas in order to store cloud metadata. A fair balance between user privacy, administration rights and roles, limiting the modification requirement in order to make the privacy technique fast and error-free, and cost efficiency in terms of computational resources without any compromise regarding information loss is one of the worthy research goals of this framework. To protect the metadata items of users in clouds, first the metadata items are segregated and stored in the cloud's database. The multibiometric scheme of this framework used a fusion of biometric and two PPPs values, ECG and EEG.

The purpose of the multibiometric scheme is to generate a long key to obtain a secure and random key. First, the scheme performs feature selection for secure intersensor communication. The features are extracted and quantized from EEG and ECG signals using discrete wavelet transform (DWT). WBSs and personal servers (PS) communicate at a sample rate of 125Hz within 5 seconds. In the second step, a key is generated on receiving data blocks of ECG and EEG sensors signals by applying a KeyGen algorithm. Two keys of 160 bits are generated by KeyGen. These keys are concatenated horizontally to generate a 320-bit long key. On receiving compressed blocks from each sensor, common blocks are extracted. The construction of the matrix uses extraction. The Hamming distance is used to measure the elements of the matrix from the ith block of sensor 1 to the jth block of sensor 2. When sensor node 'a' (SN_a) wishes to communicate with sensor node 'b' (SN_b), SN_a sends a 'Hello' message to SN_b with its ID in m1:

$$m1 : \forall SNa\epsilon (SNIA) : SNa \longrightarrow SNb : (IDSNa, Hello, nonce). \tag{4}$$

For calculating pairwise keys of ECG and EEG,

$$K1SNa, SNb = HMAC (CalculatedECG\ values |IDsna| IDSNa) \tag{5}$$

$$K2SNaSNb = HMAC (Calculated\ \ EEG\ values |IDSNa| IDSNb. \tag{6}$$

Finally, in m2, SN_b sends its ID with encrypted data and MAC to SN_a:

$$m2 : \forall SNb \longrightarrow SNa : IDSNa, EKSNa, SNb \{IDsnb, Data\}, MACksna, SNb \left(ID_{SNb}, Data, nonce\right). \tag{7}$$

This process consists of the following steps:

(a) Vertical segregation: In a cloud database, metadata items are stored by vertical segregation according to level, for example, context level, purpose level, and attribute level as first, second, and third, respectively.

(b) Attribute association: In this step, the individual attributes are associated with one member of attribute-type segregation level.

(c) Sensitivity parameterization: The segregated attributes are categorized according to the sensitivity parameterization (SP) classes as exclusively private (XP), partially private (PP), or nonprivate (NP).

Exclusively private data items are kept confidential in all circumstances. The sensitivity class XP is divided into two (ascending from 1 to 2) sensitivity levels. An XP class data item is said to be at level 2 if it fails to preserve a cloud user's privacy when disclosed alone, while XP data items are said to be at level 1 if they are disclosed with other attributes of SP classes XP/PP. However, partially private data is not confidential, yet it needs to protect integrity. Like exclusively private data items, private data items are divided into two levels (ascending from 1 to 2). Table 2 is a summary of the multibiometric key generation technique.

Pairwise Key Establishment. In contrast with the previous technique, Zhou et al. [40] proposed a scheme utilizing the body symmetric structure with Bloom's symmetric key construction (Table 2). Due to the symmetrical structure of the body, WBANs such as ECG and EEG are deployed symmetrically for patients. Patients with the same disease can create a social group, for communication. However, patients with a different disease are not allowed to communicate, for the preservation of privacy.

Members of a social group of patients with the same WBANs have the same sensor placement on their body. For N number of patients P_s $(s = 1, 2, \ldots, N)$ with the same disease in a social group, their connected data sinks are D_S $(s = 1, 2, \ldots, N)$. Pairwise key establishment for privacy key management is carried out in three steps. In step 1, a set of body sensors $BSR_{s,k}$ *(k* $\mathcal{E}$ $\{1, \ldots, N_s\}$*[23])* is deployed on the patient body for a specific disease. The physician uses a symmetric matrix Dp_s to store information of the symmetrical body sensors' positions. For example, the ECG for position '*CHEST*' to pair in symmetric elements in private matrix Dp_s is

$$DPs\left(j; i\right) = \frac{1}{4} H0\left(LocBSRs; k\right) \tag{8}$$

where LocBSNs;k DPs $(i; j)(i \neq j\hat{}i; j$ $\mathcal{E}$ $\{1; \ldots; \lambda + 1\})$ denotes each body sensor position at patient body P_s and the items on the matrix are located at the intersection of *i-th* row and *j-th* column D_s.

In step 2, the patient P_s's block location for each data sink Ds_s is accessed by GPS. The location information of the patient is represented as

$$DPs\left(i; j\right)\left(i = j^i; j\varepsilon\left[1; \lambda + 1\right]\right) = H1\left(LocPs\right). \tag{9}$$

In step 3, the data sink is calculated with initial key material matrix as UPs $=(D_{Ps}G_{Ps})^T$.

Table 2: Summary of Multibiometric Key Generation Technique in m-health.

Technique Ref. No.	M-BKG [37]
Year	2014
Main Idea	Patient data privacy and communication security would increase users' confidence at remote healthcare systems.
PPPs	ECG, EEG
Findings	A framework for cloud-based technique for mobile healthcare that securely perform intersensor communication with patient data privacy and security
Controller	Server
Patient Mode	Indoor/outdoor
Emergency Management	No
Limitation	Only key generation based on ECG and EEG signals.

Table 3: Summary of Pairwise Key Establishment Technique in m-Health.

Technique Ref. No.	4S [40]
Year	2015
Main Idea	A cloud-assisted m-healthcare social network to facilitate security and privacy of patient's data in location- and time-based attacks.
PPPs	ECG, EEG
Findings	A secure cloud-assisted WBANS-based privacy-preserving key management scheme, pliant to mobile attacks in m-healthcare for patients with the same diseases in social group.
Controller	Patient
Patient Mode	Indoor/outdoor
Emergency Management	No
Limitation	Pairwise key establishment is limited to a group of patients sharing the same disease.

In step 4, the private key rs $\mathcal{E}$ Gp(s = 1; 2; . . . ; N) is selected for each data sink DS_s for computing blinded key matrix:

$$U^{rs} p_s \; as \; U^{res} P_s \;\; \left(UP_s \; (s = 1; 2; \ldots; N)\right). \tag{10}$$

In step 5, the i and j intersection of sensor data implements pairwise key establishment with respect to Bloom's symmetric key construction as

$$K(I, j) = U^{res} p_s(i)\, GPs(j) = U^{res} p^s(j)\, Gp_s (i) = K(j; i). \tag{11}$$

The summary of pairwise key establishment in m-health is given in Table 3.

Hash Function. In the same year, wireless sensor networks and cloud computing (WSNCC) [47] were proposed to help, manage, and access sensor data in a cloud-computing environment by efficient processing, communication, and security. The conceptual architecture used Secure Hash Algorithms such as *SHA-224, SHA-256, SHA-384,* and *SHA-512* for message integrity. Symmetric key cryptography is used to provide data confidentially and to maintain the availability of data at all times. Meanwhile, cloud computing supports data with redundancy techniques. Furthermore, the framework is claimed to reduce transmission traffic bandwidth requirements, promote data security, efficient cloud storage, and processing, and reduce cost. Table 4 is a summary of hash function-based techniques.

Attribute-Based Encryption. In 2015, Guan et al. [38] proposed a Mask-Certificate attribute-based encryption (MC-ABE) scheme for secure data transfer. The aim of the study was to perform secure transmission and storage of PPPs (ECG, EEG) with fine-grained policies, privacy, and access control. This novel outsourcing encryption scheme provides patient data privacy and security in S-CI. This consists of a total of seven algorithms: *Setup, KeyGen, CerGen,* $Encrypt_{DO}$, $Encrypt_{ESP}$, $Decrypt_{DSP}$, and $Decrypt_{DR}$. The data owner (DO) encrypts M with algorithm $(Encrypt_{DO}$ *(PK, M, K)-> MM)* for outsourcing, in which a signature is used to mask M. Then, the encryption service provider (ESP) with algorithm $(Encrypt_{ESP}$*(PK, s, T, MM)-> CT)* completes the encryption phase. The encrypted data is stored with a storage service provider (SSP). The requester's data access request is sent to the TA for verification by generating a key with algorithm *(KeyGen(MK, S)-> SK)*. The TA selects a unique value to mask the certificate for the data requester (DR). Then, the TA computes SA with algorithm *(KeyGen(MK, S)->SK),* using the DR attribute set. After this, the certificate is sent to DR and the SK is sent to DSP. Meanwhile, DP receives CT from SSP. Once DR has the certificate, it performs decryption to get M with algorithms $(Decrypt_{DSP}$ *(SK, CT)->MM)* and $(Decrypt_{DR}$*(M.signature, MCert)->M)*.

In 2016, Guant et al. [39] extended their own MC-ABE scheme into another novel mechanism to secure access control. The mobile-based scheme collects PPPs from S-CI in large amount. To maintain privacy and security for mobile

Table 4: Summary of Hash Function Techniques in m-Health.

Technique Ref. No.	WSNCC [47]
Year	2015
Main Idea	Fast and reliable transmission required for cloud-based WSN data.
PPPs	BP, HR, ECG, EGG, medical images
Findings	A model to manage and access sensor data efficiently in processing, communication, and security perspective.
Controller	Server
Patient Mode	Indoor/outdoor
Emergency Management	No
Limitation	Only conceptual architecture, need real case scenarios validation.

computing is a big challenge. In this mechanism, a specific signature is designed to mask the plain text. This masked data is securely outsourced on cloud servers. For access control, an authorization certificate, based upon signature and related privilege items, is constructed. A unique value is selected to mask the authorization certificate of each data receiver. MC-ABE-based system provided access control for S-CI. Meanwhile, the proposed scheme had lower computational and storage costs than other models.

Recently, in 2017, Huang et al. [41] preserved the data privacy and security of health and social data with fine-grained access control. The authors claimed that fusion of health data with social data in smart cities is challenging patient data privacy and security. The mobile healthcare social network (MHSN) scheme is based on attribute-based encryption and identity-based broadcast encryption. The basic aim in the system setup phase is that the central authority runs a setup algorithm to select a bilinear pair map e: $G_1 \times G_2 \rightarrow G_T$ and chooses a maximum number of receivers, N. Meanwhile, in this cryptographic phase, a hash function selects a public key, PK, and a master key, MK, is selected. In the key generation phase, a central authority *AKeyGen* algorithm makes a random selection of a unique key against each user. A secure health and social data sharing collaboration scheme is proposed to preserve patient data privacy. For secure sharing of health and social data, the data is encrypted and decrypted with independent algorithms. Performance analysis and comparison show that MHSN is more efficient and secure than other schemes.

Similarly, He et al. [48] proposed a fine-grained and lightweight data access control (FLAC) scheme for WSN-integrated cloud computing (WCC). In the WCC environment, sensors and mobile devices are weaker nodes in terms of data storage and computing capacity. The aforementioned weakness of the WCC challenge is patient data confidentiality, integrity, and access control, as handling ciphertext policy attribute-based encryption (CP-ABE) and attribute-based encryption (ABS) is a tough job for lightweight devices. To facilitate the computation overheads, FLAC provides secure outsourcing computation of CP-ABE and ABS operations. First, the network controller (NC) generates PK, MK, and system attributes. Then, a sensor node generates intermediate ciphertext parameters and an encryption signature and sends it to the Encryption-Signature Proxy Server (ESPS). The ESPS performs the intensive operations and generates ciphertext CT and signature α. Finally, the ESPS uploads CT and α to the cloud server. Meanwhile, FLAC claims standard security assumptions, collusion resistance, and anonymity in WCC. Table 5 is a summary of the attribute-based encryption technique in m-health.

Chaotic Maps. Furthermore, in 2016, Li et al. [49] proposed an architecture for secure continuous monitoring of patients, based upon chaotic maps. This has five roles for participation in the system: the patient (P), the doctor (D), the healthcare centre (HC), the medical caregiver (MC), and a trusted medical cloud centre (C). Before accessing system, every participant has to register with C to get Chebyshev chaotic map-based specific certificates.

In the first scenario, patient (P) visits HC for a health check, and HC is responsible for uploading P's medical report at C. In the second scenario, P uploads his/her PPPs from WBANs to C using a personal mobile device. In the emergency monitoring application, the MC is allowed to access the uploaded data in order to treat the patient at that time while, in normal situations, when P visits hospital for treatment, D can download his/her data from C. The security analysis of attack model suggests that an attacker may guess such a low entropy password easily. However, to guess a secret parameter such as a certificate is not computationally feasible in polynomial time. Table 6 is a summary of the chaotic map technique.

Hybrid Encryption (Asymmetric/Symmetric Encryption). Recently, Hu et al. [50] proposed an intelligent and reliable IoT scheme for the sensor and cloud computing environment to secure elderly patients' privacy. The proposed scheme collects PPPs through their mobile devices. Seven entities are used in this scheme: elder (E), cloud (C), hospital (H), key generation centre (KGC), IoT medical sensor (MS), mobile device (MD), and emergency family contacts (EFC). Initially, the elderly people and the hospital need to register themselves in KGC via a secure communication channel. The elderly people visit hospital for a medical inspection and medical staff uploads their inspection report to the cloud. IoT-based medical sensors collect PPPs and send to the medical device after a set period. The mobile device is responsible for uploading the PPPs to the cloud. The cloud

Table 5: Summary of Attribute-Based Encryption Technique in m-Health.

Technique Ref. No.	MC-ABE [38]	MC-ABE [39]	MHSN [41]	FLAC [48]
Year	2015	2016	2017	2017
Main Idea	Secure data transfer from data owners to the cloud servers, secure data storage and patient data privacy with authorized access control and fine-grained policies.	As mobile computing collects large amount of data from cloud integrated body senor network, it is a challenging issue to keep data privacy and security.	The fusion of health data and social data may pose series of privacy and security issues in smart cities.	WCC environment brings new challenges to data confidentiality, data integrity, and access control.
PPPs	ECG, EGG,	ECG, EGG,	Blood pressure, Heart Rate, Pulse	Medical data
Findings	A novel encryption outsourcing scheme MC-ABE for patient data privacy and security in C-BSN.	A novel MC-ABE-based mechanism for access control in C-BSN.	A secure health and social data sharing collaboration scheme to preserve patient data privacy.	A fine-grained and light-weight access control scheme for WCC in order to control access and data confidentiality and to support diverse access policies
Controller	Trusted Authority	Data owner	Trusted Authority	Trusted Authority
Patient Mode	Indoor /outdoor	Indoor	Indoor/outdoor	Indoor
Emergency Management	No	No	No	No
Limitation	Total computational cost is proportional to the number of privileges and storage space is proportional to the number of DRs. Need to improve scalability.	No validation for real access of PPPs available.	Storage cost is not evaluated.	Real PPPs and access scenarios are not evaluated for performance measurements.

Table 6: Summary of Chaotic Map Technique in m-Health.

Technique Ref. No.	CAA [49]
Year	2016
Main Idea	Patient medical data security in public and insecure communication channels for cloud-assisted WBAN in order to save patients' lives.
PPPs	ECG, EEG, Electromyography, Pulse, oximetry, Body pulse, Heartbeat, Blood pressure
Findings	A secure mobile emergency-based cloud-assisted WBAN system for real-time monitoring of patients. It protects patient privacy and also reduces the burden of system overhead.
Controller	Trusted Authority
Patient Mode	Indoor
Emergency Management	Yes
Limitation	Evaluation of average communication cost and access time ignored.

Table 7: Summary of Hybrid Technique Encryption in m-Healthcare.

Technique Ref. No.	IoT [50]
Year	2017
Main Idea	Embedded devices with cloud servers can provide flexible medical aid to elderly people. However, this environment has various security issues.
PPPs	ECG, oxygen saturation, blood pressure, body temperature
Findings	A secure IoT-based senor cloud scheme for continuous monitoring of elderly people.
Controller	Trusted Authority
Patient Mode	Indoor/outdoor
Emergency Management	Yes
Limitation	The scheme is not easy to use for elderly people.

server compares the received PPPs with standard values of parameters on the database. In the event of an emergency, EFC approached and notified within an acceptable time. If the collected PPPs from IoT-based medical sensors are normal, the cloud sends a report to the patient. This whole process and the medical data are shared between the various entities in the scheme in asymmetric/symmetric or hybrid encryption. Meanwhile, the scheme is claimed to reduce the wastage of medical resources. Table 7 is a summary of this technique in m-health.

4.3. E-Healthcare

Attribute-Based Encryption. Efficient and secure Patient-Centric Access Control (ESPAC) scheme [42] consists of four main entities: (1) trusted authority, (2) cloud service provider, (3) registered users, and (4) data-access requester in two phases (A and B). In phase A, secure data communication is arranged between different e-health users, while in phase B a traditional cryptographic system data-access request is controlled. The encrypted data is stored in a central health cloud for access. An analysis of the security and performance of ESPAC demonstrated the desired security requirements with only a reasonable delay in communication.

Privacy and security for PHI in IoT cloud-based systems always represent a challenge. Another technique, by Yeh et al. [43] in 2015, proposed e-health as a cloud-based framework for fine-grained access control to address the challenge. A variant of ciphertext, policy attribute-based encryption, is used with Merkle hash trees and dual encryption to handle fine-grained access control for lightweight devices such as wireless body sensors. Meanwhile, the fine-grained access control framework also provides efficient dynamic auditing, batch auditing, and attribute revocation. An analysis of the security and performance showed that the scheme is excellent as a cloud-based PHI system.

Similarly, the AYA model of K. Martin et al. [44] solves data accountability issues by introducing a new concept of trusted logical agent as a private cloud with data owner control. The data owner is responsible for the processing and storage of his/her data on an outsourced private cloud. The focus of the proposed solution is on an efficient authentication service on a public cloud with a 'one-time token' algorithm and secure access granted on the private cloud by using $C_{p\text{-}ABE}$. First, a data owner selects a service (Ω, PKI) from the public cloud. Secondly, the data owner gives an instruction to the trusted point (TP) to generate a private key. Then, the service requests access to the private cloud for successful authentication.

In 2017, Shynu et al. [51] proposed an e-health cloud storage system to handle multiple users for sensitive data sharing. The system consists of four major entities: CS, key management centre (KMC), DO, data user (DU), a non-patient-centric approach adopted in which the health service provider (HSP) plays the role of DO. The patients were monitored through WBANs continuously and health data were collected in electronic health records (EHR). First, users registered with CS and obtained their pair of cryptographic

keys and smartcard. In the next step, a mutual authentication process takes place. HSP is responsible for the secure connection between DO and DU. HSP issues an attribute certificate to the trusted entities. After this, the HSP enforces access policies (read, write) for data access and enforces data encryption. Here, the system utilized is the attribute-based searchable encryption (ABE) technique. During the whole process, a trapdoor function is calculated for every patient. Table 12 is a summary of the ABE technique in e-healthcare. Table 8 shows the summary of attribute-based techniques in e-healthcare domain.

Number Theory Research Unit (NTRU). Compared to earlier studies in 2016, Chen et al. [52] proposed a trustable scheme to maintain patient privacy while sharing PPPs from wearable devices to cloudlet technology. The content sharing and privacy protection are maintained by NTRU [71]. The encryption scheme uses NTRU to encrypt PPPs (ECG, heart rate, blood pressure, and so on) before transmitting to a smartphone or any other personal handheld device. Data collected from smart cloths are usually unsigned and stored integer vectors. Table 9 is a summary of NTRU technique in e-healthcare.

Tri-Mode Algorithm. Antony et al. [3] proposed an innovative application, the Integrated Secure Authentication (ISA), by negating all traditional cryptographic approaches. ISA is a cloud-based e-health system to solve the authentication problem by proposing Tri-Mode Algorithm. The e-health system received signal strength (RSS) value from the located device and stored authentication list using the SetUp algorithm. Then, a CheckUp algorithm verifies the authentication of the user for data access to the cloud. However, this technique is limited to authorizing a single medical entity. Table 10 presents the Tri-Mode technique in e-healthcare.

Hybrid Encryption. The security model proposed [53] uses layers of security at different levels. For example, integrity and confidentiality at WBAN data collection level; network security and confidentiality at transmission level; integrity, availability, and data confidentially at storage level; and authentication and authorization at data access level. A double-layer encryption technique is used for control of access in this model. The symmetric encryption technique is used due to its efficiency, as the same key is used to both encrypt and decrypt data. While an asymmetric encryption technique is preferred due to pairs of keys (public, private) and easiness of key distribution during transmission, this process is quite slow. Taking advantage of both the aforementioned encryption techniques, double-layer hybrid encryption is used for data confidentiality and integrity. Table 11 is a summary of the hybrid encryption technique.

Dynamic Probability Packet Marking. Latif et al. [54] claimed that existing Probabilistic Packet Marking (PPM) for sensor networks is limited to fixed making probability τ_i, which results in high convergence time, uncertainty, and additional overhead due to 'Key Issues in Selecting Probability.' Its main cause is the assignment of uneven probability φ_i to n_i (sensor node), along with the attack path, whereas Dynamic Probability Packet Marking (DPPM) uses the Time-to-Live (TTL) to determine the travelling time of each packet passing by the router. By using this concept of DPPM, an Efficient Trackback Technique (ETT) is proposed to handle Distributed Denial of Service Attack (DDOS) for S-CI. Table 12 is a summary of the PPPM technique in e-healthcare.

Priority-Based Data Forwarding. To aggregate different types of health data in S-CI and priorities data according to need and availability is a big challenge. Zhang et al. proposed a 'priority-based health data aggregation' scheme (PHDA) [46] to establish a secure and reliable connection between WBANs and CS, with some security requirements. The network model is responsible for secure and reliable connection, with the assumption that S-CI is a trusted entity. Four entities, TA, SP, cloud server (CS), and mobile users, are utilized in the network model. The security model of PHDA is intended to reduce the communication overhead with security goals such as data privacy, identity privacy, and resistance to forgery attack. The PHDA protocol is able to aggregate health sensing data efficiently, based on data-forwarding strategies. Health data are classified into (1) emergency calls, (2) PPPs, and (3) normal health data. These types of data are prioritized according to their significance and size. Mobile users' u_i prioritized data has a data priority detection module. PHDA proceeds with (1) an initialization phase, (2) health data generation, (3) priority-based forwarding, (4) data aggregation, and (5) data decryption. Table 13 is a summary of the PHDA technique in health data aggregation.

4.4. Health Data Management

Attribute-Based Encryption. Lounis et al. [2] proposed a cloud-based secure architecture for the data management of large-scale data collected from WBANs. In this architecture, the mechanism of CP-ABE is used for effective and flexible security in order to achieve confidentiality and integrity with fine-grained access control for outsourced EHRs. The system is composed of three main entities: patients, cloud servers, and healthcare professionals. For communication security, the SSL protocol is assumed to be the communication channel. Medical data are encrypted at user level, as cloud servers are considered an untrusted entity. Therefore, the health authority (HA) is introduced as the trusted authority for key assurance and access policies. Each party has a public/private key pair, which can be obtained easily through Public Key Infrastructure (PKI). Extensive simulation has shown that the proposed scheme allows for efficiency and scalability with fine-grained access control in both emergency and normal scenarios. Table 14 is a summary of the ABE technique in health data management.

4.5. Security Services in S-CI. In this section, we discuss the extracted security services provided by the various techniques, according to their application areas. We observe that areas such as m-healthcare and e-healthcare are the most addressed in terms of security services. In m-healthcare, the most addressed security services are data confidentiality

Table 8: Summary of the Attribute-based Technique in e-Healthcare.

Technique Ref. No. **Year**	ESPAC [42] 2011	e-Health framework [43] 2015	AYA [44] 2015	ABE scheme [51] 2017
Main Idea	Patient self-controlled privileges to PHI for cloud storage at anytime, anywhere, and remote access.	IoT devices for pervasive personal health information (PHI) system facing privacy and security challenges in cloud-based environment.	A new paradigm 'Cloud of Things' rises some new challenges at outsourced data in terms of access control, privacy protection, and data integrity, as patient ownership.	The existing techniques are patient centric and do not provide security with fine-grained access control.
PPPs	e-Healthcare service provider provides health data	PDA based PHI	Medical data	EHR
Findings	A secure and an efficient and patient-centric access control scheme for real-time monitoring and remote access from cloud storage.	E-healthcare-based lightweight framework to achieve fine-grained access control with efficient revocation and dynamic revocation.	A security architecture for patient owner controlled at outsourced data in public cloud environment.	An ABE technique with trapdoor function to avoid unauthorized access control for cloud-based health data.
Controller	Trusted Authority	Trusted Authority	Patient	Trusted Authority
Patient Mode	Indoor/outdoor	Indoor/outdoor	Indoor	Indoor
Emergency Management	No	No	No	No
Limitation	PPPs in dataset and storage cost is not analysed; patient participation and control are not considered.	Log auditing is missing	No support for keyword similarity mechanism	Need to improve efficiency of ABE-based access control for e-health clouds.

Table 9: Summary of Number Theory Research Unit Technique in e-Healthcare.

Technique Ref. No.	NTRU [52]
Year	2016
Main Idea	Medical data sharing from wearable devices to cloudlet is critical, as it involves sensitive patient data.
PPPs	EEG, Pulse, EMG, ECG
Findings	To protect patient privacy of medical data, a cloudlet-based data sharing system.
Controller	Trusted Authority
Patient Mode	Indoor/outdoor
Emergency Management	No
Limitation	Computation and network cost is not evaluated

Table 10: Summary of Tri-Mode Technique in e-Healthcare.

Technique Ref. No.	ISA [3]
Year	2016
Main Idea	In WBANs cryptographic authentication is desirable due to their computational complexity for spoofing attacks.
PPPs	RSS value
Findings	ISA application for e-healthcare using cloud computing to prevent spoofing attacks in sensor network.
Controller	Trusted authority
Patient Mode	Indoor
Emergency Management	No
Limitation	Technique is limited to handling just one healthcare authority.

Table 11: Summary of Hybrid Encryption Technique in e-Healthcare.

Technique Ref. No.	WSN [53]
Year	2016
Main Idea	In S-CI, medical information needs strong privacy and security protection mechanism against unauthorized access.
PPPs	Medical data
Findings	A security framework for S-CI by separating data control to third party to provide fast and reliable security requirements.
Controller	Trusted authority
Patient Mode	Indoor/outdoor
Emergency Management	No
Limitation	Storage and computation cost are not evaluated.

Table 12: Summary of Probability Packet Marking in e-Healthcare.

Technique Ref. No.	ETT [54]
Year	2016
Main Idea	One of the critical attacks in WBANs environment is DDOS attack, which increases resource utilization and also affects data privacy and security.
PPPs	Body temperature, Pulse oxygen, body temperature, blood pressure, EEG
Findings	An efficient trace back technique for cloud-assisted WBANs environment to avoid DDOS attacks.
Controller	Server
Patient Mode	Indoor
Emergency Management	No
Limitation	This scheme is limited to number of bytes for nodes upon network topology. This scheme uses WBANs and MAC header. Furthermore, this scheme can use IPv6 header for deployment and is evaluated by IPv6 header.

Table 13: Summary of the Priority-Based Data Forwarding Technique in Health Data Aggregation.

Technique Ref. No.	PHDA [46]
Year	2014
Main Idea	Different types of health data aggregation are challenges in S-CI with security and privacy paramount during communication between WBAN and Cloud.
PPPs	ECG, medical images
Findings	PHD scheme for S-CI to improve health data aggregation efficiently by reserving data identity and privacy.
Controller	Trusted Authority
Patient Mode	Indoor/outdoor
Emergency Management	Yes
Limitation	Computation and computation overheads

Table 14: Attribute-Based Encryption Technique in Health Data Management.

Technique Ref. No.	CP-ABE [2]
Year	2016
Main Idea	Lack of data management in WSN due to which medical data is facing challenges like scalability, availability, and security.
PPPs	Medical, health data
Findings	Reduce data management and processing overhead in sensor cloud-based scalable architecture to guarantee the integrity, confidentiality, and fine-grained access control of medical data in emergency situations without involving patients and doctors.
Controller	Trusted Authority
Patient Mode	Indoor
Emergency Management	Yes
Limitation	Patient participation and control for data access are not concerned and it is not clearly mentioned which PPPs are accessed

(n=5) and fine-grained access control *(n=3)*. Similarly, there is much on collusion resistance, message integrity, replay attack, and man-in-the middle with *(n=2)*. By contrast, patient privacy, source authentication, attributes revocation, availability, impersonation attack, know-key security, non-repudiation, and transmission continuity with *(n=1)* are the least addressed security services in this area. Moreover, patient access controls, denial of attack, ciphertext-only attacks, patient participation, dynamic data operation, cloud reciprocity problems, scalability, resistance to forgery attack, identity attack, authorization, and network security services are totally ignored in m-healthcare.

In the area of e-healthcare, the security services that are commonly addressed are patient control with *(n=2)*, source authentication, audit control, data confidentiality, message integrity, and DOS with *(n=1)*. Meanwhile, services such as signature unforgeability or anonymity, transmission continuity, man-in-the middle attack, known-key security, nonrepudiation, replay attack, impersonation attack, resistance to forgery attack, and scalability are still answered. The least-addressed application areas in terms of security services are health data management and health data aggregation.

We can clearly observe from Table 14 that data confidentiality, fine-grained access control, collusion resistance message integrity, and availability and scalability with (n=1) are a few services that are provided in the area of health data aggregation. Similarly, there is plenty of scope for research in health data aggregation: patient privacy, identity privacy, and resistance to forgery attack are covered by a single technique. Table 15 is an overview of the extracted security services and the proposed techniques.

4.6. Patient Physiological Parameters as Dataset. We observed that studies did not adopt any particular dataset from the existing literature to propose their techniques for patient data privacy and security in S-CI. Most studies utilized common PPPs [72] as a dataset, sensed through body sensors. Table 16 shows the common set of PPPs used as a dataset. The following are some important PPPs sensed through body sensors in S-CI for patient data privacy and security.

4.6.1. Electrocardiography. Electrocardiography, or ECG, is a medical process in which electrodes are placed on the human body. In this process, the electric heart activity is recorded over a period of time. In short, the overall purpose of ECG is to obtain information about the function and structure of the heart. In patients, ECG sensors are usually placed for the timely detection of heart attacks, chest pain, shortness of breath, cardiac stress, and so on [37–40, 46, 47, 49, 50, 54].

4.6.2. Electroencephalography. Electroencephalography, or EEG, is a medical method to monitor brain activity. In this process, electrodes are placed on the human scalp to measure

Table 15: Extracted Security Services from Proposed Techniques.

Security Services	m-Healthcare	e-Healthcare	Health Data Management	Health Data Aggregation
Data Confidentiality	[41, 47, 48, 50, 55]	[44, 53]	[2]	
Fine-grained Access Control	[41, 47, 48]	[43, 51]	[2]	
Collusion Resistance	[41, 48]	[42, 43]	[2]	
Patient-centric Access Control		[42]		
Message integrity	[47, 50]	[42, 53]	[2]	
Denial of Service (Dos) Attack		[42, 54]		
Prevention of Ciphertext-only attack		[42]		
Patient Privacy	[50]	[42]		[46]
Patient Control		[43, 44, 53]		
Source Authentication	[37]	[43, 44]		
Dynamic Data operation		[43]		
Audit Control		[43, 44]		
Attribute revocation	[39]	[43]		
Cloud Reciprocity Problem		[43]		
Availability	[47]	[53]	[2]	
Scalability			[2]	
Identity Privacy		[42]		[46]
Resistance to Forgery Attack				[46]
Impersonation Attack	[55]			
Man-in-the middle Attack	[50, 55]			
Nonrepudiation	[55]			
Signature Unforgeability and Anonymity	[48]			
Replay Attack	[50, 55]			
Transmission Continuity	[50]			
Authorization		[53]		
Known-key Security	[55]			
Network Security		[53]		

Table 16: Common Set of PPPs used as Dataset.

Symbol	Description	Frequency	References
ECG	Electrocardiogram	8	[37, 38, 40, 46, 47, 49, 50, 52]
EEG	Electroencephalography	8	[37–40, 47, 49, 50, 52]
HR	Heart Rate	5	[41, 47, 49, 50, 54]
BP	Blood Pressure Rate	5	[41, 47, 49, 50, 54]
PR	Pulse Rate	5	[41, 49, 50, 52, 54]
EMG	Electromyography	2	[52, 54]
Oximetry	Oximetry Technology	1	[49]
BT	Body Temperature	1	[50]
RP	Respiratory Rate	1	[46]
SpO_2	Oxygen Saturation	1	[46]

voltage fluctuations in the neuron of the brain produced in the form of an ionic current. EEG sensors are usually placed to record epileptic seizures and psychiatric syndromes in patients [37–40, 47, 49].

4.6.3. Blood Pressure. The flow of blood circulation in blood vessels for oxygen supply is directly affected by strain at heart arteries. The blood pressure of the body recoded to measure this strain. Smart sensors using microprocessors are used to sense the flow and send the data remotely to medical staff for real-time monitoring [41, 47, 49, 50, 54].

4.6.4. Body Temperature. Normal body temperature ranges from 36.5 to 37.5°C according to age, sex, infection, exertion, reproductive status, and so. Medical sensors are placed in patients with a serious disease to diagnose changes in body temperature in order to aid the medical facility in time [50, 54].

4.6.5. Heartbeat Rate. The rate of the human heartbeat is measured in the form of contractions, or beats, in bpm. The amount of contraction varies due to the physical need for oxygen in the body. Sensors such as the Polar H10 are placed to monitor the heartbeat rate during the various physical activities that are performed during the day [41, 47, 49, 50].

4.6.6. Electromyography. Electromyography, or EMG, is a medical process in which the skeletal muscles' movements are evaluated and recorded in terms of electrical activity. Usually, it is patients with neuromuscular disease who are implanted with EMG sensors to evaluate the muscle activity in real time [54].

4.6.7. Oximetry. Oximetry is a medical technology to measure the level of oxygen in the blood with the heart rate. Patients, usually with asthma or respiratory issues, are implanted with oximetry-based sensors in order to aid medical service in case of emergency [49].

4.6.8. Oxygen Saturation. In medicine, the term oxygen saturation refers to the amount of oxygen-saturated hemoglobin compared to total hemoglobin. For example, patients suffering from severe anemia usually suffer from reduced arterial oxygen saturation with $SaO_2 < 90\%$.

5. Performance Estimation

A number of ways are adopted to evaluate the techniques. Most studies have adopted simulations of the techniques to evaluate the performance of the patient data privacy and security in S-CI. Simulation encryption with operation analysis [2], simulation with NS-2 simulator [3, 52, 54], and implantation with Jpair library and Netbeans for algorithm [44, 53] are other common methods.

The emergency case scenario is evaluated as the highest communication cost, with $45{,}760/20 * 10^{-6} = 0.9152$ ms at $20Mbps$ bandwidth [50]. For the computation cost, AES symmetric encryption, SHA-256 hash function, Menezes-Vanstone cryptosystem, and signature of ECDSA are used in [50]. Similarly, the privacy authentication scheme [55] is compared to existing techniques in terms of computation and communication cost. This scheme also uses AES symmetric encryption, SHA-256 hash function, Menezes-Vanstone cryptosystem, and signature of ECDSA by ECDSA. The average time for health data request is assumed (m_{HC}, m_{BS}, or m_D) with X.509 certificate with 8192 bits and with 245760 bits. The treatment phase, with 1,230,316 bits as 3G telecommunication cost is 2Mbps/384 kbps/144 kbps, is evaluated and as the worst in terms of communication cost while, in the simulation of experiments in the ABE scheme [51], the computation time for encryption process is 28ms (milliseconds), quite low compared to other existing techniques (39ms). However, a complexity comparison with traditional encryption schemes [46] and the timing cost of operations used in ESPAC [42], evaluated by varying number of attributes [42], are commonly used to calculate time and cost complexity. Furthermore, high communication cost is evaluated on the basis of an emergency scenario, transmitting time of message in Mbps bandwidth network, in [50].

One study also used the SPSS tool to calculate function complexity, whereby an experimental setup created in Ubuntu 14.04 LTS 64-bit system [51] and ABE is compared to other existing encryption techniques in terms of computational time. Another study evaluated data sharing time with cloudlet, based on a trust model, analysed and categorized at three different levels as 'bad, average, and good.' These levels were assigned whereby individual repute (r) was set to [0, 1], ranges with ([0, 0.2], [0.2, 0.6], [0.6, 1]) were 'bad,' 'average,'

Table 17: Summary of Performance Metrics.

Techniques	Performance Metrics
Multibiometric Key Generation [37]	(1) Entropy
Pairwise Key Establishment [40, 43]	(1) Encryption and decryption measured through Pairing-based Key generation = Pair (ADK), Exp_G (Share Key) (2) Probability compromise for data sink = Prob $_{DS}$ (α), Probability compromise for body sensors = Prob $_{Bsr}$(γ)
Hash Function [47]	(1) Data transfer rate at quality factor= 0.1
Attribute-based Encryption [2, 11, 38, 39, 41–43, 43, 44, 48, 51]	(1) Computational complexity = O(|I|), Time Complexity = O(|S|) (2) Computation cost = O(n), O(m) (3) Sensor Data encryption = $5T_0 = T_t$,, Data decryption = T_r, Key generation = $3T_0+T_t$ (4) Key generation complexity = O(n), Encryption complexity = O(1) mod,
Chaotic Maps [49]	(1) T_{Hash} = Time of execution per hash function, T_{Sig}= Time of execution for signature, T_{sym}= Time of symmetric encryption/decryption
Hybrid Encryption [50]	(1) Key generation = $3T_0+T_t$, Data decryption = T_r,
2014 Multivalued encryption [20]	(1) Coverage time= CTFBT ≥ 1 τ 1–τðÞ N–1

and 'good,' respectively [52]. In AYA [44], the efficiency analysis of $C_{p\text{-}ABE}$ algorithms is by comparison, using the Jpair library in terms of set time (ms), encryption time, and decryption time.

The quantitative measurement entropy of multibiometric-based scheme compared with ECG- and EEG-based schemes shows a high entropy, meaning high security [37]. Similarly, the efficiency of the authentication system [3] is evaluated as false positive rate (FPR), false negative rate (FNR), sensitivity, specificity, and accuracy. The FPR is the ratio of negative cases reported as positive, and vice versa for FNR. However, in this analysis, 20% of nodes are assumed to be attackers:

$$\text{“}FPR = \frac{False\ Positive}{False\ Positive} + True\ Negative\text{”} \quad (12)$$

$$\text{“}FNR = \frac{False\ Negative}{False\ Negative} + True\ Positive\text{”}. \quad (13)$$

The sensitivity of the system (true positive rate) measures the negatives correctly identified:

$$\text{“}Sensitivity = \frac{True\ Positive}{True\ Positive} + False\ Negative\text{”}. \quad (14)$$

The accuracy of the system shows the number of accurate results in both positive and negative cases:

$$\text{“}Accuracy = \frac{True\ Result}{True\ Result} + False\ Result\text{”}. \quad (15)$$

In WSN [53], a hybrid encryption technique is implemented in Java and run-time patient keys are generated for transmission. The total time required for encryption and decryption varies with file size and key size. For example, key file size with 168 bytes involves encryption in 88ms and decryption in 105ms. Similarly, a data file of 1.5MB is encrypted in 421ms and decrypted in 468ms. Meanwhile, CP-ABE [4] is compared with ABE in terms of encryption and decryption time. The comparison shows that the proposed solution, with 256 bits, is much faster than ABE. However, CP-ABE performance evolution does not show any significant gain in terms of access control. Furthermore, the ETT [54] technique is evaluated for coverage time, uncertainty, and node overhead. Coverage time is calculated with respect to packet numbers for a successful reconstruction path. The most prominent coverage time is given in $CT_{FBT}\tau(1\text{-}\tau)^{N-1} >= 1$. Similarly, the maximum uncertainty is evaluated as (m= (1/τ)-1). For nodes overhead, every node selects a marking probability of 1/d (for d = 1, 2 . . . , N) for each packet, so the overhead at each is OH_{ETT}=n/d. Summary of the performance estimation metrics of the selected techniques is given in Table 17.

6. Future Directions and Challenges

The distributed nature of S-CI highlighted a unique set of challenges for the research community in this flourishing area. This snapshot of the domain shows the following future challenges for research opportunities.

6.1. Lack of Standard Architecture. There is no standard architecture available for S-CI to ensure patient data privacy and security. Most studies follow a hierarchical architecture and introduce a separate TA entity [50] for key generation and security parameters, yet other studies take the hospital as the TA entity [2] to force access policies. Therefore, there is a great need to propose a standard architecture to access PPPs in S-CI whilst maintaining patient privacy and security.

6.2. Lack of Policy Compliance. It has been observed that no single study follows any comprehensive policy to address patient data privacy and security rights. For example, some studies address data confidentiality [47] and fine-grained access control [41] with collision resistance [41, 48], while others totally ignore these security services and focus on DOS [54] and man-in-the middle attack [28] with anonymity [28]. Furthermore, there have been few studies focusing on

patient participation and control [38, 53]. Similarly, just two studies have focused on auditing [43, 44]. Therefore, there is a great need for privacy and security policy compliance, such as HIPPA [73].

6.3. Lack of Standard Dataset. It has been observed that many studies have not used a properly defined dataset for their proposed solutions. Most of the studies have randomly utilized some common PPPs such as ECG, EGG, blood pressure, and pulse rate [10, 43, 47] as their dataset. Others just refer to it as medical data [51] or medical images [46], without specifying particular PPPs. Therefore, there is a great need to adopt some standard or 'golden' dataset.

6.4. Lack of Handling of Patient Behaviour and Intentions. It has been observed that how patient behaviour and intentions are handled to stimulate collaboration in social network is totally ignored [40]. Proper strategies and a trust model should be proposed to resolve this issue.

6.5. Lack of Emergency Management. Another important area, the emergency management, an important aspect of S-CI for PPPs real-time monitoring and access, is ignored while handling patient data privacy and security. Only a few studies handle emergencies [2, 46, 49, 50] in their solutions. There is a great need to handle emergency management using real scenarios of access.

6.6. Lack of Data Management in Multiple Accesses. We also observed that studies follow typical and traditional encryption techniques for S-CI for patient data monitoring and access. There is a great need to design new scenarios of data management for the distributed environment and multiple access of PPPs among various medical entities [2].

6.7. Lack of Search Encrypted Medical Terms and Similarity Semantics. It has been observed that no single study has reported any mechanism to search for encrypted key words of medical terms [42] and to support key word similarity semantics [44] in S-CI. Therefore, there is a great need for a search mechanism for encrypted and similarity semantics of key words in S-CI.

6.8. Non-User-Friendly Applications. The techniques and processes using S-CI applications should be user-friendly, from the patient perspective, especially for elderly or paralyzed patients to make it easy to follow the process [50].

6.9. Lack of Data Public Sharing and Data Publication. As S-CI processes a huge volume of PPPs, the sensitivity and significance of this data cannot be denied for health, technology, and government sectors. Research communities should focus on a secure design for public sharing and data publication [3].

6.10. Scalable and Efficient Data Access Control. It is observed that there is a need for improvement in the efficiency and scalability of techniques such as MC-ABE- and ABE- [51] based data access control in S-CI.

6.11. Innovative Designs for Network and Transmission Security. It is observed that network and transmission security is also ignored for S-CI. There must be improved techniques and methods for patient data privacy and security to give efficient, reliable, and continuous transmission [54].

6.12. Unreliability and Quality of PPPs. It is observed that no single study has proposed any technique to check the reliability and the quality of PPPs. As PPPs are very sensitive medical data [7], for the timely diagnosis and response to provide medical aid, there must be techniques to trace PPPs' quality and reliability with PPPs' privacy and security in S-CI.

6.13. Real-Time Implementation and Integration. It is observed that techniques are simulated in artificial environment for experiment in this research area. Therefore, techniques [70] should be implemented and integrated as real time in UEC–Eucalyptus platform to help with future enhancement.

7. Conclusion

This study has provided a detailed literature review of patient data privacy and security in S-CI. So far, we can clearly see that many techniques are proposed for mobile healthcare and e-healthcare and that there is much scope in the areas of research into health data management and health data aggregation. Meanwhile, this research area lacks in standard architecture, policy compliance, standard dataset, handling patient behaviour, search of encrypted medical terms, data sharing, data publication, and emergency and multiple access data management. Similarly, S-CI needs special attention in terms of user-friendly applications, non-user-friendly applications, efficient access control, network security, real-time implementation, and improved quality of patient data access. We also propose a generic framework, extracted from the available literature. Our framework is quite innovative and applicable to the research community. We have discussed performance estimation measures and various security services of the techniques proposed for patient data privacy and security in S-CI. Finally, we believe that our roadmap of this flourishing and innovative research area will be beneficial to highlight future enhancement.

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This article has been awarded by the National Natural Science Foundation of China (61170035, 61272420, 81674099), the Fundamental Research Fund for the Central Universities (30916011328, 30918015103), and Nanjing Science and Technology Development Plan Project (201805036).

References

[1] H. Alemdar and C. Ersoy, "Wireless sensor networks for healthcare: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2688–2710, 2010.

[2] A. Lounis, A. Hadjidj, A. Bouabdallah, and Y. Challal, "Healing on the cloud: secure cloud architecture for medical wireless sensor networks," *Future Generation Computer Systems*, vol. 55, pp. 266–277, 2016.

[3] A. A. V. Rani and E. Baburaj, "An efficient secure authentication on cloud based e-health care system in WBAN," *Biomedical Research (India)*, vol. 2016, pp. S53–S59, 2016.

[4] A. Lounis, A. Hadjidj, A. Bouabdallah, and Y. Challal, "Secure and scalable cloud-based architecture for e-Health wireless sensor networks," in *Proceedings of the 2012 21st International Conference on Computer Communications and Networks, ICCCN 2012*, Germany, August 2012.

[5] R. Negra, I. Jemili, and A. Belghith, "Wireless body area networks: applications and technologies," *Procedia Computer Science*, vol. 83, pp. 1274–1281, 2016.

[6] A. Grady, S. Yoong, R. Sutherland, H. Lee, N. Nathan, and L. Wolfenden, "Improving the public health impact of eHealth and mHealth interventions," *Australian and New Zealand Journal of Public Health*, vol. 42, no. 2, 2018.

[7] S. M. S. K. Dash and P. K. Pattnaik, "A survey on applications of wireless sensor network using cloud computing," *International Journal of Computer Science & Emerging Technologies*, vol. 1, pp. 50–55, 2010.

[8] A. Alamri, W. S. Ansari, M. M. Hassan, M. S. Hossain, A. Alelaiwi, and M. A. Hossain, "A survey on sensor-cloud: architecture, applications, and approaches," *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 917923, 18 pages, 2013.

[9] A. Tewari and P. Verma, "Security and privacy in e-healthcare monitoring with WBAN: a critical review," *International Journal of Computer Applications*, vol. 136, no. 11, pp. 37–42, 2016.

[10] A. Sajid and H. Abbas, "Data privacy in cloud-assisted healthcare systems: state of the art and future challenges," *Journal of Medical Systems*, vol. 40, no. 155, pp. 1–16, 2016.

[11] A. Celesti, F. Celesti, M. Fazio, P. Bramanti, and M. Villari, "Are next-generation sequencing tools ready for the cloud?" *Trends in Biotechnology*, vol. 35, no. 6, pp. 486–489, 2017.

[12] S. Al-Janabi, I. Al-Shourbaji, M. Shojafar, and S. Shamshirband, "Survey of main challenges (security and privacy) in wireless body area networks for healthcare applications," *Egyptian Informatics Journal*, vol. 180, no. 2, pp. 113–122, 2016.

[13] H.-P. Chiang, C.-F. Lai, and Y.-M. Huang, "A green cloud-assisted health monitoring service on wireless body area networks," *Information Sciences*, vol. 284, pp. 118–129, 2014.

[14] I. Masood, *Patient Data Privacy and Security in Sensor-Cloud Infrastructure: Past Present Future*, vol. 35, ACM Computing Surveys, 4 edition, 2018.

[15] A. D. I. Bisio, F. Lavagetto, and A. Sciarrone, "EHealth in the future of medications management: personalisation, monitoring and adherence," *IEEE Internet of Things Journal*, vol. 4, pp. 135–146, 2017.

[16] F. Firouzi, A. M. Rahmani, K. Mankodiya et al., "Internet-of-Things and big data for smarter healthcare: From device to architecture, applications and analytics," *Future Generation Computer Systems*, vol. 78, pp. 583–586, 2018.

[17] T. Péteri, N. Varga, and L. Bokor, "A Survey on Multimedia Quality of Experience Assessment Approaches in Mobile Healthcare Scenarios," in *eHealth 360◦*, vol. 181 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pp. 484–491, Springer International Publishing, Cham, 2017.

[18] E. Klaoudatou, E. Konstantinou, G. Kambourakis, and S. Gritzalis, "A survey on cluster-based group key agreement protocols for WSNs," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 3, pp. 429–442, 2011.

[19] R. Hummen, M. Henze, D. Catrein, and K. Wehrle, "A Cloud design for user-controlled storage and processing of sensor data," in *Proceedings of the 2012 4th IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2012*, pp. 233–240, Taiwan, December 2012.

[20] N. D. Han, L. Han, D. M. Tuan, H. P. In, and M. Jo, "A scheme for data confidentiality in cloud-assisted wireless body area networks," *Information Sciences*, vol. 284, no. 5, pp. 157–166, 2014.

[21] M. Yuriyama and T. Kushida, "Sensor-cloud infrastructure—physical sensor management with virtualized sensors on cloud computing," in *Proceedings of the 13th International Conference on Network-Based Information Systems (NBiS '10)*, pp. 1–8, September 2010.

[22] M. Chen, "NDNC-BAN: supporting rich media healthcare services via named data networking in cloud-assisted wireless body area networks," *Information Sciences*, vol. 284, no. 10, pp. 142–156, 2014.

[23] S. Sharma, K. Chen, and A. Sheth, "Towards practical privacy-preserving analytics for IoT and cloud based healthcare systems," *IEEE Internet Computing*, vol. 22, pp. 42–51, 2018.

[24] C. Ernsting, S. U. Dombrowski, M. Oedekoven et al., "Using smartphones and health apps to change and manage health behaviors: A population-based survey," *Journal of Medical Internet Research*, vol. 19, no. 4, article no. e101, 2017.

[25] T. Amjad, M. Sher, and A. Daud, "A survey of dynamic replication strategies for improving data availability in data grids," *Future Generation Computer Systems*, vol. 28, no. 2, pp. 337–349, 2012.

[26] I. Masood, Y. Wang, A. Daud, N. R. Aljohani, and H. Dawood, "Privacy management of patient physiological parameters," *Telematics and Informatics*, vol. 35, no. 4, pp. 677–701, 2018.

[27] G. Kambourakis, E. Klaoudatou, and S. Gritzalis, "Securing medical sensor environments: The CodeBlue framework case," in *Proceedings of the 2nd International Conference on Availability, Reliability and Security, ARES 2007*, pp. 637–643, Australia, April 2007.

[28] C.-T. Li, T.-Y. Wu, C.-L. Chen, C.-C. Lee, and C.-M. Chen, "An efficient user authentication and user anonymity scheme with provably security for IoT-based medical care system," *Sensors*, vol. 17, no. 7, 2017.

[29] R. S. Ponmagal, N. Dinesh, and U. Rajaram, "Design and development of secure cloud architecture for sensor services," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 8956, pp. 339–344, 2015.

[30] M. Henze, L. Hermerschmidt, D. Kerpen, R. Häußling, B. Rumpe, and K. Wehrle, "A comprehensive approach to privacy in the cloud-based Internet of Things," *Future Generation Computer Systems*, vol. 56, pp. 701–718, 2016.

[31] Q. Jiang, M. K. Khan, X. Lu, J. Ma, and D. He, "A privacy preserving three-factor authentication protocol for e-Health clouds," *The Journal of Supercomputing*, vol. 72, no. 10, pp. 3826–3849, 2016.

[32] P. Kumar and H.-J. Lee, "Security issues in healthcare applications using wireless medical sensor networks: a survey," *Sensors*, vol. 12, no. 1, pp. 55–91, 2012.

[33] E. Bertino, "Data security and privacy," in *Proceedings of the in IEEE 40th Annual Computer Software and Applications Conference*, 2016.

[34] A. Abbas and S. U. Khan, "E-health cloud: Privacy concerns and mitigation strategies," *Medical Data Privacy Handbook*, pp. 389–421, 2015.

[35] W. Khan, A. Daud, J. A. Nasir, and T. Amjad, "A survey on the state-of-the-art machine learning models in the context of NLP," *Kuwait Journal of Science*, vol. 43, no. 4, pp. 95–113, 2016.

[36] M. Al Ameen, J. Liu, and K. Kwak, "Security and privacy issues in wireless sensor networks for healthcare applications," *Journal of Medical Systems*, vol. 36, no. 1, pp. 93–101, 2012.

[37] F. A. Khan, A. Ali, H. Abbas, and N. A. H. Haldar, "A cloud-based healthcare framework for security and patients' data privacy using wireless body area networks," in *Proceedings of the 9th International Conference on Future Networks and Communications, FNC 2014 and the 11th International Conference on Mobile Systems and Pervasive Computing, MobiSPC 2014*, pp. 511–517, Canada, August 2014.

[38] Z. Guan, T. Yang, and X. Du, "Achieving secure and efficient data access control for cloud-integrated body sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2015, 2015.

[39] Z. Guan, T. Yang, X. Du, and M. Guizani, "Secure data access for wireless body sensor networks," in *Proceedings of the 2016 IEEE Wireless Communications and Networking Conference, WCNC 2016*, Qatar, April 2016.

[40] J. Zhou, Z. Cao, X. Dong, N. Xiong, and A. V. Vasilakos, "4S: a secure and privacy-preserving key management scheme for cloud-assisted wireless body area network in m-healthcare social networks," *Information Sciences*, vol. 314, pp. 255–276, 2015.

[41] Q. Huang, L. Wang, and Y. Yang, "Secure and privacy-preserving data sharing and collaboration in mobile healthcare social networks of smart cities," *Security and Communication Networks*, vol. 2017, pp. 1–12, 2017.

[42] M. Barua, X. Liang, R. Lu, and X. Shen, "ESPAC: enabling security and patient-centric access control for ehealth in cloud computing," *International Journal of Security & Networks*, vol. 6, no. 2-3, pp. 67–76, 2011.

[43] L.-Y. Yeh, P.-Y. Chiang, Y.-L. Tsai, and J.-L. Huang, "Cloud-based fine-grained health information access control framework for lightweight IoT devices with dynamic auditing and attribute revocation," *IEEE Transactions on Cloud Computing*, no. 99, 2015.

[44] K. Martin and W. Wang, "Aya: An efficient access-controlled storage and processing for cloud-based sensed data," in *Proceedings of the 12th International Computer Conference on Wavelet Active Media Technology and Information Processing, ICCWAMTIP 2015*, pp. 130–134, China, December 2015.

[45] J. Zhou, Z. Cao, X. Dong, and X. Lin, "PPDM: A privacy-preserving protocol for cloud-assisted e-Healthcare systems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 7, pp. 1332–1344, 2015.

[46] K. Zhang, X. Liang, M. Baura, R. Lu, and X. Shen, "PHDA: a priority based health data aggregation with privacy preservation for cloud assisted WBANs," *Information Sciences*, vol. 284, pp. 130–141, 2014.

[47] S. Saha, "Secure sensor data management model in a sensor-cloud integration environment," in *Proceedings of the 2015 2nd International Conference on Applications and Innovations in Mobile Computing, AIMoC 2015*, pp. 158–163, India, February 2015.

[48] H. He, J. Zhang, J. Gu, Y. Hu, and F. Xu, "A fine-grained and lightweight data access control scheme for WSN-integrated cloud computing," *Cluster Computing*, vol. 20, no. 2, pp. 1457–1472, 2017.

[49] C.-T. Li, C.-C. Lee, and C.-Y. Weng, "A secure cloud-assisted wireless body area network in mobile emergency medical care system," *Journal of Medical Systems*, vol. 40, no. 5, Article ID 117, 2016.

[50] J.-X. Hu, C.-L. Chen, C.-L. Fan, and K.-H. Wang, "An intelligent and secure health monitoring scheme using IoT sensor based on cloud computing," *Journal of Sensors*, vol. 2017, Article ID 3734764, 11 pages, 2017.

[51] P. G. Shynu and K. J. Singh, "An enhanced ABE based secure access control scheme for E-health clouds," *International Journal of Intelligent Engineering and Systems*, vol. 10, no. 5, pp. 29–37, 2017.

[52] M. Chen, Y. Qian, J. Chen, K. Hwang, S. Mao, and L. Hu, "Privacy protection and intrusion avoidance for cloudlet-based medical data sharing," *IEEE Transactions on Cloud Computing*, vol. PP, no. 9, pp. 1–9, 2016.

[53] S. Saha, R. Das, S. Datta, and S. Neogy, "A cloud security framework for a data centric WSN application," in *Proceedings of the the 17th International Conference*, pp. 1–6, Singapore, Singapore, January 2016.

[54] R. Latif, H. Abbas, S. Latif, and A. Masood, "Distributed denial of service attack source detection using efficient traceback technique (ETT) in cloud-assisted healthcare environment," *Journal of Medical Systems*, vol. 40, no. 161, pp. 1–13, 2016.

[55] C.-L. Chen, T.-T. Yang, M.-L. Chiang, and T.-F. Shih, "A privacy authentication scheme based on cloud for medical environment," *Journal of Medical Systems*, vol. 38, article no. 143, 2014.

[56] F. Nadeem and R. Qaiser, "An early evaluation and comparison of three private cloud computing software platforms," *Journal of Computer Science and Technology*, vol. 30, no. 3, pp. 639–654, 2015.

[57] J. Araujo, R. Matos, V. Alves et al., "Software aging in the eucalyptus cloud computing infrastructure: Characterization and rejuvenation," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 10, no. 1, 2014.

[58] SNIA, "CloudDataManagementInterface(CDMI)Version1.0.1/," http://cdmi.sniacloud.com/S, 2011.

[59] G. Galante, L. C. Erpen De Bona, A. R. Mury, B. Schulze, and R. da Rosa Righi, "An analysis of public clouds elasticity in the execution of scientific applications: a survey," *Journal of Grid Computing*, vol. 14, no. 2, pp. 193–216, 2016.

[60] T. Chippendale, P. A. Gentile, M. K. James, and G. Melnic, "Indoor and outdoor falls among older adult trauma patients: A comparison of patient characteristics, associated factors and outcomes," *Geriatrics & Gerontology International*, vol. 17, no. 6, pp. 905–912, 2017.

[61] G. N. Boysen, M. Nyström, L. Christensson, J. Herlitz, and B. W. Sundström, "Trust in the early chain of healthcare: Lifeworld hermeneutics from the patient's perspective," *International Journal of Qualitative Studies on Health and Well-being*, vol. 12, no. 1, 2017.

[62] M. B. M. Nateghizad and M. A. Maarof, "Secure searchable based asymmetric encryption in cloud computing," *International Journal of Advances in Soft Computing and Its Applications*, vol. 6, no. 1, pp. 1–13, 2014.

[63] N. Saini, N. Pandey, and A. P. Singh, "Enhancement of security using cryptographic techniques," in *Proceedings of the 4th International Conference on Reliability, Infocom Technologies and Optimization, ICRITO 2015*, India, September 2015.

[64] J. Patel, "Secure hashing algorithm and advance encryption algorithm in cloud computing," *International Journal of Computer and Information Engineering*, vol. 11, pp. 754–758, 2017.

[65] N. Koblitz and A. Menezes, "Pairing-based cryptography at high security levels," in *Cryptography and Coding*, vol. 3796 of *Lecture Notes in Computer Science*, pp. 13–36, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[66] N. El Mrabet, J. J. Fournier, L. Goubin, and R. Lashermes, "A survey of fault attacks in pairing based cryptography," *Cryptography and Communications*, vol. 7, no. 1, pp. 185–205, 2015.

[67] Z. Lv, J. Chi, M. Zhang, and D. Feng, "Efficiently attribute-based access control for mobile cloud storage system," in *Proceedings of the 13th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2014*, pp. 292–299, China, September 2014.

[68] S. Saleh, A. Farah, H. Dimassi et al., "Using mobile health to enhance outcomes of noncommunicable diseases care in rural settings and refugee camps: randomized controlled trial," *JMIR mHealth and uHealth*, vol. 6, no. 7, p. e137, 2018.

[69] L. Liu, H. Zhang, X. Yu, Y. Xin, M. Shafiq, and M. Ge, "An efficient security system for mobile data monitoring," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 9809345, 10 pages, 2018.

[70] A. Waqar, A. Raza, H. Abbas, and M. K. Khan, "A framework for preservation of cloud users' data privacy using dynamic reconstruction of metadata," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 235–248, 2013.

[71] D. Nuñez, I. Agudo, and J. Lopez, "NTRUReEncrypt: An efficient proxy re-encryption scheme based on NTRU," in *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security, ASIACCS 2015*, pp. 179–189, Singapore, April 2015.

[72] T. Schradi and G. Tivig, "Representation of a review of a patent's physiological parameters," *Google Patents*, 1999.

[73] W. B. Lee and C. D. Lee, "A cryptographic key management solution for HIPAA privacy/security regulations," *IEEE Transactions on Information Technology in Biomedicine*, vol. 12, no. 1, pp. 34–41, 2008.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 2783658, 9 pages
https://doi.org/10.1155/2018/2783658

Research Article

BaDS: Blockchain-Based Architecture for Data Sharing with ABS and CP-ABE in IoT

Yunru Zhang,[1,2] Debiao He ,[1,2] and Kim-Kwang Raymond Choo[3]

[1]*Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China*
[2]*Jiangsu Key Laboratory of Big Data Security & Intelligent Processing, Nanjing University of Posts and Telecommunications, Nanjing 210023, China*
[3]*Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249, USA*

Correspondence should be addressed to Debiao He; hedebiao@163.com

Received 20 August 2018; Revised 28 September 2018; Accepted 18 October 2018; Published 4 November 2018

Guest Editor: Georgios Kambourakis

Internet of Things (IoT) and cloud computing are increasingly integrated, in the sense that data collected from IoT devices (generally with limited computational and storage resources) are being sent to the cloud for processing, etc., in order to inform decision making and facilitate other operational and business activities. However, the cloud may not be a fully trusted entity, like leaking user data or compromising user privacy. Thus, we propose a privacy-preserving and user-controlled data sharing architecture with fine-grained access control, based on the blockchain model and attribute-based cryptosystem. Also, the consensus algorithm in our system is the Byzantine fault tolerance mechanism, rather than Proof of Work.

1. Introduction

The Internet of Things (IoT) has many applications in a wide range of industries and settings, such as smart homes and intelligent transportation systems, as well as in consumer applications (e.g., medical and health-care equipment) [1, 2]. One typical role of IoT devices (e.g., sensors and smart devices) is to collect and transmit (the collected) data via the Internet, like further processing and statistical analysis. However, IoT devices are generally resource-constrained, for example, having limited computational and storage resources. Thus, there has been a trend for integrating IoT and the cloud, to which data storage, processing, and sharing functionalities are being outsourced [3, 4].

As shown in Figure 1, *Owner1* can store and share the collected data with *Owner2* via the cloud to minimize costs. However, there is a risk that data and user's privacy may be leaked and compromised since the cloud is not fully trusted (i.e., semitrusted). Although there exists many privacy-preserving data processing solutions (e.g., utilizing cryptographic tools based on access control policies [5, 6]) for cloud storage systems, these approaches are vulnerable to attacks at the cloud end (e.g., access control policies may be tampered or deleted by a malicious cloud service provider or its employee).

Blockchain is a distributed ledger technology that underpins Bitcoin [7] and has been used in many other decentralized applications, such as digital currency [8, 9], data storage [10, 11], data provenance [12], Internet of Things [13–16], and so on. In this paper, we posit the potential of integrating blockchain with attribute-based cryptosystems [17, 18] in the design of a privacy-preserving and user-controlled solution to IoT data sharing. In other words, users can independently decide who can share their data without compromising data and identity privacy. Specifically, in our proposed BaDS (Blockchain-Based Architecture for Data Sharing with ABS and CP-ABE in IoT) architecture:

(i) IoT data are first encrypted (e.g., AES). Then, we integrate smart contract technology with an attribute encryption scheme [19] to realize its fine-grained sharing. The access policies are set on the encrypted key (the encrypted key are encrypted by attributes,

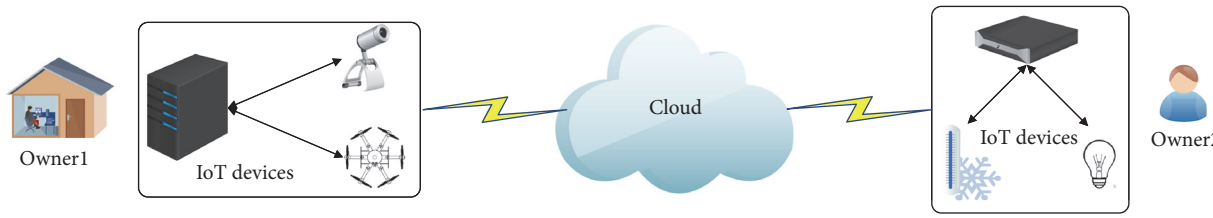


Figure 1: IoT data storage and sharing in a cloud-based model.

ABE) to decide who can obtain this encrypted key to decrypt the ciphertext.

(ii) The smart contract used in our architecture is to ensure the scalability of access control table. All the data sharing (or access) requests in the system interact with smart contracts through transactions (e.g., smart contract in Section 3.5).

The rest of the paper is organized as follows: In Section 2, we present the relevant cryptographic techniques, monotone span program, the network model, and security requirements. In Section 3, we briefly introduce ABS, CP-ABE, PACT, PBFT, and smart contract. After that, we describe our BaDS architecture and its security analysis in Section 4. In Section 5, we describe our evaluation of the proposed architecture, prior to concluding the paper in the last section.

2. Preliminaries

In this section, we will introduce the cryptographic techniques, monotone span program, the network model of our propose BaDS architecture, and the security requirements that need to be satisfied.

2.1. Bilinear Pairings. We define G_1 and G_2 as two additive cyclic groups on elliptic curve $F(p)$, G_T as a multiplication cyclic group. Let q be a big prime number, which is the order of G_1, G_2, and G_T. $e : G_1 \times G_2 \longrightarrow G_T$ denote a bilinear map. Suppose that the generators of G_1 and G_2 are P and Q; g is the element that P and Q map to G_T. Thus, the map e is a bilinear pairing on condition that e satisfies the following properties:

(i) **Bilinearity**. Given any two elements $a, b \in Z_q^*$, and $\forall X \in G_1, \forall Y \in G_2$, there is $e(a \cdot X, b \cdot Y) = e(X, Y)^{a \cdot b}$.

(ii) **Nondegenerate**. There exists at least one element X which satisfies $e(X, X) \neq 1$.

(iii) **Efficient Computability**. Given any two elements $\forall X \in G_1, \forall Y \in G_2$, there exists at least one efficient algorithm to compute $e(X, Y)$.

We define the computationally hard mathematical problems.

(i) **Discrete Logarithm (DL) Problem**. Given an element $X \in G_1$ or $x \in G_T$, for any P.P.T (probability polynomial time) attacker, it is computationally hard to calculate $\tau \in Z_q^*$ which satisfies $X = \tau \cdot P$ or $x = g^\tau$.

(ii) **Computational Diffie-Hellman (CDH) Problem**. Given a tuple (g, e, P, Q, aP, bQ) in which $(a, b) \in Z_q^*$, P, Q and g are the generators of G_1, G_2, and G_T, respectively. The purpose of **CDH** problem is to compute $\omega = g^{ab} \in G_T$, in which $(a, b) \in Z_q^*$ are unknown.

2.2. Monotone Span Program. Let $\Gamma : \{0,1\}^n \longrightarrow \{0,1\}$ as a monotone Boolean function. For an $l \times t$ matrix M over a filed F and every $(x_1, \ldots, x_n) \in \{0,1\}^n$, a monotone span program is defined as follows:

$$\Gamma(x_1, x_2, \ldots, x_n) = 1 \Longleftrightarrow \exists \overrightarrow{v} \in F^{l \times t} : M = [1, 0, 0, \ldots, 0] \tag{1}$$

in which $\forall i : x_{a(i)} = 0 \Longrightarrow v_i = 0$ and the labeling function $a : [l] \longrightarrow [n]$. That means $\Gamma(x_1, x_2, \ldots, x_n) = 1$ if and only if the index to the rows of matrix spans the vector $[1, 0, 0, \ldots, 0]$. We say the length and width of span program are l and t, and the size of it is $l + t$.

2.3. Network Model. Our BaDS architecture consists of the following participants: **IoT devices**, **Data Owner**, **Blockchain Network,** and **Cloud;** see Figure 2.

(i) **IoT Devices.** IoT devices collect data and send data to the network layer (e.g., cloud or some other applications). Such devices are also responsible for data acquisition, preliminary processing, encryption (if they can support the encryption), and transmission. The devices can usually remotely request access and handle the commands. When the devices need to request data from other devices, they should publish a corresponding request to the cloud or the data owner.

(ii) **Data Owner.** There is a very large number of data owners, who are divided into administrators and ordinary data owners. The administrators are responsible for vetting the participants. When data owner receives an access data request from other IoT device, he/she should authenticate the identity before responding to the request accordingly.

(iii) **Blockchain Network.** In this architecture, we adopt a permissioned model (e.g., hyperledger fabric). Specifically, its security is guaranteed under the assumption that most participants are honest and the difficult problems. In other words, the average time an attacker

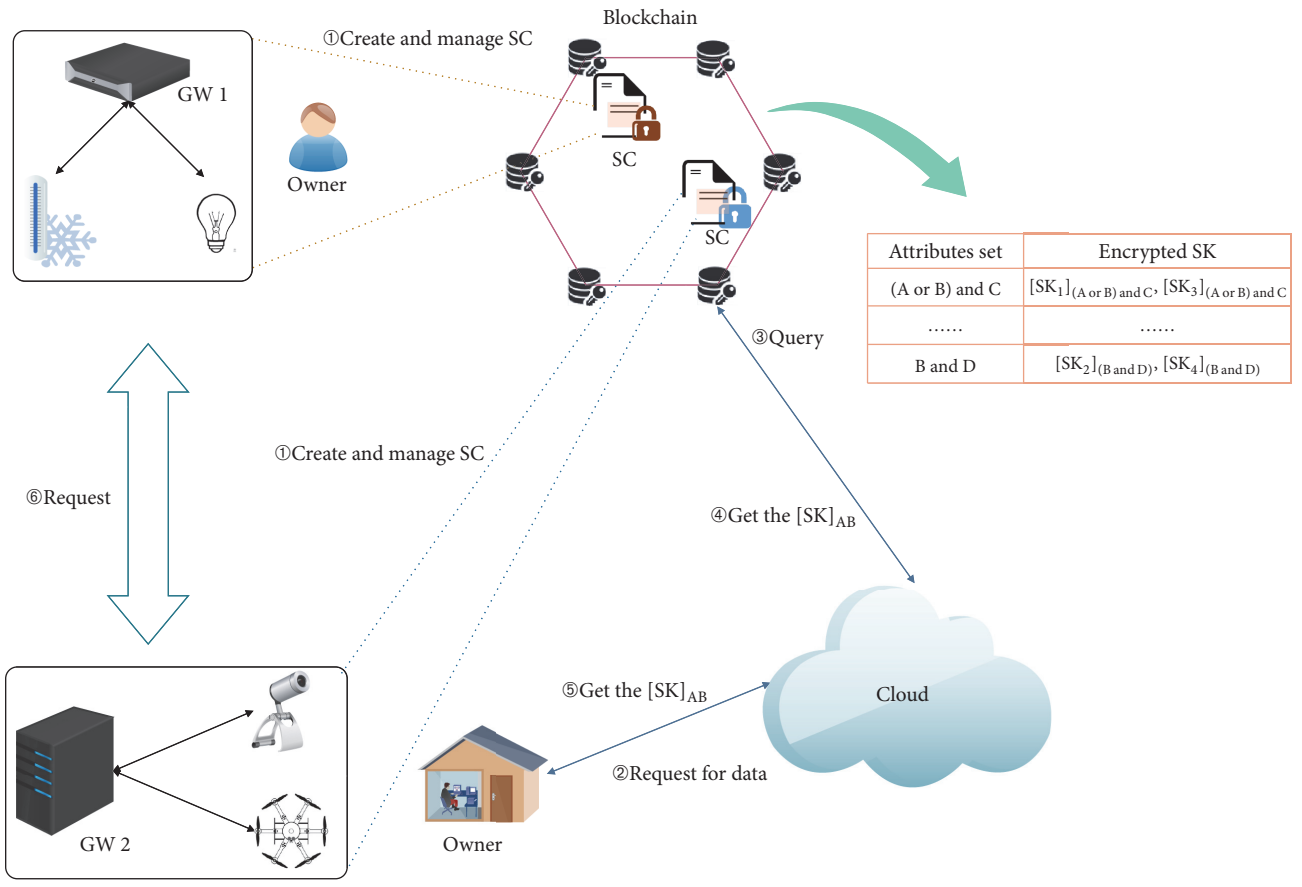


Figure 2: IoT device data sharing system.

can solve the problem is much less than the time it takes to disseminate information over the network. The consensus algorithm in our system is the Byzantine fault tolerance mechanism **PBFT**, rather than Proof of Work (**POW**) used in Bitcoin. The fabric contains verification nodes (to verify the transaction) and ordering nodes (to pack the verified transaction into the block). When the nodes receive the transaction (request from cloud or data owner), they will verify and pack them into the blockchain.

(iv) **Cloud.** It is used to store the encrypted devices' data, and sends a corresponding request transaction to the blockchain network to query the permission of the device, when the cloud receives a request from IoT device. That means the cloud monitors the blockchain network and responds to the requested data.

In the BaDS architecture, the following steps are undertaken to request data between devices.

At first, owner *A* sends the encrypted devices' data to the cloud and generates the responding access control table in the smart contract. When other device (e.g., device belonging to owner *B*) wants to access the data, it invokes the getPACT algorithm in the smart contract to obtain the predicates. If its attributes satisfy the predicates, then it sends a responding transaction to the smart contract with an attribute-based signature. If the signature can be successfully verified, then the device can receive the encrypted private key (CP-ABE).

Then, the device sends a request containing an attribute-based signature to the cloud to obtain the requested data. On receiving the request, the mutual authentication and session key between the cloud and device will be established by executing the existing authentication key agreement protocol [20]. After both sides have mutual authenticated each other, the device can get the encrypted data through the "secure channel" protected by the session key.

If the signature cannot be verified, it implies that the attributes of the device do not satisfy the policy of the data. Thus, owner *B* should sends a request for data access to owner *A*. Both owners will authenticate each other and generate a session key through the existing authentication key agreement protocol, and the session key is used to guarantee the subsequent session. If owner *A* permits the access from the device of *B*, he/she will send a transaction to the smart contract to update the access control table and the device can obtain the data from the cloud as before. Otherwise, *A* rejects the request.

2.4. Security Requirement. Based on recent literatures [20–22], the blockchain-based architecture for data sharing with ABS and CP-ABE needs to satisfy the following security requirements:

(1) **Confidentiality.** To protect the privacy of data, in this architecture, only devices which satisfy the attribute policy can access the data and get the corresponding decryption key.

(2) **Fine-Grained Access Control.** The data manager or authority generate the corresponding access policy for their data, and they can grant or revoke devices' access on a fine-grained basis, by modifying the access attributes.

(3) **Mutual Authentication.** To protect the safety of participants, our system should provide mutual authentication. The participants should authenticate their communicating partner.

(4) **User Anonymity.** To preserve privacy, the architecture should protect the device's anonymity. Even if the adversary analyzes a series of transactions, (s)he cannot learn the devices' real identity.

(5) **Impersonation Attack Resilience.** If the adversary impersonates a legitimate device and sends a request to the cloud, it cannot be authenticated due to the invalid attribute signature.

(6) **Collision Attack Resilience.** There is an extremely small possibility of generating two identical blocks at the same time. Thus, the system should resist collision attack.

(7) **Man-in-the-Middle Attack Resilience.** The device can identify and abandon the messages transmitted in the open environment, which have been intercepted or replaced by the adversary.

(8) **Link Attack Resilience.** Even if the adversary links multiple transactions which use the same address or public key, the adversary also cannot find users' private messages.

3. Definitions and Security Model

We briefly introduce the attribute-based signature, ciphertext-policy attribute-based encryption, permission access control table to generate the attribute policy, and PBFT (the consensus algorithm) used in our architecture, in this section.

3.1. Attribute-Based Signature (ABS). In the attribute-based signature scheme, the devices are tagged with a set of attributes whose certificates are issued by an attribute center [23]. Due to the fact that ABS scheme can provide fine-grained access control, we use ABS in our architecture to replace the original ECDSA signature in the blockchain.

A signature consists of $s+2$ elements, where s is the width of the monotone span program of the claim-predicate [24]. The maximum width of monotonic span program is defined as q_m, and $\mathbb{A} = \mathbb{Z}_p^*$, in which p is a big prime number and also the order of the cyclic group.

(i) ABS.PSetup. Let G_1, G_2, and $e : G_1 \times G_2 \longrightarrow G_T$ denote two cyclic groups and a bilinear map, respectively, where the order of cyclic groups is p. Suppose g and h_i are the generators of G_1 and G_2, where $i \in [0, q_m]$. Choose a hash function $H : \{0,1\}^* \longrightarrow \mathbb{Z}_p^*$. Thus, the public parameters are Param $= \{G_1, G_2, H, g, h_0, \ldots, h_{q_m}\}$.

(ii) ABS.MSetup. Randomly choose four numbers $l_0, l, m, n \in \mathbb{Z}_p^*$ and compute: $N = g^n$, $L_0 = h_0^{l_0}$, $L_j = h_j^l$, and $M_j = h_j^m$, $j \in [1, q_m]$. Thus, the master key is MSK $= \{l_0, l, m\}$, and the public key is MPK $= \{N, L_0, \ldots, L_{q_m}, M_1, \ldots, M_{q_m}\}$.

(iii) ABS.Gen. Input the master key MSK and an attribute set $\mathscr{A} \subseteq \mathbb{A}$. Choose a random generator K_{base} in cyclic group G_1 and compute $K_0 = K_{base}^{l_0^{-1}}$ and $K_u = K_{base}^{(l+mu)^{-1}}$, $u \in \mathscr{A}$. Thus, the device private key is $\mathrm{SK}_{\mathscr{A}} = \{K_{base}, K_0, K_u\}$, $u \in \mathscr{A}$.

(iv) ABS.Sign. Input the public key MPK, device private key $\mathrm{SK}_{\mathscr{A}}$, the message m, and a monotone Boolean function γ, in which $(\gamma(\mathscr{A}) = 1)$. $\gamma \longrightarrow \mathbf{MN} \in (\mathbb{Z}_p)^{x \times y}$, $u : [x] \longrightarrow \mathbb{A}$, and the vector $\overrightarrow{v}$ meets the assignment $\mathscr{A}$. Compute $\mu = H(m \parallel \gamma)$ and choose the random numbers $\beta_0 \in \mathbb{Z}_p^*$ and $\beta_1, \ldots, \beta_x \in \mathbb{Z}_p$. After that compute $C = K_{base}^{\beta_0}$, $F = K_0^{\beta_0}$, $S_i = (K_{u(i)}^{v_i})^{\beta_i} \cdot (Ng^{\mu})^{\beta_i}$, and $R_i = \prod_{i=1}^{x} (L_j \cdot M_j^{u(i)})^{T_{ij} \cdot \beta_i}$, $\forall i \in [x]$ and $\forall j \in [y]$. The signature is $\sigma = \{C, F, S_i, R_j\}$, $\forall i \in [x]$ and $\forall j \in [y]$.

(v) ABS.Veri. Input the public key MPK, the signature σ, the message m, and the monotone Boolean function γ. First compute $\gamma \longrightarrow \mathbf{MN} \in (\mathbb{Z}_p)^{x \times y}$, $u : [x] \longrightarrow \mathbb{A}$, and $\mu = H(m \parallel \gamma)$. If $C = 1$, then output 0. Otherwise, checks $e(F, L_0) = e(C, h_0)$ and

$$\prod_{i=1}^{x} e\left(S_i, \left(L_j M_j^{u(i)}\right)^{T_{ij}}\right) = \begin{cases} e\left(C, h_1\right) e\left(Ng^{\mu}, R_1\right), & j = 1 \\ e\left(Ng^{\mu}, R_j\right), & j > 1, \end{cases} \tag{2}$$

If they are all equal, then return 1; otherwise, return 0.

3.2. Ciphertext-Policy Attribute-Based Encryption (CP-ABE). Attribute-based encryption has been used to share data with some target devices which have specified attributes [19]. The data owner can make policies (s)he wishes to share the data with. The users will be assigned a secret key associated with the attributes, and they can decrypt (or access) the shared data if their attributes "satisfy" the predicates [25].

(i) ABE.Setup. Input an attribute set $\mathbb{A}$. Choose two multiplicative cyclic groups G_3 and G_T, whose order is p. Let $e : G_3 \times G_3 \longrightarrow G_T$ denote a bilinear map. Suppose k is a generator of G_3, and $f_1, \ldots, f_{|\mathbb{A}|}$ are $|\mathbb{A}|$ random elements in group G_3. Select two random numbers $s, a \in \mathbb{Z}_p$, and compute the public parameters $Param = \{k, G_3, f_1, \ldots, f_{|\mathbb{A}|}\}$ and the master private key ESK $= k^s$.

(ii) ABE.KeyGen. Input the master private key ESK and an attribute set $\mathscr{A}$. Choose a random number $t \in \mathbb{Z}_p$, and compute $sk = k^s k^{at}, T = k^t, sk_i = f_i^t (\forall i \in \mathscr{A})$. The private key is $SK = (sk, T, sk_i)$; the public key is $PK = \{e(k,k)^s, k^a\}$.

(iii) ABE.Encrypt. Input the public parameters *Param*, the public key *PK*, the message m, and an Linear Secret Sharing Scheme (LSSS) access structure (M, β). M is a $x \times y$ matrix and β is a function that links rows of M with attributes. Choose a vector $\overrightarrow{v} = (l, b_2, \ldots, b_y)^T \in \mathbb{Z}_p^y$, and compute $\gamma_i = \overrightarrow{v} \cdot M_i$, M_i is the ith row of matrix M. Select x random numbers $n_1, \ldots, n_x \in \mathbb{Z}_p$, and compute $C = Me(k,k)^{sl}, C' = k^l$, $(C_1 = k^{a\gamma_1} f_{\beta(1)}^{-n_1})$,..., $(C_x = k^{a\gamma_x} f_{\beta(x)}^{-\gamma_y}, D_x = k^{n_x})$. The ciphertext is $CT = (C, C', (C_1, D_1), \ldots, (C_x, D_x))$.

(iv) ABE.Decrypt. Input the private key *SK* for attribute set $\mathscr{A}$ and the ciphertext *CT* for (M, β). Suppose $\mathscr{A}$ satisfies (M, β) and define $U = \{i : \beta(i) \in \mathscr{A}\}$. Let $\rho_i \in \mathbb{Z}_p$, $i \in U$. and $\Sigma_{i \in U} \rho_i \gamma_i = l$. Decrypt the ciphertext:

$$\frac{e\left(C', sk\right)}{\left(\prod_{i \in U} \left(e\left(C_i, T\right) e\left(D_i, sk_{\beta(i)}\right)\right)^{\rho_i}\right)} = \frac{e\left(k,k\right)^{sl} e\left(k,k\right)^{alt}}{\prod_{i \in U} e\left(k,k\right)^{ta\gamma_i \rho_i}} = e\left(k,k\right)^{sl} \quad (3)$$

3.3. Permission Access Control Table PACT. We generate a permission access control table (PACT) to achieve fine-grained access control by using smart contract. The owner first deploys smart contract with the access control table in the blockchain. This allows other devices to request and/or access data when their attributes satisfy the predicates. For example, "(A or B) and C" mapping to "$[SK_1]_{(AorB)andC}$" and "$[SK_3]_{(AorB)andC}$" mean the devices which satisfy the attribute "(A or B) and C" can access the encrypted private keys of device with identities 1 and 3. Only the device (or smart contract) owner can update the PACT by calling the smart contract function.

3.4. Practical Byzantine Fault Tolerance (PBFT). The consensus algorithm used in this paper is Practical Byzantine Fault Tolerance (PBFT). We assume that there are a total of **3f + 1** nodes in the system, where **f** is the maximum number of nodes that may be failed. When more than **2f + 1** normal authorized nodes confirm the transaction, the authorized nodes come to a consensus. This means that users will eventually receive replies from authorized nodes pertaining to their requests.

This algorithm is suitable for asynchronous systems such as the Internet. It contains important optimization functions that enable it to be executed efficiently. Here, we introduce the working process of PBFT, which consists of the following five phases: **Request**, **Preprepare**, **Prepare**, **Commit,** and **Replay**.

(i) **Request.** When the leader is found not to be honest, the other replica is elected as a new leader by the algorithm. The primary sends a request to a replica; here it is replica 0.

(ii) **Preprepare.** When replica 0 receives the request, it broadcasts a preprepared message to other replicas.

(iii) **Prepare.** When the other replicas receive the preprepared message, if they accept, they broadcast the prepare message to all the other replicas and add preprepare and prepare messages into their logs. Otherwise, they do nothing.

(iv) **Commit.** When replicas receive more than a certain number $(2f)$ of prepare messages during the **Prepare** phase, it enters the **Commit** phase. The replicas broadcast commit message.

(v) **Replay.** If more than $2f+1$ replicas accept the commit message, it means that there is a replica receiving more than $2f+1$ commit messages. After completing the request operation, each replica sends a replay message to the primary node.

Both **Preprepare** and **Prepare** phases are used to ensure the ordering of the request. The consensus algorithm does not rely on the orderly propagation of messages, so replicas can submit requests in a disorderly manner. Because each replica backs up the message log in the preprepare, prepare, and commit phases, the corresponding requests can be executed in order.

3.5. Smart Contract. The concept of smart contract is introduced in 1994 by Nick Szabo and defined as "a computerized transaction protocol that executes the terms of a contract" [26]. Smart contracts are autonomous scripts stored on blockchain and have unique addresses. The creator can compile, deploy, and update his/her smart contract, and the output is recorded in blockchain network as a transaction. In our architecture, we use smart contract to manage PACT. The devices can send a request to the cloud with its signature, and the cloud interacts with the smart contract to verify the signature and retrieve the policy.

4. Proposed BaDS Architecture

In the proposed BaDS architecture, the devices use their attribute private key to sign the data request. The architecture comprises **Initialization**, **Request and Transaction for Cloud,** and **Request and Transaction for Owner**. Let us assume that a device belonging to owner A wants to access the data of owner B. The steps below are carried out among the parties.

(1) **Initialization.** In this phase, the authority or system manager selects the system private key and computes the parameters by executing the following steps.

(i) **ABS.Initialization.** As explained in Section 3.1, select a maximum width of monotonic span program q_m and cyclic groups G_1, G_2. Generate the public parameters $Param = \{G_1, G_1, H, g, h_0, h_{q_m}\}$. Choose the master key $MSK = \{l_0, l, m\}$, and compute the public key $MPK = \{N, L_0, \ldots, L_{q_m}, M_1, \ldots, M_{q_m}\}$ by

calling ABS.PSetup and ABS.MSetup. When each device registers on the system for the first time, based on its attribute tags $\mathscr{A}$, manager generates the private key $SK_{\mathscr{A}} = \{K_{base}, K_0, K_u\}$ for device using the ABS.Gen algorithm.

(ii) **ABE.Initialization.** Select a cyclic group G_3 and a generator k. Choose the master key $ESK = k^s$, and generate the public parameters $Param = \{k, G_3, f_1, \ldots, f_{|\mathbb{A}|}\}$ by calling ABE.Setup. As explained in Section 2.3, the assumption is that there are π IoT devices, two device managers (e.g., owner A and owner B) and a cloud in the system.

When the nodes register on the system for the first time, based on their attribute tags $\mathscr{A}$, manager generates their public and private key pairs $(PK_j, SK_j)(j = 1, \ldots, \pi, oa, ob, c)$ using ABE.KeyGen. $PK_j = \{e(k,k)^s_j, k^a_j\}$, and $SK_j = \{sk_j, T_j, sk_{ij}\}$, where $sk_j = k^s_j k^{at}_j, T_j = k^t_j, sk_{ij} = f^t_{ij}(\forall i \in \mathscr{A})$.

(iii) **Contract Deployment.** First, we generate a smart contract which is designed to achieve permission access control table. Then we compile and deploy it on the blockchain, after that the smart contract will have its own address (e.g., PID). The access control table is made up of the device access policy and corresponding predicates. The data sharing private keys are encrypted with relevant attributes.

(2) **Request and Transaction for Cloud.** In this phase, a device belonging to owner A invokes the algorithm in the smart contract for sharing data belonging to other devices. It executes as follows.

(i) **Query.** The device invokes the getPACT algorithm in smart contract to get the corresponding predicate of the target device's access policy. And it checks whether its attributes can satisfy the predicate or not. If yes, then it sends a transaction to smart contract. Otherwise, the device requests permission from owner.

(ii) **Transaction.** The device prepares and constructs the corresponding transaction based on its request. For instance, a transaction consists of $\{to, from, value\}$ and other parts, "to" is filled with the address of smart contract that the device wants to call, and "from" is filled with the device's address. After all the fields are constructed, the device uses its attribute private key $SK_{\mathscr{A}}$ to sign the transaction and broadcasts it to the blockchain network.

(iii) **State.** The other nodes invoke the ABS.Veri algorithm to verify the attribute-based signature in it, when they receive the broadcast transaction. The architecture uses the PBFT consensus mechanism to achieve consensus, and the transaction can be recorded in the blockchain network, only if there are at least two-thirds nodes that have accepted it.

(iv) **Response.** The smart contract returns the attribute-based encrypted private key in the access policy as a response message to the device, after the transaction is recorded in the blockchain network. The device can use its attribute private key SK_j to get the private key. Then, the device sends a request to the cloud. The two parties (between device and cloud) should authenticate each other using the existing authentication protocol [20] before the cloud deals with the request. Finally, the device obtains the data from the cloud and uses the private key to decrypt the data.

(3) **Request and Transaction for Owner.** In this phase, the device requests permission from data owner. It executes as follows.

(i) **Request.** The device invokes the getPACT algorithm in smart contract to get the corresponding predicate of the target device's access policy. If its attributes cannot satisfy the predicate, then it requests for permission from the owner. Both device and owner should authenticate each other using the existing authentication protocol [20], prior to the owner dealing with the request. If the owner allows the device to access the data, then he/she calls the smart contract to modify the access control table by transaction. Otherwise, the device cannot access the data.

(ii) **Permission Update.** As described in Section 3.3, we use permission access control table in smart contract to achieve fine-grained access control. The data owner can invoke the UpdatePACT algorithm in smart contract to add, delete, and modify the access policies and the predicates. Only if the device's attributes satisfy the predicates in PACT can it obtain the encrypted key from access policies.

(iii) **Access.** After the data owner has modified PACT, the device sends a request to cloud and performed as described in **Request and Transaction for Cloud**.

4.1. Security Analysis. In this section, we analyze how the architecture is resilient to the following typical security and privacy attacks.

(1) **Confidentiality.** The permission access control table (PACT) is generated to restrict access; if an adversary sends request to cloud or smart contract, the request will be rejected owing to his/her invalid signature. Thus, only authorized (satisfied attribute policy) devices can access the data and get the corresponding decryption key.

(2) **Fine-Grained Access Control.** The attribute-based signature provides fine-grained access control. In other words, the data manager or authority generate policies (some attributes set), and only devices satisfying the policies can access the data. In addition, the manager can grant or revoke device access by modifying the policy.

(3) **Mutual Authentication.** Before the device can communicate securely with the cloud or data owner, they will confirm the identity of each other by their signatures. Any probabilistic polynomial time adversary cannot forge a valid signature due to the underpinning DL problem. Hence, mutual authentication can be achieved between the device and the cloud or data owner.

(4) **User Anonymity.** In the BaDS architecture, we use attribute-based signature and encryption to protect the devices' real identities. All the message transmitted in the open channel are signed or encrypted by some attributes or session key. Thus, when verifying the signature or decrypting the information, only the attributes public or secret keys are needed. Hence, the architecture can protect user anonymity.

(5) **Impersonation Attack.** As discussed earlier, mutual authentication between devices and cloud or data owner is achieved in the BaDS architecture. If the adversary impersonates a legitimate device and sends a request to cloud, it cannot be authenticated, and the manager can revoke access of the malicious or compromised device. That is, only a legitimate device can generate a valid signature.

(6) **Collision Attack Resistance.** In the BaDS architecture, we use the PBFT consensus algorithm to record new blocks, which effectively avoids collisions of blocks.

(7) **Man-in-the-Middle Attack Resistance.** Man-in-the-middle attack means the adversary can intercept and replace the encrypted date transmitted in the open environment. Suppose that if an adversary modifies other response message, the device can identify them due to the use of the attribute signature and session key encryption; thus we can say that the adversary cannot modify the transaction message.

(8) **Link Attack Resistance.** Link attack is defined as the adversary can link multiple transaction, which use the same address or public key, to find users' private messages. Similar to our explanation for man-in-the-middle attack, all messages transmitted in the open environment are signed or encrypted by the attributes and session key. Even if the adversary obtains session key, he/she cannot extract something useful due to the attribute encryption of real device's data.

5. Performance Analysis

In this section, we implement the BaDS architecture and analyze the computation cost of smart contract based on Ethereum (https://www.ethereum.org/). The blockchain platform allows one to write smart contract with a special language, and compile and deploy it to the blockchain network. The smart contracts are autonomous scripts stored on blockchain and have unique addresses. Thus, it can be regarded as a database in which function can be called by sending a transaction with corresponding parameters.

Table 1: Simulation platform.

Operating System	Ubuntu 16.04
CPU	Intel (R) Core (TM) i7-6700 CPU @ 3.40 GHZ
Memory	3 GB RAM
Configuration	go-ethereum nodejs npm truffle

We define Auth.KA as authentication key agreement algorithm and PACT.deploy and SC.deploy as the deployment of permission access control table in smart contract and the deployment of smart contract in blockchain, respectively. getPACT and UpdataPACT denote invoking the corresponding get and update functions in smart contract. The operations needed at each phase in this architecture is shown in Table 2. We then evaluate the operation computation cost of those algorithm (e.g., ABS, CP-ABE) by using the pairing-based library and GNU multiple precision arithmetic library. Table 3 shows the computation cost of attribute-based signature and encryption algorithms.

We publish the smart contract on a private Ethereum network, which we constructed by ourselves, then we can compute the time of deploying and invoking a smart contract. Publishing transactions on private chain does not need transaction fees and has the same accurate results as public chains. Table 1 presents the information of the simulation platform. *Web3j* is used to evaluate the time cost of publishing a designed smart contract. However, the existing Ethereum platform does not provide ABS and ABE algorithm; in this research, we only use the smart contract to realize management of policies and execute the signing and encrypting in the external environment.

6. Conclusion

We proposed a novel blockchain-based architecture for data sharing with attribute-based cryptosystem (BaDS) in this paper. The architecture can achieve privacy-preserving, user-self-controlled data sharing, and decentralization by using blockchain and several attribute-based cryptosystems. Specifically, ABS and CP-ABE provide the capability for fine-grained access control. We introduced the security requirements of the proposed BaDS architecture and then explained how the proposed BaDS architecture satisfies the security requirement. We also implement the BaDS architecture and analyze its computation cost.

Table 2: Operations needed at each phase in BaDS.

Phase	Cryptographic Algorithms	Smart Contract Publication	Smart Contract Trigger
Initialization	ABS.PSetup + ABS.MSetup +ABS.Gen + ABE.Setup + ABE.KeyGen + ABE.Encrypt	PACT.deploy + SC.deploy	Null
Request and Transaction for cloud	Auth.KA + ABE.Decrypt + ABS.Sign + ABS.Veri	Null	getPACT
Request and Transaction for Owner	Auth.KA + 2∗ABS.Sign + 2∗ABS.Veri	Null	getPACT + UpdataPACT

Table 3: Computation cost(s) of cryptographic algorithms.

ABS				
Algorithm	ABS.PSetup & ABS.MSetup	ABS.Gen	ABS.Sign	ABS.Veri
Max Time	0.17939	0.068969	0.136605	0.18168
Min Time	0.049207	0.021929	0.055461	0.050144
Average Time	0.058182	0.028793	0.069066	0.064048
CL-MRE				
Algorithm	ABE.Setup	ABE.KeyGen	ABE.Encrypt	ABE.Decrypt
Max Time	0.161446	0.058168	0.164257	0.234115
Min Time	0.037582	0.017119	0.050459	0.025966
Average Time	0.045421	0.02222	0.05951	0.035304

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work was supported in part by the National Key Research and Development Program of China (no. 2018YFC1315400), the National Natural Science Foundation of China (nos. 61572370, 61572379, and 61501333), and the fund of the Jiangsu Key Laboratory of Big Data Security & Intelligent Processing (no. BDSIP1807).

References

[1] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao, "A Survey on Security and Privacy Issues in Internet-of-Things," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1250–1258, 2017.

[2] H. Debiao, K. Neeraj, W. Huaqun, L. Wang, K. K. R. Choo, and A. Vinel, "A provably-secure cross-domain handshake scheme with symptoms-matching for mobile healthcare social network," *IEEE Transactions on Dependable & Secure Computing*, vol. 15, no. 4, pp. 633–645, 2018.

[3] Y. Yu, J. Ni, M. H. Au, Y. Mu, B. Wang, and H. Li, "Comments on a public auditing mechanism for shared cloud data service," *IEEE Transactions on Services Computing*, vol. 8, no. 6, pp. 998-999, 2015.

[4] H. Debiao, K. Neeraj, K. M. Khurram, L. Wang, and J. Shen, "Efficient privacy-aware authentication scheme for mobile cloud computing services," *IEEE Systems Journal*, vol. 12, no. 2, pp. 1621–1631, 2018.

[5] W. Ding, Z. Yan, and R. Deng, "Privacy-preserving data processing with flexible access control," *IEEE Transactions on Dependable and Secure Computing*, 2017, In press.

[6] K. Yang, X. Jia, and K. Ren, "Attribute-based fine-grained access control with efficient revocation in cloud storage systems," in *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security (ASIACCS '13)*, pp. 523–528, Hangzhou, China, May 2013.

[7] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Consulted, 2008.

[8] I. Miers, C. Garman, M. Green, and A. D. Rubin, "Zerocoin: Anonymous distributed e-cash from bitcoin," in *Proceedings of the 34th IEEE Symposium on Security and Privacy, SP 2013*, pp. 397–411, Berkeley, Calif, USA, May 2013.

[9] H. Yining, A. Manzoor, P. Ekparinya et al., "A delay-tolerant payment scheme based on the ethereum blockchain," 2018.

[10] H. Kopp, D. Mödinger, F. Hauck, F. Kargl, and C. Bösch, "Design of a privacy-preserving decentralized file storage with financial incentives," in *Proceedings of the 2nd IEEE European Symposium on Security and Privacy Workshops, EuroS and PW 2017*, pp. 14–22, Paris, France, April 2017.

[11] H. Kopp, C. Bösch, and F. Kargl, "KopperCoin – a distributed file storage with financial incentives," in *Information Security Practice and Experience*, vol. 10060 of *Lecture Notes in Computer Science*, pp. 79–93, Springer International Publishing, Cham, 2016.

[12] X. Liang, S. Shetty, D. Tosh, C. Kamhoua, K. Kwiat, and L. Njilla, "ProvChain: A Blockchain-Based Data Provenance Architecture in Cloud Environment with Enhanced Privacy and Availability," in *Proceedings of the 17th IEEE/ACM International*

Symposium on Cluster, Cloud and Grid Computing, CCGRID 2017, pp. 468–477, Madrid, Spain, May 2017.

[13] M. Banerjee, J. Lee, and K. R. Choo, "A blockchain future for internet of things security: a position paper," *Digital Communications and Networks*, vol. 4, no. 3, pp. 149–160, 2018.

[14] C. Lin, D. He, X. Huang, K. R. Choo, and A. V. Vasilakos, "BSeIn: A blockchain-based secure mutual authentication with fine-grained access control system for industry 4.0," *Journal of Network and Computer Applications*, vol. 116, pp. 42–52, 2018.

[15] S. Huh, S. Cho, and S. Kim, "Managing IoT devices using blockchain platform," in *Proceedings of the 19th International Conference on Advanced Communications Technology, ICACT 2017*, pp. 464–467, Bongpyeong, Republic of Korea, February 2017.

[16] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, "Smart Contract-Based Access Control for the Internet of Things," *IEEE Internet of Things Journal*, 2018, In press.

[17] 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 (CCS '06)*, pp. 89–98, Alexandria, Va, USA, November 2006.

[18] K. H. Maji, M. Prabhakaran, and M. Rosulek, "Attribute-based signatures," *IACR Cryptology ePrint Archive*, vol. 2010, article 595, 2010.

[19] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *PKC 2011: Public Key Cryptography – PKC 2011*, Lecture Notes in Computer Science, pp. 53–70, Springer, Berlin, Germany, 2011.

[20] D. He, S. Zeadally, N. Kumar, and W. Wu, "Efficient and Anonymous Mobile User Authentication Protocol Using Self-Certified Public Key Cryptography for Multi-Server Architectures," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 9, pp. 2052–2064, 2016.

[21] N. Z. Aitzhan and D. Svetinovic, "Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 5, pp. 840–852, 2018.

[22] Q. Feng, D. He, S. Zeadally, N. Kumar, and K. Liang, "Ideal lattice-based anonymous authentication protocol for mobile devices," *IEEE Systems Journal*, pp. 1–11.

[23] J. Li, M. H. Au, W. Susilo, D. Xie, and K. Ren, "Attribute-based signature and its applications," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communication Security (ASIACCS '10)*, pp. 60–69, Beijing, China, April 2010.

[24] H. K. Maji, M. Prabhakaran, and M. Rosulek, "Attribute-based signatures," in *CT-RSA 2011: Topics in Cryptology – CT-RSA 2011*, vol. 6558 of *Lecture Notes in Computer Science*, pp. 376–392, Springer, Heidelberg, Germany, 2011.

[25] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology – EUROCRYPT 2005*, vol. 3494 of *Lecture Notes in Computer Science*, pp. 457–473, Springer, Berlin, Germany, 2005.

[26] N. Szabo, "The idea of smart contracts," *Nick Szabos Papers and Concise Tutorials*, vol. 6, 1997.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 4029591, 14 pages
https://doi.org/10.1155/2018/4029591

Research Article

A Blockchain-Based Contractual Routing Protocol for the Internet of Things Using Smart Contracts

Gholamreza Ramezan and Cyril Leung

Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, Canada

Correspondence should be addressed to Gholamreza Ramezan; gramezan@ece.ubc.ca

Received 27 July 2018; Accepted 10 October 2018; Published 1 November 2018

Guest Editor: Jiageng Chen

In this paper, we propose a novel blockchain-based contractual routing (BCR) protocol for a network of untrusted IoT devices. In contrast to conventional secure routing protocols in which a central authority (CA) is required to facilitate the identification and authentication of each device, the BCR protocol operates in a distributed manner with no CA. The BCR protocol utilizes smart contracts to discover a route to a destination or data gateway within heterogeneous IoT networks. Any intermediary device can guarantee a route from a source IoT device to a destination device or gateway. We compare the performance of BCR with that of the *Ad-hoc* On-Demand Distance Vector (AODV) routing protocol in a network of 14 devices. The results show that the routing overhead of the BCR protocol is 5 times lower compared to AODV at the cost of a slightly lower packet delivery ratio. BCR is fairly resistant to both Blackhole and Greyhole attacks. The results show that the BCR protocol enables distributed routing in heterogeneous IoT networks.

1. Introduction

Recent progress in wireless communications and mobile computing has enabled a large variety of devices to connect to the Internet, forming the Internet of Things (IoT) [1, 2]. The IoT is a heterogeneous network of various types of devices from different vendors which collect, transfer, process, and analyze data and take appropriate actions [3, 4]. The IoT faces numerous challenges due to the need to integrate a large number of dissimilar objects.

Routing, which establishes a communication path from a source IoT device to a destination node, for example, a gateway, is one such challenge. A variety of routing protocols for IoT networks have been studied [5–9]. In [5], a routing protocol for low-power and lossy networks (RPL) was proposed. The RPL protocol is a promising routing protocol that is used in the large-scale BC Hydro smart meter project in British Columbia, Canada [10]. Providing secure communication and preventing attackers from interfering with the routing process are major concerns in this network.

The utilization of cryptographic algorithms is the first approach in securing routing protocols. However, in the design of most existing routing protocols, such as Secure *Ad-hoc* On-Demand Distance Vector (SAODV) [11], Ariadne [12], Optimized Link State Routing (OLSR), and optimal and secure routing (OSR) [13], the availability of a central authority (CA) to distribute the secret keys between network nodes is assumed [14–16]. The major problem is that the large number of IoT vendors cannot simply agree on a centralized management system. This is due to the trust issue between IoT vendors and the high cost of implementing trust management infrastructures such as the Public Key Infrastructure (PKI).

The second approach is the reputation-based method that measures the degree to which a network node contributes to the routing process [17, 18]. In [17], a reward mechanism is proposed to incentivize nodes to participate in the routing process. Each network node is selected based on its reputation in the routing process. The reputation information is derived either from observing the behaviour of its neighbors or from trusted external advisors in the network. In both cases, the accuracy of the reputation system can be affected either because of the limited network view of a network node based solely on viewing its neighbors, or from its attackers'

falsification of reputation information coming from external trusted systems [19].

The lack of trust in a central management system and the need for a publicly verifiable reputation system lead us to leverage public ledger techniques, such as blockchain, to design routing protocols for the IoT.

In this paper, we introduce a decentralized blockchain-based contractual routing (BCR) protocol. The BCR protocol enables IoT devices from diverse vendors to trust one another and cooperate during data communication. Using this approach, the devices in a delay-tolerant IoT network can find routes to a gateway or destination device in a decentralized manner. The main contributions of the paper are as follows:

(i) We propose contractual routing as a blockchain-based routing protocol for the IoT. A public ledger system is used to decentralize the BCR protocol.

(ii) We provide a proof of concept of the BCR protocol using the Ethereum blockchain and consider the following four performance metrics: Packet Delivery Ratio (PDR), Throughput (TP), Routing Overhead (RO), and Route Acquisition Latency (RAL).

(iii) We compare the performance of BCR with that of *Ad-hoc* On-Demand Distance Vector (AODV) which is a commonly used routing protocol [20]. Our results show that the BCR has a slightly lower PDR but a much lower routing overhead.

(iv) We study the performance of BCR under Blackhole and Greyhole attacks by malicious devices which do not necessarily follow the smart contract rules.

The remainder of this paper is structured as follows. In Section 2, we review related works. Section 3 presents the system model. In Section 4, we discuss the attack model. In Section 5, we describe the proposed routing protocol. In Section 6, we compare the performances of the BCR and AODV protocols. Finally, the main conclusions are discussed in Section 7.

2. Related Works

Financial incentive models have been introduced in various works [17, 18, 21–24]. For example, *Ad-hoc* VCG [17] provides a game-theoretical setting for routing within mobile *ad-hoc* networks in which a node accepts a payment for forwarding data packets from other agents provided the payment exceeds its cost. The system provides the incentive for users to cooperate. In [18], Sprite is proposed as a model to reward each participant node when routing data packets. However, the approach still requires that nodes access a central system, such as a bank, to send a proof message which shows a data packet is delivered. The proof message includes digital signatures and node identities, so as to receive rewards from the bank. This method is vulnerable, as attackers can forge a proof message to be sent to a central management system to generate rewards. The Onion Router proposed in [23] is based on [24], a blockchain-based reward mechanism for anonymous routing. This routing needs a centralized network since it requires that nodes be assigned to their specific relay nodes, after which only these nodes will receive the data. The authors of [22] introduce the idea of monetizing routing protocols based on public ledger techniques, whereby reputation is traded as an asset. In contrast, we propose a communications network model and describe an implementation of our proposed decentralized BCR protocol. Furthermore, we analyze the performance of the proposed protocol.

3. System Model

In this section, we describe a model to implement the proposed BCR protocol. The system consists of a multihop IoT network $\mathcal{I}_{\mathcal{S},\mathcal{I},\mathcal{D}}$ which cooperates with blockchain network $\mathcal{B}^{\mathcal{P},\mathcal{Q},\mathcal{G}}_{\mathcal{BP},\mathcal{BG}}$, as shown in Figure 1.

3.1. Multihop IoT Network. The IoT network $\mathcal{I}_{\mathcal{S},\mathcal{I},\mathcal{D}}$ consists of a set of Source devices $\mathcal{S}$, a set of intermediary devices $\mathcal{I}$, and a set of Destination devices and Data gateways $\mathcal{D}$. There is no central management for registration, authentication, or device authorization. The source device aims to send data to a destination device or a data gateway.

(i) ***Source devices*** $\mathcal{S}$: A Source device originates a request access to send data to a destination, or a data gateway. The gateway allows the source device access to the Internet to periodically update firmware or upload data to its vendors' cloud.

(ii) ***Intermediary devices*** $\mathcal{I}$: The devices with no direct connection to a destination or data gateway use other devices to relay their traffic. An IoT device that relays source device data traffic to a gateway or destination is referred to as an intermediary device.

(iii) ***Destination devices*** or ***Data gateways*** $\mathcal{D}$: Data gateways provide source devices access to larger networks, or the Internet. Data gateways can be access points within Wi-Fi networks, base stations in Multihop Cellular Networks (MCN) [25], or sink nodes in Wireless Sensor Networks (WSN) [6].

3.2. Blockchain Network. The system includes a blockchain network denoted by $\mathcal{B}^{\mathcal{P},\mathcal{Q},\mathcal{G}}_{\mathcal{BP},\mathcal{BG}}$ with the following parameters, components, and capabilities:

(1) Parameters: The blockchain has the following parameters [26]:

(i) **Common Prefix property $\mathcal{P}$ with the parameter $\mathcal{K} \in \mathbb{N}$:** Suppose the honest blockchain nodes, $\mathcal{BN}_1$ and $\mathcal{BN}_2$, maintain chains $\mathcal{C}_1$ and $\mathcal{C}_2$; then $\mathcal{C}_1^{-\mathcal{K}}$ would be a prefix of $\mathcal{C}_2$ and $\mathcal{C}_2^{-\mathcal{K}}$ a prefix of $\mathcal{C}_1$, where $\mathcal{C}^{-\mathcal{K}}$ is Chain $\mathcal{C}$ minus its last $\mathcal{K}$ blocks. We would call $\mathcal{K}$ the depth parameter.

(ii) **Chain Quality property $\mathcal{Q}$ with parameters $\mathcal{L} \in \mathbb{N}$ and $\mu \in (0, 1]$,** where $\mathcal{L}$ is the length of the blockchain owned by an honest node and $1 - \mu$ is the ratio of the greatest chain that can be created by an adversary. μ is called the chain quality coefficient.

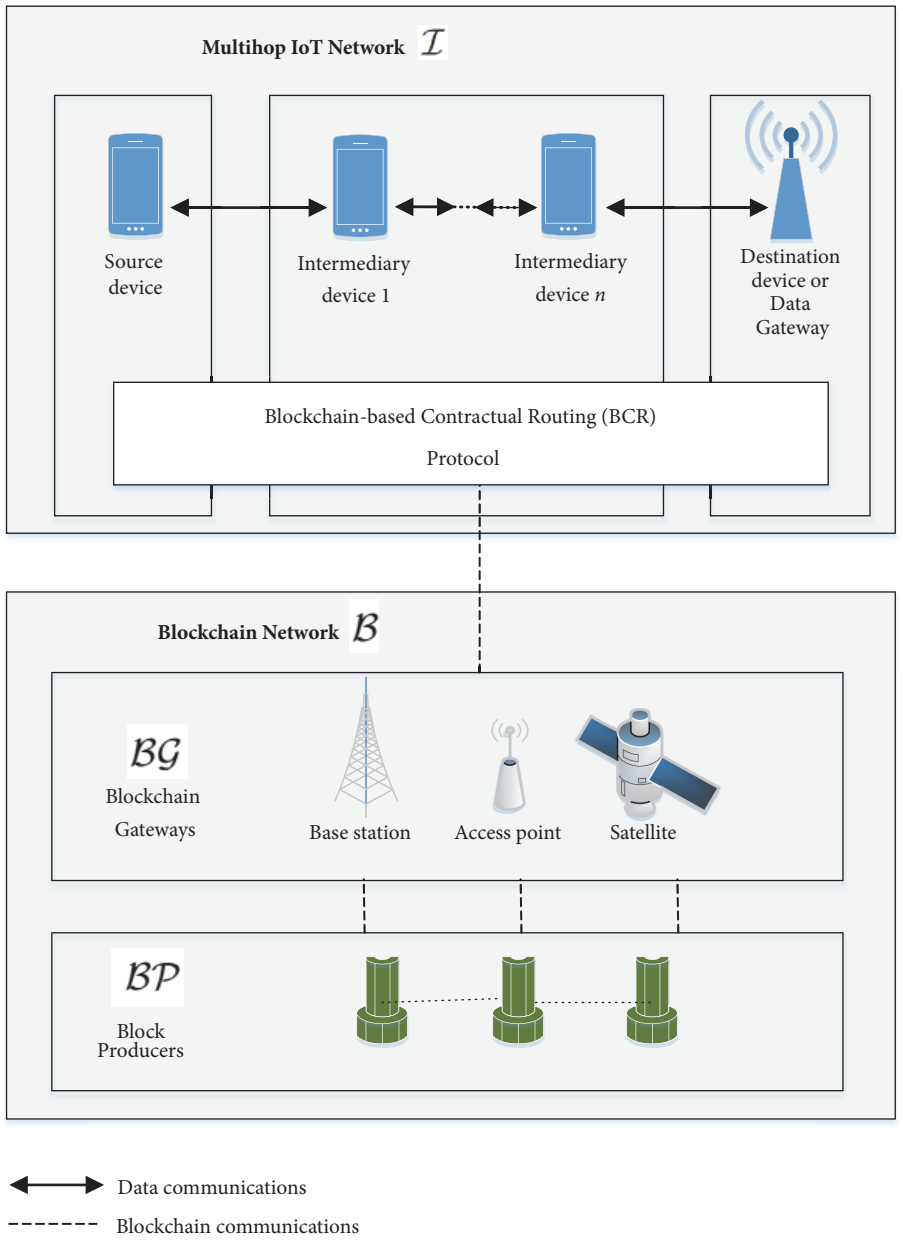


Figure 1: Setup for a decentralized communications network for IoT devices.

(iii) **Chain Growth property $\mathcal{G}$ with parameters $\mathcal{S} \in \mathbb{N}$ and $\tau \in (0, 1]$**, where, for any honest blockchain nodes, $\mathcal{BN}_1$ with Chain $\mathcal{C}$ for any $\mathcal{S}$ block times at least $\tau.\mathcal{S}$ blocks will be added to the blockchain. τ is called the speed coefficient.

The above parameters imply that the public ledger has the following two properties [27]:

(i) ***Liveness***: A submitted transaction from a network node to the blockchain block producers will appear in a block after a sufficient period of time. In other words, all transactions originating from the network nodes will eventually end up at a block within the blockchain.

(ii) ***Persistence***: Persistence means that once a transaction goes into the blockchain of one honest block producer, it will be included with very high probability in every honest block producer's blockchain and be consequently assigned a permanent position in the blockchain.

(2) Components: Our proposed blockchain network contains block producers $\mathcal{BP}$ and blockchain gateways $\mathcal{BG}$ as components:

(i) ***Blockchain gateways*** $\mathcal{BG}$: The blockchain gateways enable communication between IoT devices and the blockchain network. These gateways may be cellular base stations, Wi-Fi access points, or satellites.

(ii) ***Block Producers*** $\mathcal{BP}$: Each block producer receives transactions from the IoT network and assembles them into a block. It then attempts to add the newly generated block into the blockchain. Block producers may belong to IoT device vendors but none of them are trusted by other block producers. They must come to a consensus through blockchain consensus mechanisms about the transactions. Depending on the applied consensus algorithm, different security assumptions should be considered to preserve the properties of liveness and persistence. For example, the honest block producers should control at least 75% of the processing power in the block producers network if the Proof-of-Work (PoW) consensus mechanism is used [28].

(3) Capabilities: To apply blockchain technology to our system model, the blockchain network $\mathcal{B}$ should be capable of running programs. Several works have developed programming frameworks that take in high-level programs as specifications and generate cryptographic implementations [29–31]. The idea of programmable *smart contracts* dates back nearly twenty years [32]. Ethereum [29] is the first Turing-complete decentralized smart contract system. A contract can be run by calling on one of its functions, where each function is defined by a sequence of instructions. The smart contract maintains an internal state and can receive/transfer blockchain tokens from/to users or other smart contracts. Users send transactions to the Ethereum block producers network to invoke functions. Each transaction may contain input parameters for the contract and an associated token amount which is transferred from the user to the smart contract. The authors of [30] propose Hawk as a framework for building privacy preserving smart contracts. The Hawk compiler is in charge of compiling the program to a cryptographic protocol between the blockchain and its users. Hyperledger [31] is another blockchain development platform which supports smart contracts. Smart contracts on the Hyperledger platform are called chaincodes.

All the IoT devices, block producers, and gateways agree on the monetary value of a token. One of the ways for an IoT device to acquire tokens is by direct deposit from its owner into its blockchain address. The tokens can also be acquired from smart contracts. When an IoT device provides services, such as routing services for other IoT devices, the tokens assigned to a smart contract can be transferred from the smart contract address to the IoT device address on the blockchain.

4. Attack Model

In this section, we define the attackers' capabilities when they attack the BCR protocol. Attackers can be classified into two main categories: selfish and malicious nodes. A selfish node does not intentionally disrupt routing, but it drops other nodes' routing messages while using their resources to route its own messages. Detecting and mitigating a selfish node is difficult, since the node does not actively violate the routing protocol rules. Malicious nodes purposefully disrupt routing messages [22]. An attacker is a dishonest IoT device which holds a sufficient number of tokens to allow it to join a network and then attempts to interfere with the network's routing process by preventing honest IoT devices from accessing the data gateways.

(1) Anonymity: The network does not use any centralized authority to authenticate IoT devices. Any IoT device can generate its own private/public key pair. Based on the generated public key, the IoT device derives its own blockchain address. This provides anonymity for the network nodes because no one knows the identity of the owner of a new blockchain address.

(2) Token-based Authorization: Every IoT device which possesses a sufficient number of blockchain tokens is authorized to generate a smart contract and request a route to a gateway.

(3) Attacker's Violation Scope: An attacker can manipulate the routing protocol in its own IoT device. Therefore, it can violate the routing protocol procedures and rules. It is assumed that honest IoT devices have not been compromised; that is, the attackers are unable to access the private keys within honest IoT devices. An honest IoT device can process and properly follow the contractual routing protocol. For example, if an honest IoT device receives a smart contract with a zero-token bond, it will treat this as an invalid request.

(4) Attacker Exhaustion Defense Strategy: The defense strategy in the BCR protocol does not instantly halt an attack but, instead, it deters the attacker by gradually exhausting the attacker's tokens. Each honest IoT device has an internal mechanism which blacklists malicious IoT devices that interfere in the routing of previous data packets by preventing the packets from reaching a gateway. When an IoT device B is blacklisted by another IoT device A, A will prevent B from participating in the next smart contracts for a specified period.

(5) Sequential Punishment: If an attacker drops a data packet, every other intermediary IoT device on that route will be penalized by having to pay tokens to its previous intermediary IoT device. Each intermediary device will be paid in turn by the next intermediary device on the same route. This sequential punishment mechanism allows the routing protocol to punish the attacker which drops data packets.

(6) Transparency: All network nodes and attackers have access to the blockchain gateways and can acquire a copy of the blockchain data to learn about the smart contracts.

(7) Block producers: The blockchain is not compromised since it is assumed that the blockchain consensus algorithm works correctly. Thus, attackers cannot place a false transaction within a block in the blockchain through the block producers network.

The aim of the BCR protocol is to discourage attackers from interfering with packet routing, as such interference requires the expenditure of tokens. This mechanism permits different vendors' IoT devices to build trust in one another based on their past behaviors as they seek a route to a gateway, without the need for centralized certificated authority.

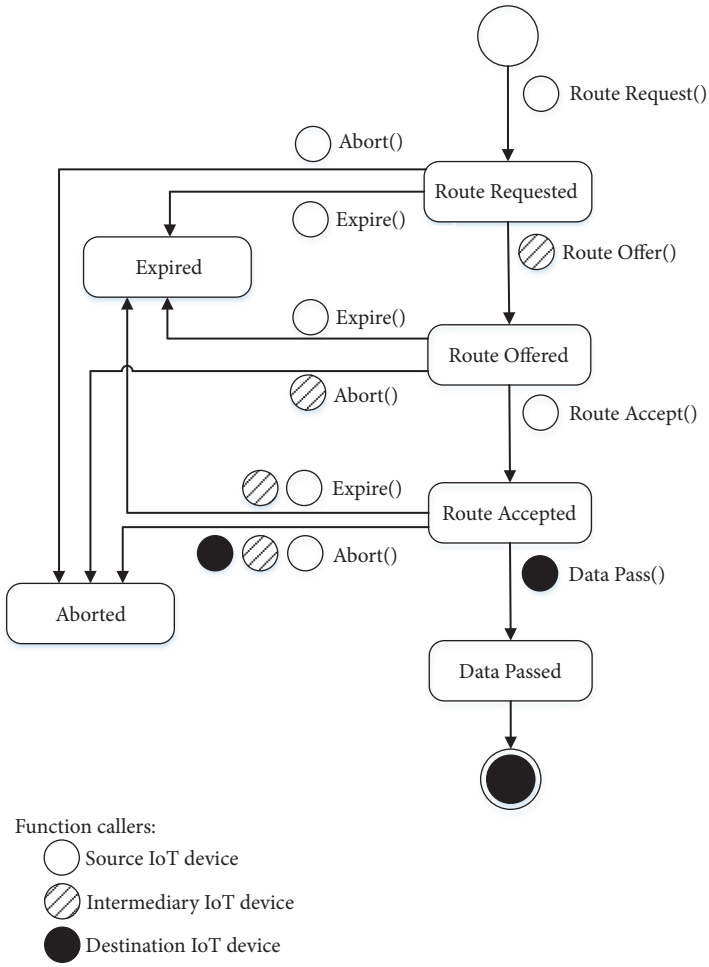


Figure 2: The protocol state machine of BCR protocol has 6 states. Transition between states occurs when IoT devices call functions inside smart contracts.

5. The Blockchain-Based Contractual Routing (BCR) Protocol

We first provide an overview of a general approach towards designing routing protocols. Existing routing protocols typically consist of two major phases. Phase 1 is for route establishment, while Phase 2 is for route maintenance. In Phase 1, a source IoT device sends a Route Request (RREQ) control message to find a route to a destination device. Each intermediary or destination device which receives the RREQ packet can respond by sending a Route Reply (RREP) message to the source IoT device. A Route Error (RERR) message is used to notify other devices that a certain device is no longer reachable, and they have to remove that route from their routing table.

In the proposed BCR protocol, each source IoT device creates a smart contract to request a route to a destination or data gateway for a specific period instead of creating RREQ control messages. Each smart contract created by an IoT device has a separate address within the blockchain that is generated by a block producer when placing a smart contract in a block. The IoT device can broadcast this address to its neighbors to inform them about a new routing request. The BCR protocol is implemented using smart contracts within the blockchain. The IoT devices request that the functions within the smart contract follow the BCR protocol. Thus, transmission of control messages in existing routing protocols is replaced by smart contract function calls in the BCR protocol. The BCR protocol is next explained in detail.

5.1. BCR Protocol States. Figure 2 shows the state machine diagram of the BCR protocol smart contract. The smart contract states are described below:

(i) **Route Requested:** When a source IoT device needs to reach a gateway, it creates a smart contract within the blockchain and sends the smart contract address to its neighbors. It also sets the state field within the smart contract to *Route Requested*. IoT devices do not necessarily need to know the data gateway address but can instead use an IPv6 address scheme, such as FF01::2, which allows devices to address any gateways or routers in the network [33]. The source IoT device transfers some of its own blockchain tokens as a bond to a smart contract address to create

a smart contract. The possibility of earning tokens encourages intermediary IoT devices to respond to the route request (*Route_Request_Bond*). The source IoT device also specifies the period for which the state of the route request within a smart contract is valid (*Route_Request_Expiry*). This smart contract is termed the original contract.

(ii) **Route Offered:** Each neighboring IoT device, which has a valid route entry to a gateway and would like to participate in relaying data packets (*Route_Offer_Validity*), can respond to an original smart contract. The intermediary IoT device offers its services to the source device by calling on a function within the original contract and transferring some of its own tokens to the smart contract address (*Route_Offer_Bond*). The function call goes to the block producers' network which changes the state of the received smart contract to *Route Offered*. A maximum of 3 route offers from different intermediary IoT devices can be stored in each contract. If the neighboring intermediary IoT device is unaware of a route to the data gateway or destination, it can still participate in relaying data packets by creating a new smart contract, namely, the intermediary contract. The intermediary contract stores the address of the originally issued smart contract or another intermediary contract in the *Parent_Contract* parameter.

(iii) **Route Accepted:** The source IoT device determines whether to accept an offered route to send its data packets. It selects the next neighbor to reach a gateway based on its own internal policies. It can choose a low-cost route offered by one of its neighbors or multiple neighbors to act as a relay(s) in order to increase the security and throughput of data packets.

(iv) **Route Passed:** When data is received by the data gateway, the smart contract state is changed to *Data Passed* by the gateway. If an IoT intermediary device B offers a route, but is unable to successfully relay the source IoT device's data packets within the specific time mentioned in the smart contract, the source IoT device will place the B's address to its internal blacklist for a limited period (*Blacklist_Timer*). The source IoT device will add its current blacklist addresses to any newly created smart contract's blacklist (*Blacklisted_Addresses*).

(v) **Aborted:** At any time, each device in the IoT network can abort the routing process by calling on the Abort function inside the smart contract. However, the smart contract Abort function acts accordingly based on its caller IoT device type and the current state of the smart contract.

(vi) **Expired:** As the BCR protocol has various timers, an IoT device can request that the Expire function inside a smart contract to review the timers and take action accordingly.

5.2. BCR Protocol Transitions. A protocol transition specifies the required conditions that triggers a state change. IoT devices perform the trigger when calling up a function inside the BCR protocol smart contract. We next review the parameters used by the functions inside the BCR protocol smart contract. Then, we explain the functions of the smart contract. The IoT devices call on these functions to run the BCR protocol.

(1) BCR protocol parameters: BCR protocol parameters within a smart contract are used by smart contract functions and can be seen publicly. The BCR protocol parameters within an IoT device are set by the IoT device based on its own internal policy. Each IoT device can have its own values for these internal parameters. The required parameters for a BCR protocol as listed in Table 1:

(i) *Contract_Address* stores the smart contract address. A smart contract can be dynamically created inside a blockchain by a source IoT device, or previously created by the IoT device owner. In the latter case, the IoT device owner, after creating a smart contract inside a blockchain, writes the address inside the IoT device.

(ii) *State* indicates the current state of a smart contract. Possible states are *Route Requested*, *Route Offered*, *Route Accepted*, *Data Passed*, *Expired*, and *Aborted* as explained in the previous section.

(iii) *Source, Intermediary,* and *Destination* store the addresses of the source, intermediary, and destination IoT devices. The source IoT device has requested access to a data gateway. The intermediary devices are ready to relay the data packets from the source IoT device to a destination or data gateway. This field in each smart contract stores up to three intermediary IoT device addresses. Destination IoT device is the destination node to be reached. In the Performance Evaluation section, we attempt to reach a data gateway network address as the destination, for example, FF01::2, that refers to any routers in an IPv6 network.

(iv) *Route_Request_Expiry (RRE)* is the expiry time until which the route request is valid.

(v) *Route_Request_Bond (RRB)* is set by the source IoT device and shows the number of tokens that the source IoT device will pay to the intermediary IoT device if the route to the destination works properly and the destination receives the data packets.

(vi) *Route_Offer_Validity (ROV)* shows the period for which the route offered by an intermediary IoT device to a source IoT device is valid. In other words, the intermediary IoT device relays the data packets to a gateway for the source IoT device only for a period which is specified by the ROV parameter.

(vii) *Route_Offer_Bond (ROB)* is the number of tokens an intermediary IoT device puts as a bond to guarantee that the intermediary IoT device can successfully pass the data packets to the gateway.

Table 1: BCR protocol parameters.

	Location	Parameter Name	Abbreviation
1	Inside smart contract	*Contract_Address*	
2	Inside smart contract	*State*	
3	Inside smart contract	*Source*	
4	Inside smart contract	*Intermediary*	
5	Inside smart contract	*Destination*	
6	Inside smart contract	*Route_Request_Expiry*	RRE
7	Inside smart contract	*Route_Request_Bond*	RRB
8	Inside smart contract	*Route_Offer_Validity*	ROV
9	Inside smart contract	*Route_Offer_Bond*	ROB
10	Inside smart contract	*Blacklisted_Addresses*	
11	Inside smart contract	*Selected_Route*	
12	Inside smart contract	*Timestamp*	
13	Inside smart contract	*Parent_Contract*	
14	Inside smart contract	*Hop*	
15	Inside smart contract	*Gas*	
16	Inside IoT device	*Blacklist_Timer*	
17	Inside IoT device	*Max_Hop*	

∗**Location** shows whether the parameter is used within a smart contract or an IoT device.

(viii) *Blacklisted_Addresses* stores a list of device addresses which are not allowed to participate in the smart contract for a certain period of time (*Blacklist_Timer*). This parameter is set by the source IoT device every time one of its neighbors fails in relaying data to a data gateway. Therefore, the intermediary addresses are restricted from putting forward any smart contract offer.

(ix) *Selected_Route* stores the intermediary address which is selected by the source IoT device for data packet forwarding. This address is selected from one of addresses in *Intermediary* parameter.

(x) *Timestamp* logs the time at which the smart contract is created in the blockchain. This field is set by block producers.

(xi) *Parent_Contract* stores the address of the previously issued smart contract. If the smart contract is an original one not preceded by a previously issued smart contract, the *Parent_Contract* parameter is empty. After receiving a smart contract, the IoT device checks this parameter to ensure that the previous smart contract was not self-issued. Using this mechanism, the routing protocol avoids a loop from occurring in the routing protocol.

(xii) *Hop* stores the number of hops from the source IoT device to the current intermediary IoT device. The intermediary device, after receiving a smart contract, checks its own routing table. If no route to a data gateway is found, it creates a new smart contract and sets this field in the newly created smart contract by increasing the *Contract_Hop* parameter value in the previous contract. Intermediary nodes use this parameter to prevent the creation of a routing loop if the parameter exceeds a *Max_Hop* or maximum value.

(xiii) *Gas* is a term used in the Ethereum blockchain to define the cost of calling on a function inside a smart contract via a source or intermediary IoT device. *Gas* shows the number of tokens that an IoT device should pay to the block producers when a smart contract's internal functions are run by the block producer.

(2) BCR protocol functions: The transition between smart contract states is performed by calling on the smart contract functions. Every time an IoT node calls on a function, some tokens as specified in the *Gas* of the function will be moved from the IoT device blockchain account to that of the block producer.

(i) **Route Request():** Each IoT device, whenever it needs to reach a destination or data gateway, can request that the blockchain producers create a smart contract on the blockchain. The source IoT device digitally signs a transaction for this purpose and sets the smart contract's parameters. This function is shown in Algorithm 1.

(ii) **Route Offer():** This takes place when an intermediary IoT device establishes a route to the destination or data gateway in its internal routing table and is ready to relay data packets to it for a source IoT device. Each contract accepts up to three route offers from intermediary devices. This function is shown in Algorithm 2.

(iii) **Route Accept():** Whenever a source IoT device decides to accept an offered route, it calls on the Route Accept function within the blockchain. The Block Producer runs this function if the function caller IoT

```
1: function ROUTE REQUEST(DESTINATION, RRB, RRE, BLACKLIST, PARENTADDRESS(OPTIONAL), HOP(OPTIONAL) )
2:     Transfer Gas tokens from the function caller to the block producer.
3:     Transfer RRB tokens from the function caller to the current contract address
4:     Set RRE to Route_Request_Expiry
5:     Set Blacklist to Blacklisted_Addresses
6:     if this is an original smart contract then
7:         Set Hop to 0
8:     end if
9:     if this is an intermediary smart contract then
10:        Set Hop to Hop
11:        Set Parent_Contract to ParentAddress
12:    end if
13:    Set Timestamp to Now
14: end function
```

Algorithm 1: Route Request function.

```
1: function ROUTE OFFER(ROB, ROV)
2:     Transfer Gas tokens from the function caller to the block producer.
3:     if the function caller address is not in Blacklisted_Addresses and the number of offers is less than three then
4:         Transfer ROB tokens from the function caller to the current contract address
5:         Set ROV to Rout_Offer_Validity
6:     end if
7: end function
```

Algorithm 2: Route Offer function.

```
1: function ROUTE ACCEPT(INTERMEDIARY)
2:     Transfer Gas tokens from the function caller to the block producer.
3:     if the function caller is Source then
4:         Move the intermediary to Selected_Route
5:         Transfer the ROB tokens of the other intermediary devices back
6:     end if
7: end function
```

Algorithm 3: Route Accept function.

device's address is identical to that of the source IoT device within the smart contract. This function is shown in Algorithm 3.

(iv) **Data Pass():** Whenever a destination IoT device receives data packets, it can call on the Data Pass function within the blockchain. The block producer runs the function if the function caller address is the same as the destination address within the smart contract. This function is shown in Algorithm 4.

(v) **Expire():** Whenever a destination IoT device receives the data packets, it can call on the Data Pass function inside the blockchain. The Block Producer runs the function if the function caller's IoT device's address is identical to that of the destination IoT device's address within the smart contract. This function is shown in Algorithm 5.

(vi) **Abort():** Whenever an IoT device wishes to leave the contract, it can call on the Abort function. Depending on the state of the contract, the Abort function returns the tokens to the IoT devices. This function is shown in Algorithm 6.

6. Performance Evaluation

We now study the performance of the BCR protocol in a network with no CA or node authentication support. We compare the performance of the BCR with that of the AODV routing protocol. We also assess the impact of Blackhole and Greyhole attacks on the BCR protocol.

6.1. Simulation Setup. We investigate the BCR protocol by developing a simulator using the Ethereum blockchain and Solidity language [29] to provide a proof of concept of the protocol. The average time between two consecutive blocks

```
1: function DATA PASS()
2:    Transfer Gas tokens from the function caller to the block producer
3:    if the function caller is Destination then
4:       Transfer the Rout_Request_Bond and Rout_Offer_Validity tokens to the Selected_Route
5:    end if
6: end function
```

Algorithm 4: Data Pass function.

```
1: function EXPIRE()
2:    Transfer Gas tokens from the function caller to the block producer.
3:    if state is Route Requested then
4:       if current time is more than Route_Request_Expiry then
5:          Transfer Route_Request_Bond tokens back to Source
6:          Transfer Route_Offer_Bond tokens back to Intermediary
7:       end if
8:    end if
9:    if state is Route Offered then
10:      if current time is more than Route_Offer_Validity then
11:         Transfer Route_Request_Bond tokens back to Source
12:         Transfer Route_Offer_Bond tokens back to Intermediary
13:      end if
14:   end if
15:   if state is Route Accepted then
16:      if the function caller is Intermediary or Destination then
17:         Transfer Route_Request_Bond and Route_Offer_Bond tokens to Source
18:      end if
19:      if the function caller is Source then
20:         Transfer Route_Request_Bond and Route_Offer_Bond tokens to Selected_Route
21:      end if
22:   end if
23: end function
```

Algorithm 5: Expire function.

in a blockchain is called block time. Since the Ethereum block time is 14 seconds, it may not be suitable for real time telecommunication applications as it is too long for interactive applications. In the EOS blockchain, the block time is much shorter, 0.5 sec, that makes it suitable for real implementation of the BCR protocol.

We study different scenarios for Greyhole and Blackhole attacks [34]. The source IoT device generates one Route Request smart contract for each 1000-byte data packet. The simulation parameters are summarized in Table 2.

The performance of the BCR protocol is evaluated based on the following metrics:

(i) *Packet Delivery Ratio (PDR)* is given by

$$PDR = \frac{D_{\text{rcv}}}{D_{\text{total}}}, \tag{1}$$

where D_{rcv} is the number of data packets successfully received by the gateway and D_{total} is the total number of data packets sent by the source IoT device.

(ii) *Throughput (TP)* is the average number of data packets successfully received per second by the gateway and is given by

$$TP = \frac{D_{\text{rcv}}}{T_{\text{sim}}}, \tag{2}$$

where T_{sim} is the simulation duration.

(iii) *Routing Overhead (RO)* is given by

$$RO = \frac{D_{\text{net}} + D_{\text{ctrl}}}{D_{\text{net}}}, \tag{3}$$

where D_{net} is the total number of passed data packets. We considered 1000 data packets for each smart contract. D_{ctrl} is the total number of control messages; that is, the number of function calls in smart contracts. Each function call in a smart contract is assumed to need a 100-byte control packet.

(iv) *Route Acquisition Latency (RAL)* is the average time between the generation of a smart contract and

```
1: function ABORT()
2:     Transfer Gas tokens from the function caller to the block producer.
3:     if state is Route Requested and the function caller is Source then
4:         Transfer Route_Request_Bond tokens back to the function caller
5:     end if
6:     if state is Route Offered then
7:         if the function caller is Source then
8:             Transfer Route_Request_Bond tokens back to the function caller
9:             Transfer Route_Offer_Bond tokens of all intermediary devices back to them
10:        end if
11:        if the function caller is Intermediary then
12:            Transfer Route_Offer_Bond tokens back to the function caller
13:        end if
14:    end if
15:    if state is Route Accepted then
16:        if the function caller is Intermediary or Destination then
17:            Transfer Route_Request_Bond of the Selected_Route and Route_Request_Bond tokens back to Source
18:        end if
19:        if the function caller is Source then
20:            Transfer Route_Request_Bond of the Selected_Route and Route_Request_Bond tokens back to Intermediary
21:        end if
22:    end if
23: end function
```

Algorithm 6: Abort function.

Table 2: Simulation parameter values.

	Device Type	Parameter Name	Value
1	S	Route_Request_Bond (RRB) (tokens)	100
2	S	Route_Request_Expiry (RRE) (sec)	800
3	I	Route_Offer_Bond (ROB) (tokens)	10
4	I	Route_Offer_Validity (ROV) (sec)	650
5	I	Route_Request_Bond (tokens)	(*prevRRB*)–10
6	I	Route_Request_Expiry (sec)	(*prevRRE*)–150
7	S,I	Blacklist_Timer (sec)	300
8	S,I	Max_Hop	5
9	-	simulation period (sec)	3600

∗ S: Source IoT Device.
∗ I: Intermediary IoT Device.
∗ (*prevRRB*): As shown in line 1, the value for RRB in an original smart contract is 100 tokens. When an intermediary IoT device receives an original smart contract, it sets the RRB value of its own intermediary smart contract to the value of RRB value of the received original smart contract minus 10, as 100 − 10 = 90. This trend continues for next intermediary smart contract; that is, if the previous contract is an intermediary smart contract, the RRB value would be (*previousRRB*) − 10.
∗ (*prevRRE*): As shown in line 2, the value for RRE in an original smart contract is 800 seconds. When an intermediary IoT device receives an original smart contract, it sets the RRE value of its own intermediary smart contract to the value of RRE value of the received original smart contract minus 150, as 800 − 150 = 650. This trend continues for next intermediary smart contract; that is, if the previous contract is an intermediary smart contract, the RRE value would be (*previousRRE*) − 150.

the reception of the first valid route offer from an intermediary device. This is calculated only for the contracts of data packets successfully received by the gateway:

$$RAL = \frac{\sum_{i \in S} \left(T_{i,\text{rep}} - T_{i,\text{req}} \right)}{|S|}, \tag{4}$$

where S is the set of successful smart contracts, $T_{i,\text{req}}$ is the time at which a contract is generated to request a route for data packet i, $T_{i,\text{rep}}$ is the time at which the first valid route offer for data packet i is received by the source IoT device, and $|S|$ is the size of set S.

We conduct the simulations/numerical experiments for a network topology with 14 devices, as shown in Figure 3. The source device has three possible paths to reach the data gateway (destination device). The devices 8, 3, and 4 are the first, second, and third malicious devices, respectively.

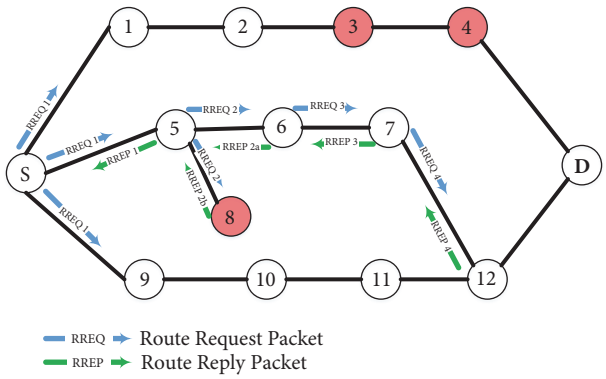


Figure 3: The route establishment process in BCR. The source and destination nodes are labeled S and D, respectively.

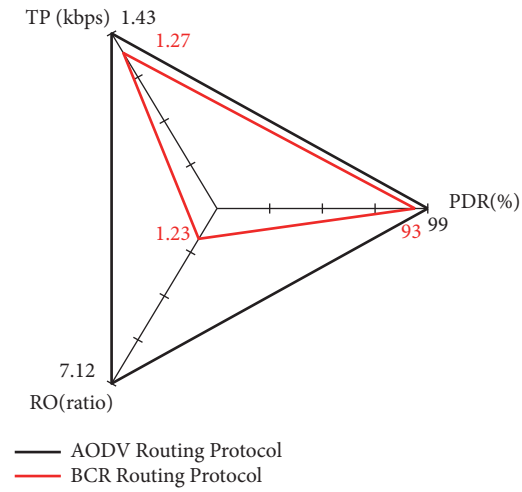


Figure 4: A comparison of the BCR and AODV routing protocols based on PDR, TP, and RO performance.

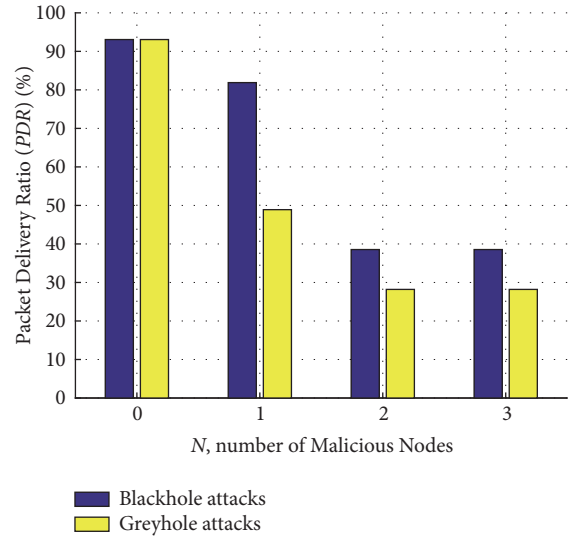


Figure 5: PDR of the BCR protocol in the absence of any attacks ($N = 0$) and in the presence of Blackhole and Greyhole attacks ($N \geqslant 1$).

The departure of data packets at the source device follows a Poisson process with an average packet interarrival time of 5 seconds.

6.2. Simulation Results. In this section, we compare the performance of the BCR protocol with that of the AODV routing protocol. We also assess the performance degradation of the BCR protocol in the presence of Blackhole and Greyhole attacks. In a Blackhole attack, the malicious device replies to the route request smart contracts by offering wrong routes in order to disturb the network. In Greyhole attacks, the malicious device passes or drops each data packet with probability 0.5. The malicious device aims to confuse its neighbors as to whether it is malicious or not.

(1) Comparison of BCR and AODV routing protocols: We evaluate the performance of AODV using ns-3 simulator. The ns-3 is an open source software providing a discrete-event network simulator for Internet research and educational use [35]. The ns-3 complies to the technical norms of standard organizations for emerging networks like 3GPP, IEEE, and Wi-Fi Alliance. This is the main reason we choose ns-3 as a prototyping tool for the performance analysis presented in this paper. We obtain the simulation results using the same data traffic and network topology as for BCR.

Figure 4 shows a comparison of the BCR and AODV routing protocols. The BCR protocol has a lower PDR (93%) than AODV (99%). The TP of the BCR protocol is 1.27 kbps which is 9% less than the AODV TP of 1.43 kbps. However, AODV incurs much higher RO ratio (7.12) than that of the proposed routing protocol (1.2). This is because, unlike AODV, our proposed routing protocol does not require IoT devices to start route establishment processes for sending each packet.

(2) Blackhole and Greyhole attacks: Figures 5–8 show the PDR, TP, RO, and RAL for BCR as a function of the number, N, of malicious nodes in the absence of attacks (i.e., $N = 0$) and in the presence of Blackhole and Greyhole attacks (i.e., $N \geqslant 1$).

Figure 5 shows the PDR of BCR for Blackhole and Greyhole attacks. It can be seen that the BCR protocol is less vulnerable to Blackhole attacks than to Greyhole attacks. This is due to the unpredictable behaviour of the Greyhole.

Figure 6 shows that the TP of BCR for different number of malicious nodes N. When $N = 3$, the TP decreases to almost one third of its value at $N = 0$. This is due to the presence of the malicious devices on two of the three possible paths from the source to the destination. This shows that BCR can complete the route establishment phase successfully without a CA.

Figure 7 shows the RO of BCR. The RO increases from 32% when there is no attack (i.e., $N = 0$) to 69% for Greyhole attacks with $N = 3$.

Figure 8 shows the RAL (in Block times) of BCR protocol. It can be seen that a route to the gateway is found in 5.5 Block times where there is no malicious device (i.e., $N = 0$). The RAL increases to 6.9 Block times when the network is under Greyhole attack by N=3 malicious nodes. The actual latency (in seconds) can be reduced by shortening the Block time

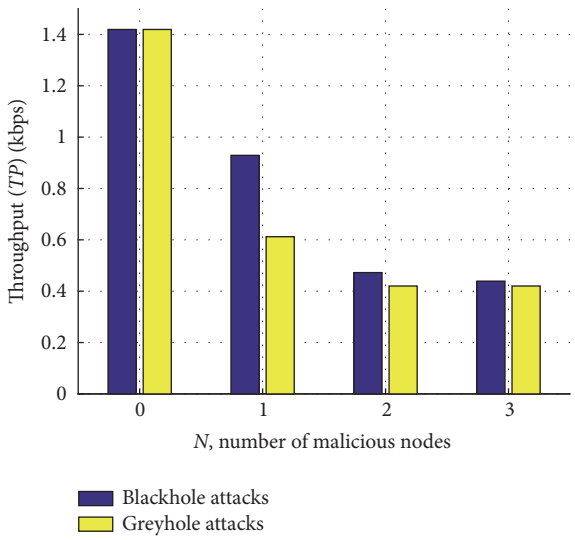


Figure 6: TP of the BCR protocol in the absence of any attacks ($N = 0$) and in the presence of Blackhole and Greyhole attacks ($N \geqslant 1$).

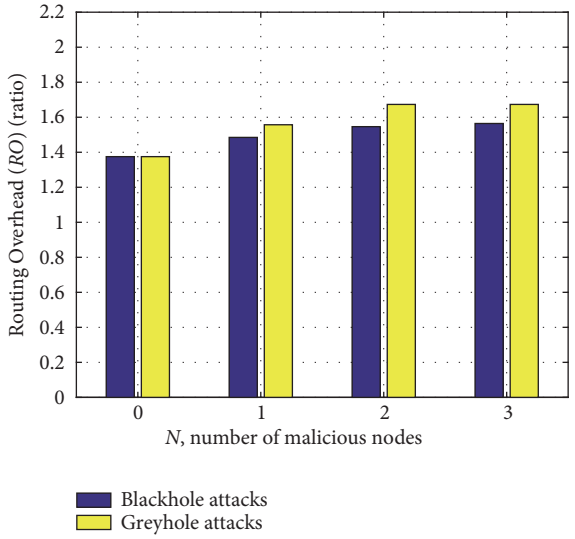


Figure 7: RO of the BCR protocol in the absence of any attacks ($N = 0$) and in the presence of Blackhole and Greyhole attacks ($N \geqslant 1$).

using other blockchain technologies such as EOS blockchain. With the Ethereum Block time of 14 seconds, the BCR protocol can be used only for delay-tolerant applications.

The EOS blockchain is a smart contract platform which is an alternative to the Ethereum blockchain. EOS uses a delegated proof of stake (DPoS) consensus algorithm in contrast to the energy-consuming PoW consensus mechanism used in Ethereum. Moreover, EOS can process 1,000-6,000 transactions per second while Ethereum can process only 15 transactions per second [29, 36]. These features make EOS more suitable for future development of the BCR protocol.

7. Conclusion

We have proposed a novel blockchain-based routing protocol for IoT networks, referred to as BCR, which can be

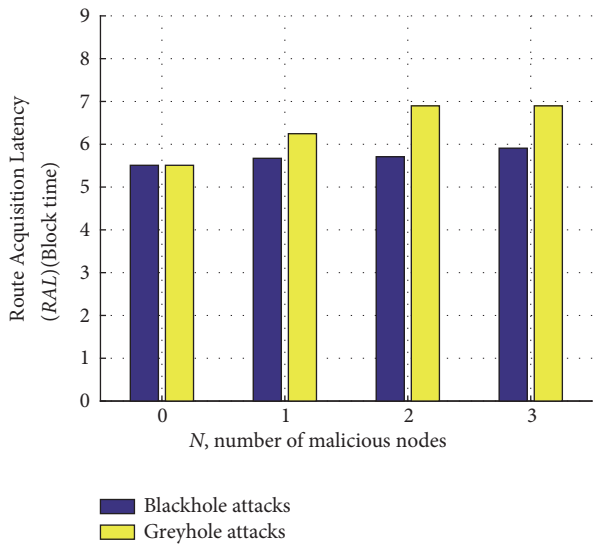


Figure 8: RAL of the BCR protocol in the absence of any attacks ($N = 0$) and in the presence of Blackhole and Greyhole attacks ($N \geqslant 1$).

enabled within a network of untrusted IoT devices. IoT devices can relay one another's data packets to gateways in a decentralized manner. The proposed BCR protocol does not require a central authority to authorize, add, or remove IoT devices, or a secret key sharing mechanism as required by traditional centralized routing protocols. We evaluated the performance of our proposed protocol compared to the AODV using extensive experiments. Our results show that the BCR reduces the routing overhead by a factor of 5 compared to the AODV. It is also resistant to Greyhole and Blackhole attacks. The proposed routing protocol can also be applied to *ad-hoc* networks.

Data Availability

The simulation data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada under Grant RGPIN 1731-2013 and by the UBC PMC-Sierra Professorship in Networking and Communications.

References

[1] G. Glissa, A. Rachedi, and A. Meddeb, "A secure routing protocol based on RPL for internet of things," in *Proceedings of the 59th IEEE Global Communications Conference, GLOBECOM 2016*, pp. 1–7, USA, December 2016.

[2] M. Bouaziz and A. Rachedi, "A survey on mobility management protocols in Wireless Sensor Networks based on 6LoWPAN technology," *Computer Communications*, vol. 74, pp. 3–15, 2016.

[3] "IOTA Whitepaper," 2018, http://iotatoken.com/IOTA_Whitepaper.pdf=0pt.

[4] D. Airehrour, J. Gutierrez, and S. K. Ray, "Secure routing for internet of things: A survey," *Journal of Network and Computer Applications*, vol. 66, pp. 198–213, 2016.

[5] J. W. Hui, "RFC 6553 - The Routing Protocol for Low-Power and Lossy Networks (RPL) Option for Carrying RPL Information in Data-Plane Datagrams," https://tools.ietf.org/html/rfc6553=0pt.

[6] C. Deepa and B. Latha, "HHSRP: A cluster based hybrid hierarchical secure routing protocol for wireless sensor networks," *Cluster Computing*, pp. 1–17, 2017.

[7] P. L. R. Chze and K. S. Leong, "A secure multi-hop routing for IoT communication," in *Proceedings of the 2014 IEEE World Forum on Internet of Things, WF-IoT 2014*, pp. 428–432, Republic of Korea, March 2014.

[8] J. Duan, D. Yang, H. Zhu, S. Zhang, and J. Zhao, "TSRF: A Trust-aware Secure Routing Framework in Wireless Sensor Networks," *International Journal of Distributed Sensor Networks*, vol. 10, no. 1, pp. 1–14, 2014.

[9] X. Anita, J. Martin Leo Manickam, and M. A. Bhagyaveni, "Two-way acknowledgment-based trust framework for wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 9, no. 5, Article ID 952905, pp. 1–14, 2013.

[10] "A Foundation for Improved Protection and Automation," 2018, https://www.cisco.com/c/dam/en_us/solutions/industries/energy/downloads/bc-hydro-cisco.pdf.

[11] N. R. Yerneni and A. K. Sarje, "Secure AODV protocol to mitigate Black hole attack in Mobile Ad hoc," in *Proceedings of the 2012 3rd International Conference on Computing, Communication and Networking Technologies, ICCCNT 2012*, pp. 1–5, India, July 2012.

[12] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Ariadne: A secure on-demand routing protocol for ad hoc networks," *Wireless Networks*, vol. 11, no. 1-2, pp. 21–38, 2005.

[13] J. Zhou and J. Cao, "OSR: Optimal and secure routing protocol in multi-hop wireless networks," in *Proceedings of the 32nd IEEE International Conference on Distributed Computing Systems Workshops, ICDCSW 2012*, pp. 187–193, China, June 2012.

[14] A. Jain and B. Buksh, "Solutions for secure routing in mobile ad hoc network (MANET): A survey," *Imperial Journal of Interdisciplinary Research*, vol. 2, no. 4, pp. 5–8, 2016.

[15] M. Kassim, R. A. Rahman, and R. Mustapha, "Mobile ad hoc network (MANET) routing protocols comparison for wireless sensor network," in *Proceedings of the 2011 IEEE International Conference on System Engineering and Technology, ICSET 2011*, pp. 148–152, Malaysia, June 2011.

[16] S. Boora, Y. Kumar, and B. Kochar, "A survey on security issues in mobile ad-hoc networks," *IJCSMS International Journal of Computer Science and Management Studies*, pp. 129–137, 2011.

[17] L. Anderegg and S. Eidenbenz, "Ad hoc-VCG: a truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents," in *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom '03)*, pp. 245–259, ACM Press, New York, NY, USA, September 2003.

[18] S. Zhong, J. Chen, and Y. R. Yang, "Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks," in *Proceedings of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Society (INFOCOM 2003)*, vol. 3, pp. 1987–1997, San Francisco, CA, USA, 2003.

[19] Y.-C. Hu and A. Perrig, "A survey of secure wireless ad hoc routing," *IEEE Security & Privacy*, vol. 2, no. 3, pp. 28–39, 2004.

[20] C. Perkins, E. Belding-Royer, S. Das et al., "RFC 3561- Ad hoc On-demand Distance Vector (AODV) Routing," *Internet RFCs*, pp. 1–38, 2003.

[21] Q. He, D. Wu, and P. Khosla, "SORI: A secure and objective reputation-based incentive scheme for ad-hoc networks," in *Proceedings of the 2004 IEEE Wireless Communications and Networking Conference, WCNC 2004*, pp. 825–830, USA, March 2004.

[22] B. David, R. Dowsley, and M. Larangeira, "MARS: Monetized Ad-hoc Routing System (A Position Paper)," in *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems*, pp. 82–86, Munich, Germany, June 2018.

[23] H.-Y. Huang and M. Bashir, "The onion router: Understanding a privacy enhancing technology community," in *Proceedings of the 79th ASIS&T Annual Meeting: Creating Knowledge, Enhancing Lives through Information & Technology*, p. 34, 2016.

[24] A. Biryukov and I. Pustogarov, "Proof-of-work as anonymous micropayment: Rewarding a Tor relay," in *Proceedings of the International Conference on Financial Cryptography and Data Security*, vol. 8975 of *Lecture Notes in Computer Science*, pp. 445–455, Springer, Heidelberg, 2015.

[25] R. Ananthapadmanabha, B. S. Manoj, and C. Siva Ram Murthy, "Multi-hop cellular networks: The architecture and routing protocols," in *Proceedings of the 12th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2001)*, vol. 2, USA, 2001.

[26] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in *Proceedings of the Annual International Cryptology Conference (CRYPTO'17)*, vol. 10401 of *Lecture Notes in Computer Science*, pp. 357–388, Springer, Cham, 2017.

[27] J. Garay, A. Kiayias, and N. Leonardos, "The bitcoin backbone protocol: analysis and applications," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 281–310, 2015.

[28] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *Proceedings of the International Conference on Financial Cryptography and Data Security*, vol. 8437 of *Lecture Notes in Computer Science*, pp. 436–454, Springer Berlin Heidelberg, 2014.

[29] V. Buterin, *Ethereum: A Next-generation Smart Contract and Decentralized Application Platform*, 2014, https://github.com/ethereum/wiki/wiki/%5BEnglish%5D-White-Paper.

[30] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts," in *Proceedings of the 2016 IEEE Symposium on Security and Privacy, SP 2016*, pp. 839–858, USA, May 2016.

[31] E. Androulaki, A. Barger, V. Bortnikov et al., "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proceedings of the the Thirteenth EuroSys Conference (EuroSys '18)*, pp. 1–15, Porto, Portugal, April 2018.

[32] N. Szabo, "Formalizing and securing relationships on public networks," *First Monday*, vol. 2, no. 9, 1997, http://ojphi.org/ojs/index.php/fm/article/view/548=0pt.

[33] R. Hinden and S. Deering, "RFC 4291 - IP Version 6 Addressing Architecture," pp. 13–15, 2006, https://tools.ietf.org/html/rfc4291=0pt.

[34] R. Kaur and P. Singh, "Black hole and greyhole attack in wireless mesh network," *American Journal of Engineering Research (AJER)*, vol. 3, no. 10, pp. 41–47, 2014.

[35] NS-3 Project, "NS-3 - Network Simulator - Tutorial - Release 3.29," https://www.nsnam.org/docs/release/3.29/tutorial/ns-3-tutorial.pdf.

[36] "EOS.IO Technical White Paper v2," 2018, https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md.

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 3029638, 18 pages
https://doi.org/10.1155/2018/3029638

Research Article

Shielding IoT against Cyber-Attacks: An Event-Based Approach Using SIEM

Daniel Díaz López,[1] **María Blanco Uribe**,[1] **Claudia Santiago Cely**,[1] **Andrés Vega Torres**,[1] **Nicolás Moreno Guataquira**,[1] **Stefany Morón Castro**,[1] **Pantaleone Nespoli**,[2] **and Félix Gómez Mármol**[2]

[1]*Computer Science Faculty, Colombian School of Engineering Julio Garavito, Colombia*
[2]*Department of Information and Communications Engineering, University of Murcia, Spain*

Correspondence should be addressed to Daniel Díaz López; daniel.diaz@escuelaing.edu.co

Received 28 May 2018; Revised 10 September 2018; Accepted 4 October 2018; Published 25 October 2018

Guest Editor: Constantinos Kolias

Due to the growth of IoT (Internet of Things) devices in different industries and markets in recent years and considering the currently insufficient protection for these devices, a security solution safeguarding IoT architectures are highly desirable. An interesting perspective for the development of security solutions is the use of an event management approach, knowing that an event may become an incident when an information asset is affected under certain circumstances. The paper at hand proposes a security solution based on the management of security events within IoT scenarios in order to accurately identify suspicious activities. To this end, different vulnerabilities found in IoT devices are described, as well as unique features that make these devices an appealing target for attacks. Finally, three IoT attack scenarios are presented, describing exploited vulnerabilities, security events generated by the attack, and accurate responses that could be launched to help decreasing the impact of the attack on IoT devices. Our analysis demonstrates that the proposed approach is suitable for protecting the IoT ecosystem, giving an adequate protection level to the IoT devices.

1. Introduction

The diversity of IoT devices has grown rapidly and attracted the attention of both the industry and academic society [1]. The Internet of Things (IoT) is considered as an emerging technology with considerable potential of development during this decade [2]. IoT is considered as a part of the Internet of the future, where billions of intelligent "things" will communicate to provide services to humans as an ultimate goal [3].

In [4], authors define the IoT as "*a world where physical objects are seamlessly integrated into the information network, and where the physical objects can become active participants in business processes. Services are available to interact with these 'smart object' over the Internet, query their state and any information associated with them, taking into account security and privacy issues*". Besides, IoT solutions are composed of a complex network of intelligent devices, sensors, and Internet connectivity, through which data can be collected, exchanged, and processed. These devices may be located in vehicles, buildings [5], home appliances [6], or cell phones, and they are controlled by software that allows them to be managed [7]. Under the IoT paradigm, some crucial application domains will be enhanced, such as healthcare and environmental and industrial plant monitoring [3]. To better perceive the IoT potential benefit, in 2016, Gartner estimated 6 billion things connected, forecasting additionally 21 billion smart objects in 2020 [2]. Moreover, Cisco Systems predicted that the IoT would create $14.4 trillion as a result of the combination of increased revenues and lower costs for companies from 2013 to 2022 [8].

IoT devices are equipped with sensors that capture data which must be transmitted to other places for processing and the IEEE and the IETF have defined a communication protocols stack for IoT designed to guarantee interoperability

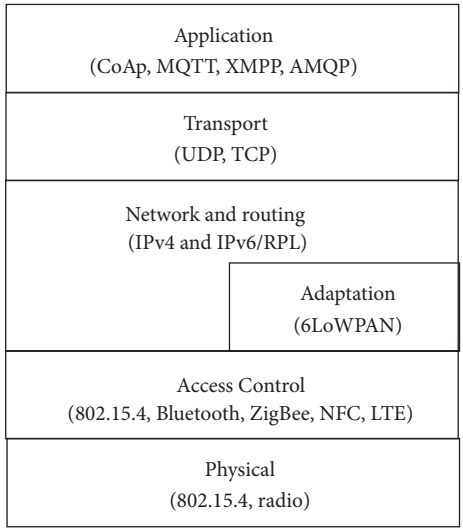


Figure 1: Communication protocols stack for IoT.

with other existing Internet devices [9]. The stack, illustrated in Figure 1, is composed of different interoperable protocols: (i) 802.15.4 (Low-Rate Wireless Personal Area Networks, LR-WPANs(low data rate solution with multimonth to multiyear battery life and very low complexity. It is operating in an unlicensed, international frequency band. Potential applications are sensors, interactive toys, smart badges, remote controls, and home automation)) and radio for the physical layer, (ii) Bluetooth, WiFi, NFC, 802.15.4, LTE, and Zigbee for the access control layer (MAC), (iii) IPv4 and 6LoWPAN (IPv6 Over Low-Power Wireless Personal Area Networks) [10] as a connector to the network layer (Adaptation) in 802.15.4 networks, (iv) RPL (IPv6 Routing Protocol for Low-Power and Lossy Networks) as a routing protocol, (v) UDP or TCP in transport layer, and (vi) CoAP (Constrained Application Protocol), MQTT (Message Queue Telemetry Transport), XMPP (Extensible Messaging and Presence Protocol), or AMQP (Advanced Message Queuing Protocol) for the application layer.

And to study the IoT solutions, an IoT architecture based on 5 components is proposed in [11]: (i) **perception or device** consisting of physical objects and sensor devices and their capacity to capture information, (ii) **network or transmission** responsible for transferring information from the device layer to the information processing system, (iii) **middleware** receiving the information from the network layer and storing it in a database, (iv) **application** consisting in the applications using the information gathered for the IoT devices, like smart health or home automation, and (v) **business** responsible for the management of the overall IoT system, including applications and services.

However, although IoT solutions are being implemented in a growing and fast way in almost all the aspects of humans' everyday life, their security exposition is not decreasing [12]. The integration of real-world objects with the Internet poses several security challenges, so that currently researchers and developers worldwide are struggling to find innovative solutions to address them [13, 14]. Specifically, the nature of the IoT ecosystem itself presents limitations from a security viewpoint. Due to the different standards and communication stacks involved, the resources constraints, and the massive amount and heterogeneity of interconnected devices, traditional security measures may not be employed efficiently in IoT systems [15]. Consequently, the process of reviewing different security alternatives for IoT devices becomes of crucial importance, so that one can spot the main challenges inherent to the IoT trend and possibly propose solutions [16].

Additionally, the large amount of data exchanged among the smart objects makes it unfeasible for a human to appropriately deal with all the events generated. In this direction, some proposals have been presented based on the management of security events generated by IoT devices. Concretely, these proposals suggest the use of a Security Information and Event Management (SIEM) system to centrally acquire knowledge stemming from IoT sources and, by doing so, to contribute with the development of IoT security [17, 18]. Although these proposals show promising features, none of them considers the possible correlations between IoT layers, security events, and attack surfaces, focusing mainly on suggesting methodologies to manage IoT security events coming into the SIEM. Furthermore, the reaction phase to the security incidents is commonly neglected, despite its vital importance due to the strong integration of the smart things into the real world. Thus, the proposed correlation is supportive for the security administrators, who can benefit from our studies to build strong correlation rules and to plan more robust reaction strategies against malicious threats.

The main contributions of this paper can be summarized as follows:

(i) Proposal of a security architecture for IoT adopting event management to deal with security incidents.

(ii) Detailed review of 11 IoT security event categories (defined by OWASP (https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project)) and proposal of a mapping between events, vulnerabilities, and attack surfaces in an IoT ecosystem.

(iii) Application of the proposed architecture to three common IoT scenarios, showing its feasibility through the generation of alerts and correlation rules.

(iv) Extension of the proposed solution through a defensive and offensive approach defining different incident responses.

The remainder of the paper is organized as follows: Section 2 describes the security problems for IoT devices, including the particularities of security event management. Section 3 exposes different SIEM solutions from a commercial and academic perspective. Following, Section 4 presents the details about the proposal of the security architecture using security events as a basis to protect IoT ecosystems. In Section 5 different alerts and correlation rules are implemented for the processing of security events of IoT devices allowing detecting and predicting security incidents. Different responses to security incidents from a defensive

security strategy are considered in Section 6. Last but not least, Section 7 presents relevant conclusions and future works.

2. Problem Statement

2.1. IoT Attack Surfaces. An *attack surface* comprises the enablers (i.e., communication channels and protocols) and the targets (i.e., processes and data) that are required to perform an attack [19]. The attack surface generally depends on the interconnection between components of a system and the authorization definitions; i.e., privileges represented as access control policies. In an IoT architecture composed of a number of building blocks, it is reasonable to think about different attack surfaces. An attack surface also clusters all the different points that an attacker could use to get into a system and to steal/leak data out [20]. In this way, behind an attack surface there is a set of vulnerabilities that are the specific functions and/or elements of an IoT component that need to be reviewed from a security perspective.

Once an attack surface has been identified, it is possible to enumerate the security vulnerabilities and high-risk areas requiring defense-in-depth protection. Such protection refers to a commonly used strategy to shield critical resources on enterprise networks where security controls are established at multiple levels of the IT Information Technology infrastructure. In this context, the nine most relevant attack surfaces identified by OWASP in their Internet of Things Project (https://www.owasp.org/index.php/Attack_Surface_Analysis_Cheat_Sheet) for IoT ecosystems are listed as follows:

(1) Administrative interface: it consists of all the attack vectors (path or means by which an attacker can gain access to a computer or network server in order to deliver a malicious outcome [21]) related to the system's administrative web interface.

(2) Device web interface: it is composed of all the web application vulnerabilities of the IoT device web interface and credential management.

(3) Cloud web interface: the vulnerabilities involved in this attack surface are the standard web application vulnerabilities, credential management, transport encryption, and lack of two-factor authentication, applied to IoT cloud components like applications and web services.

(4) Mobile application: an attacker could use the vulnerabilities associated with a mobile application connected to an IoT device as attack vectors. Vulnerabilities can include username enumeration, weak passwords, lack of encryption and account lockout, among others.

(5) Device network services: insecure network services could allow Denial of Service (DoS) attacks, injections, Man in the Middle attacks, buffer overflow, and others.

(6) Update mechanism: lack of encryption or signature, writable locations, or missing update mechanisms could be used by an attacker to get into a system or introduce malicious scripts.

(7) Device physical interfaces: insecure elements involved in this attack surface could be used to tamper IoT devices, reset, get more privileges, extract device's firmware, or remove media storage. It also refers to the JTAG (Joint Test Action Group)/(Serial Wire Debug) interfaces that are used in chips for testing and debugging embedded applications. An unauthorized physical connection to these interfaces could introduce data injection, operation malfunction and also reverse engineering actions.

(8) Device firmware: firmware can expose sensitive data like credentials, encryption keys, authentication keys, firmware version, installed services, APIs, etc. This information could be used by the attacker to pivot toward another valuable service information. This attack surface includes vulnerabilities like backdoor accounts, firmware version display, vulnerable services, sensitive data exposure related to keys and accounts, among others.

(9) Local data storage: storage can be an attack surface when data is not encrypted, when it is encrypted using a compromised key, or when there are no integrity checks.

It is clear that the IoT ecosystem is exposed to security threats from different perspectives. The number of possible attack surfaces an attacker could leverage to perform their malicious activities represents unquestionably a motivation to build strong defense strategies. In addition, due to the resource-constraint nature of the IoT nodes, the traditional defensive schemes cannot be directly enforced, exposing the entire network to a higher risk. A clear example of such threats is the *Mirai botnet* that, together with its variants, was able to take control of the IoT devices and perform an incredibly powerful DDoS [22].

2.2. IoT Vulnerabilities. Information security in IoT devices, and in general in IT infrastructure, can be understood through three key concepts: confidentiality, integrity, and availability. In the context of IoT, availability would refer to the ability to use the information managed by the IoT device whenever it is required. Confidentiality would refer to the access to the information generated, stored, and processed by the IoT device allowed only to authorize users or entities. Integrity would consist of avoiding IoT data to be modified by unauthorized persons or entities. The above-mentioned security principles are better known as *CIA attributes*, and they represent a measure of the security level of the underlying system. Unfortunately, these attributes can be violated by exploiting vulnerabilities existing in the IoT devices. Therefore, the vulnerabilities are flaws in an element of a system that increase the attack surface. Specifically, an attacker may exploit the HW/SW breaches of the IoT system aiming at performing their malicious activities. Assuming that the detection of such violations is reliable, a security event is subsequently generated by the security monitors within the system.

Following the OWASP IoT project, the 16 most harmful IoT vulnerabilities grouped by security aspects are the following (https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project):

(i) Authentication:

(a) lack of controls to avoid username enumeration: an attacker could collect valid usernames through authentication mechanisms.
(b) use of weak passwords: this refers to passwords used in some authentication process against a component of the IoT service, which would be easy to guess because of its low complexity.
(c) lack of an account lockout control after multiple failed attempts: it would allow an attacker to attempt authenticating many times without being blocked.
(d) lack of two-factor authentication for critical functions: it claims that the authentication processes involved in the IoT service do not consider multiauthentication to access critical functions.

(ii) Cryptography

(a) unencrypted network services allowing eavesdropping: communication between components of the IoT services are not protected allowing possible tampering.
(b) failing in the implementation of encryption mechanisms: encryption methods are poorly implemented, improperly configured, or not being properly implemented.

(iii) Updating mechanism

(a) update sent without encryption: updates at different architecture levels are transmitted over the network without using TLS or encrypting the files involved in the process.
(b) remote update is done without security controls: there is no authentication method or secure remote control to regulate the remote device update.
(c) the storage location for updates files is writable: storage location for update files has write and read permissions to any user, allowing firmware modification or distribution.

(iv) Physical access

(a) firmware and data could be extracted allowing access to sensitive information: firmware could be extracted or gives more information than it should.
(b) possible access to the device console because of lack of controls: it is possible to obtain full console access through serial interfaces, or lack of controls to avoid entering single user mode.
(c) storage media is physically unprotected and could be removed.
(d) lack of controls to avoid physical connection to the device causing manipulation of the code execution flow: it is possible to modify the execution code in order to bypass security controls or access to sensitive information.

(v) Device control

(a) lack of controls against Denial of Service: it is possible to deny service through network or the device itself.
(b) lack of controls to avoid command injection: there are no countermeasures to avoid injections, e.g., command or SQL, which can affect the data managed by the IoT service.
(c) IoT service contains insecure third party components: applications or components developed by third-parties are out of date or they have security weaknesses.

One could easily argue that the task of preventing and reacting against the previously listed vulnerabilities is not trivial. In the IoT context, there is a particular need to support the system administrators, who bear the burden of controlling hundreds different events stemming from different sources.

In conclusion, the security analysis of the IoT ecosystem is complex, since it is exposed to different types of attacks, employing different attack surfaces and exploiting a variety of vulnerabilities present in IoT infrastructures. Nonetheless, the IoT environments generate an enormous amount of different events that should be leveraged to increase the security level of the entire system.

3. State of the Art

As previously stated, IoT ecosystems attracted the attention of both the academia and the information industry which are struggling to address a number of security challenges. On the one hand, IoT devices generate, process, and exchange vast amounts of security and safety-critical events, as well as privacy-sensitive information, making them appealing targets for ill-motivated entities [23]. And on the other hand, these devices can be involved as part of an attack, where vulnerable IoT devices are infected and become part of a botnet in order to subsequently reach further devices and compromise them [24]. Moreover, the attack surfaces that can affect traditional computer systems are extended in the case of IoT environments, adding some peculiarities due to their inherent characteristics [25]. For the ease of understanding, in this section we firstly introduce the concept of Security Information and Event Management (SIEM), and then we thoroughly review the related works of the field, highlighting their pros and cons.

3.1. Security Information and Event Management (SIEM). In order to understand what security event management means,

it is necessary to define the concept of an information security event. Based on ISO 27000 [26], an *information security event* identifies the occurrence of a system, service or network state indicating a possible breach of information security policy or failure of controls or a previously unknown situation that may be security relevant. The security events can come from virtual private networks, firewalls, intrusion detection systems, intrusion prevention systems, routers, hosts, switches, and servers, among others [27]. Then, a security event management (SEM) is a system that allows real-time monitoring, reporting, normalization, correlation, and aggregation of security events. SEM combines with Security Information Management (SIM) allowing log management and reporting to conform a Security Information and Event Management (SIEM) [28]. Thus, SIEM supports threat detection and security incident response through the real-time collection and historical correlation and analysis of security events from a wide variety of events and contextual data sources across an IT infrastructure. It also supports compliance reporting and incident investigation through analysis of historical data from these sources (https://www.gartner.com/it-glossary/security-information-and-event-management-siem). A SIEM provides long-term analysis of security events and real-time reporting.

3.2. Related Works. Until now, some authors tried to tackle the IoT security weak points by investigating in hardware solutions [29, 30], while others suggested to solve the problem pinpointing to higher IoT layers, proposing innovative authentication methods included in MQTT, NFC [31], and smart card [32]. Above all, traditional defense mechanisms such as intrusion detection systems (IDSs) have been analyzed to further protect IoT devices. However, applying traditional IDS techniques is difficult due to the particular characteristics of IoT systems, such as constrained-resource devices, specific protocol stacks, and communication standards [33] as well as the network traffic generated [34]. Despite the motivations, the mentioned proposals do not consider the huge number of connected IoT devices in a real scenario with exception of works such as [34], where authors propose a mechanism to filter large volumes of traffic before performing analysis using a Bayesian inference method through an IDS. Consequently, the analysis of the generated volume of events is undervalued, as well as their correlation. To this extent, one could argue that the presence of a system which can centrally collect and analyze IoT events is crucial in a full-fledged IoT system, guiding the security experts to undertake optimal decisions.

Following this reasoning, some proposals have been presented among the research community to handle the above-mentioned drawbacks. From a design and functional perspective, [35] recommends the use of IoT events to determine the correct operational conditions; that is, the endpoints must be enabled with a warning threshold that helps to determine the state of a service. Reference [35] also suggests that the endpoints should log these events in a persistent memory to ensure that administrators can perform forensic analysis at a later stage. As the events are related to operational phases (e.g., power, temperature), the information included can be leveraged to detect possible malicious activities.

In addition, authors in [17] propose a system to analyze IoT security events as a "self-similar" system within the normal operation of its objects. Starting from the assumption that classical SIEM systems are incompatible to manage the vast amount of events generated by the IoT devices, authors suggest the study of an IoT architecture as a native network device. This network is represented as a graph with nodes (i.e., devices) and links (i.e., interconnection between devices) and the security analysis consists of the measurement of graph anomalies using fractal geometry.

Furthermore, in [18] authors propose integrating SIEM systems within IoT ecosystems using data aggregation, data digital signing, and swarm routing algorithm. Applying their methodology, they claim to tackle some problems related to this integration, such as data volume reduction, data integrity, and data delivery warranty for the messages exchanged between the devices and the SIEM. In particular, the last feature is enforced using a swarm algorithm, so that malicious network nodes are revised.

Another attempt of integration of SIEM technology into the IoT world is proposed in [36]. The authors present an IDS framework for IoT empowered by 6LoWPAN devices, which in turn sends the reported security events to an open-source SIEM (i.e., prelude (https://www.prelude-siem.com/en)). The feasibility of the proposed architecture is then demonstrated by conducting preliminary tests, where the performance of the implemented IDS framework is evaluated via penetration testing. Similarly, authors in [37] proposed a wireless IDS architecture applicable to a variety of IoT scenarios. Extending the work presented in [38], the authors suggest its application in the context of eHealth intrusion detection. In this case, the security events generated by open-source IDSs (i.e., Kismet (https://www.kismetwireless.net/) and Snort (https://www.snort.org/)) are sent to another open-source SIEM (i.e., OSSIM (https://www.alienvault.com/products/ossim)).

An approximation of SIEM suitable for IoT is presented in AMSEC [39, 40]. AMSEC is a proposal extending a traditional SIEM by adding interesting features to make it applicable to IoT scenarios. Specifically, authors propose a framework to model attack scenarios, which if deployed in tandem with a SIEM system is capable of (i) generating Attack Trees and Service Dependency Graphs based on the topological vulnerability analysis, (ii) applying algorithms to provide a near real-time attack modeling, (iii) analyzing attack models to predict future attacker's steps, (iv) calculating security metrics, and (v) selecting the optimal security solution through an interactive decision support process.

Attacks over IoT devices have become a reality, as described in [41], where a complete taxonomy of Mirai botnet is presented. Mirai attacked different Internet targets through the control of nearly half a million IoT devices. The vulnerabilities exploited in this case over the IoT devices were "Use of weak password" and "lack of controls to avoid username enumeration". In the same direction, [42] presents a threat analysis for smart home scenarios, which focuses on flaws caused by interactions among devices. Different kinds

of IoT threats (mainly eavesdropping, DoS, impersonation, and software exploitation) are resumed and analyzed. Additionally, an analysis of how the attacks can be performed is developed for each threat.

All the above-mentioned solutions share a common background; that is, they propose the progressive integration of the SIEM technologies into the IoT systems to manage the events stemming from the IoT devices. Nevertheless, none of them focuses on the identification of proper IoT events and/or IoT event categories. Moreover, the existent connections between the IoT security events, vulnerabilities and attack surfaces is neglected. One could safely argue that a deep study in this direction constitutes an appealing research goal, since system administrators can leverage these connections to create stronger correlation rules and eventually plan strategic counteractions against suspicious alerts. Our work focuses on these challenges by highlighting the existing multirelations between symptoms (i.e., IoT events), causes (i.e., related vulnerabilities being exploited), and attack vectors (i.e., attack surfaces). The proposed multirelations can be used to identify the potential security risks to which IoT ecosystems are exposed and to make the correlation processes more efficient within the SIEM.

4. IoT Security Events Categories Analysis

The IoT events can be depicted as a representation or a symptom of something happening within the system which under certain circumstances may represent a security incident. Security events represent a comprehensive set of data which can be used in cyber security processes. However, it can be tremendously difficult to understand and use those events if there is not a clear understanding about their causes. For example, the causes can be related to the exploitation of some vulnerabilities on some components of the ecosystem. Additionally, since IoT ecosystems usually have a wide attack surface due to all the involved components, it is important to identify which particular attack vector or attack surface is directly related to that symptom and cause. Understanding the mapping between symptoms (i.e., IoT events), causes (i.e., vulnerabilities), and attack vectors is fundamental to develop different security tasks such as prevention, detection, and reaction against an IoT attack.

In this section, we present a security solution which uses a SIEM to process security events and protect IoT ecosystems. Figure 2 illustrates how the proposed solution is integrated in an IoT ecosystem. These security events are specially processed in the SIEM based on previously defined multirelations between security event categories, attack surfaces, and vulnerabilities.

The proposed multirelations help to contextualize the event, as it allows determining the vulnerabilities that could have been exploited and the related attack surfaces inside the IoT ecosystem. Reviewing the events bearing in mind the context in which they are generated is fundamental to achieve a security incident management process that can find the root cause of incidents and define proper remediation.

A schematic view of the proposed mapping between these concepts can be seen in Table 1, which is the result of the analysis developed around 11 security event categories (as we will see next) and can be utilized as a basis to understand the risks that IoT ecosystems face. Thus, an event belonging to an event category is generated upon the exploitation of one or more vulnerabilities present in one or more attack surfaces of an IoT ecosystem. Likewise, attack surfaces of an IoT ecosystem comprise one or more vulnerabilities, which after exploitation will generate events belonging to one or more event categories. In turn, when an attacker launches an ill-intentioned activity over one of the IoT attack surfaces, exploiting an IoT vulnerability, at least one security event belonging to an event category is generated.

It has to be stated that the analyzed security event categories are based on the event categories proposed by OWASP. Furthermore, the relation with the attack strategy, representing the techniques used by the attacker, was performed leveraging the taxonomy of Common Attack Pattern Enumeration and Classification (CAPEC) (https://capec.mitre.org/data/definitions/1000.html.), from the US-CERT at the US Department of Homeland Security. Next we analyze in depth each of these 11 security event categories.

4.1. Request Exceptions. This category clusters events related to abnormal requests addressed to the device, e.g., invocation of unsupported HTTP methods or reception of parameters in an unexpected way (quantity, type). The existence of these request events over a vulnerable IoT device could indicate that an attack is in progress. A number of vulnerabilities that, when exploited, would generate events of request exceptions are as follows:

(1) Lack of controls to avoid username enumeration: a username enumeration could be executed when there are not enough mechanisms that monitor and restrict the insertion of custom fields or the reception of maliciously prepared authentication requests, e.g., an attacker sends abnormal requests to an IoT device, using a fuzzer, to collect valid usernames (CAPEC-261). This vulnerability can be found in different parts of the IoT architecture, mainly in the following attack surfaces: mobile application and device web interface.

(2) Lack of two-factor authentication for critical functions: the bypass of a primary authentication method could be achieved by delivering abnormal requests, e.g., an attacker sends requests to a web server without a token/cookie, but containing an unexpected quantity of characters, avoiding the web interface validation (CAPEC-31). The IoT attack surface device web interface can expose this concrete vulnerability.

(3) Lack of controls against Denial of Service: a DoS attack may be generated sending abnormal requests against a target which could produce disruption in operations, e.g., an attacker floods the network to impede communications for the IoT device (CAPEC-482). This vulnerability might be contained in the IoT attack surface device network service.

(4) IoT service contains insecure third party components: an attack could exploit a third party component

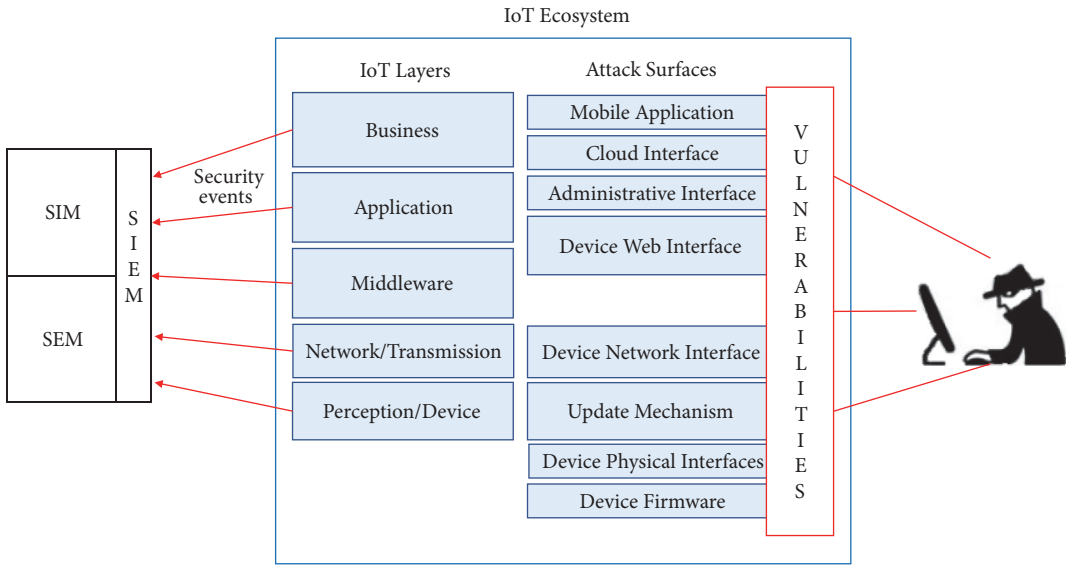


Figure 2: Proposal of integration of SIEM in an IoT ecosystem.

Table 1: Relations between event categories, vulnerability categories and attack surfaces on IoT ecosystems.

IoT Vulnerabilities	Event Categories											IoT Attack Surfaces								
	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
Lack of controls to avoid username enumeration	✘	✘		✘					✘			✘	✘	✘	✘			✘		
Lack of two-factor auth for critical functions	✘	✘		✘					✘			✘	✘	✘	✘			✘		
Lack of control against DoS attacks	✘	✘	✘	✘			✘		✘		✘	✘	✘	✘	✘			✘		✘
IoT service contains Insecure 3r party components	✘								✘					✘						
Use of weak password		✘		✘					✘			✘	✘	✘	✘			✘		
Lack of an account lockout after multiple failed attempts		✘		✘		✘			✘			✘	✘	✘	✘			✘		
Unencrypted network services allowing eavesdropping					✘				✘					✘				✘		
Lack of controls against manipulation of the code execution flow					✘	✘			✘											✘
Storage location for updates files is writable					✘				✘	✘				✘		✘			✘	
Lack of control for device console access				✘			✘	✘	✘		✘									✘
Update sent without encryption						✘			✘										✘	
Storage Media is physically unprotected							✘		✘		✘									✘
Possible Firmware and data extraction							✘		✘								✘			✘
Fail in the implementation of encryption mechanisms									✘	✘								✘		
Remote update is done without security controls									✘	✘									✘	
Lack of controls to avoid command injection									✘	✘				✘	✘					

Acronym and event categories: E_0, request exceptions; E_1, authentication exceptions; E_2, input exceptions; E_3, access control exceptions; E_4, session exceptions; E_5, ecosystem member exceptions; E_6, Device Access Events; E_7, admin mode events; E_8, honey trap exceptions; E_9, command injection exceptions; E_{10}, reputation exceptions. **acronym and IoT attack surfaces**: S_0, mobile application; S_1, cloud web interface; S_2, device web interface; S_3, admin interface; S_4, local data storage; S_5, Device firmware; S_6, device network services; S_7, update mechanism; S_8, device physical interfaces.

which has some functions for an IoT service, e.g., an attacker invokes abnormal HTTP methods to a third party component generating request events (CAPEC-86, CAPEC-460). The IoT attack surface device web interface may expose this specific vulnerability.

4.2. Authentication Exceptions. Those events related to the authentication process are grouped within this category and could include, for example, excessive login attempts or untrusted user location. Next we present some vulnerabilities that could exist in IoT ecosystems which when exploited would generate events of authentication exceptions type.

(1) Lack of controls to avoid username enumeration: a username enumeration could result successful through the delivery of probes to the authentication functions; e.g., an attacker starts a dictionary attack trying different common usernames (CAPEC-16). This vulnerability could be exposed across different parts of the IoT architecture, mainly in the following

attack surfaces: administrative interface, device web interface, coud web interface, and mobile application.

(2) Use of weak password: an attack against an IoT infrastructure can be performed by exploiting a weak password; e.g., an attacker starts a dictionary attack trying different common passwords. This vulnerability might be found throughout several components of the IoT architecture, specifically in the following attack surfaces: administrative interface, device web interface, cloud web interface, and mobile application.

(3) Lack of two-factor authentication for critical functions: a critical function in an IoT ecosystem should be protected by multifactor authentication so its access can not be exploited easily; e.g., an attacker introduces a valid username and password and tries to bypass a biometric control with a fake fingerprint (CAPEC-180). This vulnerability is contained in the following IoT attack surfaces: administrative interface, cloud web interface, and mobile application.

(4) Lack of an account lockout control after multiple failed attempts: an account lockout should be triggered upon a brute force attack; e.g., an attacker starts a brute force attack against an account that is not protected against multiple failed attempts, allowing an illegitimate access (CAPEC-49). We can find this vulnerability mainly in the next IoT attack surfaces: administrative interface, device web interface, cloud web interface, and mobile application.

(5) Lack of controls against Denial of Service: DoS attacks may succeed by exploiting the authentication functionalities of a IoT system; e.g., an attacker performs a DDoS attack through failed SSH connection attempts drowning the server or the IoT device (CAPEC-489). The IoT attack surface device network services exposes this particular vulnerability.

4.3. Session Exceptions. This category refers to those events stating that something abnormal in the creation, establishment or revocation of a session is happening. It could include, for example, a cookie being modified, receiving a request containing a valid session ID of a user different from the current one or unexpected changes of location of the user during the same session. In the following we introduce a few vulnerabilities from IoT ecosystems whose exploitation could generate events of session exceptions type.

(1) Unencrypted network services allowing eavesdropping: a session running under an unencrypted channel could affect seriously the operation of an IoT service, e.g., an attacker captures the unencrypted traffic transmitted in a session creation process and then they are able to sniff (CAPEC-158) and even modify packets (CAPEC-31) to send requests using a captured IoT device authentication token. We can observe this vulnerability in the IoT attack surfaces: device web interface and device network services.

(2) The storage location for updates files is writable: if a user is able to write in the file system of some of the components of the IoT ecosystem, she could affect the integrity of the service; e.g., the uploads folder has all permissions, or the uploads web interface is exposed to the Internet without authentication, so anyone could upload malicious files to the IoT device or cloud server (CAPEC-75). This vulnerability might be exposed by the next IoT attack surfaces: local data storage and device web interface.

(3) Lack of controls to avoid physical connection to the device causing manipulation of the code execution flow: the access to the physical components of an IoT ecosystem must also be protected to avoid affectations to confidentiality, integrity, or availability of the services; e.g., an attacker connects an external device to the JTAG interface of the IoT device and then starts an adulterated process to establish a session, which after being successful manipulates the code execution flow (CAPEC-391). The IoT attack surface device physical interfaces usually contain this specific vulnerability.

4.4. Access Control Exceptions. This category logs those events indicating that some access control functions are not working properly, specifically authorization functions. For example, a URL argument or POST parameter that has been modified in order to access a confidential object through an Insecure Direct Object Reference. The exploitation of the following vulnerabilities, present in IoT ecosystems, might generate events of access control exceptions type:

(1) Possible access to the device console because a lack of controls: console access should be restricted since an improper access to it could lead to the execution of an attack over the IoT service; e.g., an attacker gets legitimate accesses to an IoT device console but forces a switch out of users to get elevated privileges (CAPEC-233). This vulnerability is mainly found in the IoT attack surface device physical interfaces.

(2) Any vulnerability included in the authentication exception category could also generate an access control exception: a proper working of the authentication functions in IoT services is fundamental to support the execution of specific instructions; e.g., an attacker exploits an authentication vulnerability and then they get access to the privileges of the hacked account and they start invoking commands. The combination of authentication exception events and the later execution of commands could generate access control exception events. These vulnerabilities might be exposed by the following IoT attack surfaces: administrative interface, device web interface, cloud interface, mobile application, and device network services.

4.5. Ecosystem Membership Exceptions. This category encloses events related to the registration of the different components of the IoT ecosystems. For example, it puts on evidence traffic coming from an unregistered/revoked device and it also could include exceptions related to the registration

process. Next we describe a number of vulnerabilities commonly found in IoT ecosystems, whose exploitation might imply the generation of events of ecosystem membership exceptions type.

(1) Lack of account lockout control after multiple failed attempts: a fake IoT device may try a registration process repeatedly using different combinations of parameters to cheat the IoT ecosystem; e.g., attacker starts a brute force attack. This vulnerability in particular may come along with the following IoT attack surfaces: administrative interface, device web interface, cloud web interface, and mobile application.

(2) Update sent without encryption: the transmission of an update through an unencrypted channel may yield confidentiality affectations; e.g., an attacker captures the unencrypted traffic related to the update and uses it to discover the mechanism that is being used to register IoT devices, so, it can be replicated to register fake IoT devices (CAPEC-609, CAPEC-615). The IoT attack surface update mechanism usually exposes this specific vulnerability.

(3) Lack of controls to avoid physical connection causing manipulation of the code execution flow: unauthorized physical connections can harm the operations of IoT devices and consequently their services; e.g., an attacker connects an external device to the JTAG interface of the IoT device and then forces a reregistration or a revocation (CAPEC-390). The IoT attack surface device physical interfaces commonly contains this concrete vulnerability.

4.6. Device Access Events. In this category we can cluster the events generated whenever an access to the physical device occurs, such as the removal of some protection element belonging to the device or the manipulation of the hardware. A number of vulnerabilities specific to IoT ecosystems and whose exploitation could yield events of device access type are as follows:

(1) Storage media is physically unprotected and could be removed: storage protection should be implemented to avoid extraction or modification of sensitive data; e.g., an attacker accesses an IoT device and extracts the media containing the operative system and potentially confidential data (CAPEC-547). The IoT attack surface device physical interfaces usually contains this vulnerability.

(2) Firmware and data could be extracted allowing access to sensitive information: repository for firmware and data should also be physically protected to avoid data exposition; e.g., an attacker accesses to an IoT device and extracts the media containing firmware and potentially confidential data (CAPEC-547). This vulnerability might be found in the next IoT attack surfaces: device physical interfaces and device firmware.

(3) Lack of controls against Denial of Service: a successful DoS attack might be achieved by the physical access to some components for which an IoT component depends on; e.g., an attacker disconnects the device power supply and the administrative interfaces notice that the IoT device is offline (CAPEC-547). This specific vulnerability might be exposed across the following IoT attack surfaces: device physical interfaces and device network services.

(4) Possible access to the device console because of a lack of controls: the absence of physical barriers could entail an unauthorized access to an IoT device; e.g., an attacker accesses to an IoT device console and logs in as a user with some preset privileges (CAPEC-40). The IoT attack surface device physical interfaces often manifests this vulnerability in particular.

4.7. Administrative Mode Events. This category of security events deals with the use of administration privileges to perform some action over the device. Events can be outputted due to the raise of privileges over the console (even if this is done in an authorized or unauthorized way) and of each administrative action that is invoked thereafter. The principal vulnerability in the context of IoT ecosystems whose exploitation might end up in the generation of events of administrative mode type is as follows:

(1) Possible access to the device console access because of a lack of controls: console access should be restricted and monitored to avoid the use of privileged functions; e.g., an attacker accesses to an IoT device console and logins as a user with some preset privileges, intending an elevation of privileges (CAPEC-40, CAPEC-249). The IoT attack surface device physical interfaces usually embraces this concrete vulnerability.

4.8. Input Exceptions. This category includes events that are generated upon unexpected input data coming to applications or devices. This includes, for example, coding or format errors. The exploitation of the next key vulnerability, commonly found in IoT ecosystems, might yield events of input exceptions type.

(1) Lack of controls against Denial of Service: DoS attacks can be originated introducing malformed inputs in some user fields exposed in the IoT components; e.g., an attacker sends device abnormal input data in requests addressed to an IoT device, with the intention of generating a failure in the device network services and achieve a successful DoS (CAPEC-469). The IoT attack surface device network services typically contains this vulnerability.

4.9. Command Injection Exceptions. This category of events embraces those attempts of injections that contain execution commands. In the case of IoT devices, a code injection could be harmful as it could extract, modify, or delete information kept within an IoT device. A number of potential vulnerabilities that could be exploited in order to generate events of command injection exceptions type are as follows:

(1) Failure in the implementation of encryption mechanisms: commands can be injected when a channel does not use encryption or the encryption implementation is weak; e.g., an attacker takes benefit of a failure in the implementation of the encryption algorithm, so they can decrypt, modify (inject commands), and encrypt again (CAPEC-20, CAPEC-463). This vulnerability could be found in the IoT attack surface device network services.

(2) Remote update is done without security controls: the updating process should be conducted under a ciphered channel to avoid affectations to the integrity; e.g., an attacker intercepts the unencrypted traffic related to an update and modifies it to inject a specific command that will be executed during a critical task (CAPEC-187, CAPEC-533). The IoT attack surface update mechanism commonly exposes this specific vulnerability.

(3) The storage location for updates files is writable: data stored in file systems should be protected to avoid modification and injection of commands; e.g., an attacker discovers a way to overwrite a data file in an IoT device, so they modify it to include a specific command (CAPEC-75). This file is executed later by the IoT device and the commands could get operative. The IoT device could detect the syntax of the command since such element is not expected in a data file. This vulnerability is typically exposed in the IoT attack surface update mechanism.

(4) Lack of controls to avoid command injection: appropriated controls to avoid command injection in all levels should be implemented to avoid a high impact risk; e.g., an attacker notices that the configuration web interface of an IoT device has insecure inputs and they inject command parameters to be executed on the IoT device, corrupting device data, and disabling the accountability functionality or information disclosure (CAPEC-500). This specific vulnerability is commonly found in the following IoT attack surfaces: device web interface and administrative interface.

4.10. Reputation Exceptions. This category contains those events related to situations where trust and reputation are involved. For instance, a user access from a new and remote location could be considered as dangerous and this could affect the trust that the IoT service would give to such access. Next we introduce some vulnerabilities in the context of IoT ecosystems whose exploitation could generate events of reputation exceptions type.

(1) Lack of controls against Denial of Service: reputation could be used to define the normal access to IoT services and avoid attacks such as DoS; e.g., an attacker programs a botnet to send multiple ill-intentioned requests to an IoT device (CAPEC-469). Such device detects requests coming from unusual sources and therefore it assigns a low trust to these connections. The IoT attack surface device network services commonly exposes this specific vulnerability.

(2) Storage media is physically unprotected and could be removed: IoT device integrity can be considered to avoid unauthorized hardware modifications and redefine reputation indicators; e.g., an attacker changes the device storage media to collect information (CAPEC-547). This vulnerability is frequently present in the IoT attack surface device physical interfaces.

(3) Possible access to the device console access because of a lack of controls: abnormal executions can also influence the reputation indicators of the IoT ecosystem; e.g., when a user gets access to the console under suspicious circumstances, the IoT ecosystem would trust less in such device until a confirmation of safety. The IoT attack surface device physical interfaces commonly includes this vulnerability in particular.

4.11. Honey Trap Exceptions. This category includes those events outputted by traps for the attackers. A trap could, for example, expose tempting but fake data, so that the attacker gets motivated to start a penetration activity toward the IoT ecosystem. This includes, for example, events generated from the access to a trap resource. The key vulnerability that could exist in IoT ecosystems whose exploitation would generate events of honey trap exceptions type is described next.

(1) The exploitation of all the IoT vulnerabilities that have been mentioned previously could entail the generation of honey trap exceptions. However, considering the nature of honey traps, the most commonly exploited vulnerabilities could be (i) lack of controls to avoid username enumeration, (ii) lack of two-factor authentication for critical functions, (iii) the storage location for updates files being writable, and (iv) remote update being done without security controls.

Honey traps exposing obvious (but bogus) vulnerabilities can be useful to make intelligence over the adversaries; e.g., an attacker probes a system structure related to an IoT service to evaluate its security level, mapping the application, and exploiting an apparent update location writable vulnerability coping files to common resource locations and executing commands (CAPEC-40, CAPEC-150).

Likewise, each of these vulnerabilities can be related to multiple IoT attack surfaces.

In this section we presented a security solution (Figure 2) employing a SIEM to receive, process, and manage security events coming from the different components of an IoT ecosystem. The key element to consider behind the process done within the SIEM is the proposed multirelation between security event categories, attack surfaces, and vulnerabilities, represented in Table 1. The multirelations were built as a result of the analysis done in this section around 11 security event categories. This proposed multirelations will be used in the next section to support building correlation rules in a SIEM, allowing getting a better knowledge about the IoT security risks.

It has to be stated that the process of integrating a SIEM solution within the IoT ecosystem is not trivial. As a matter of fact, some challenges have to be addressed in order to accommodate this procedure. Among them, one of the most relevant is represented by the heterogeneity exposed by the IoT devices. That is, the intelligent *things* can be any kind of device, using diverse protocols and data format to exchange information [43]. Consequently, the system administrator is in charge of understanding and manually managing the events stemming from the different sources in order to collect and organize them. To this extent, great effort is required in order to standardize the communication among the IoT nodes. Moreover, the scalability of the SIEM can be undermined as the amount of resources needed to handle the IoT data and then run analytics increases exponentially. To tackle this problem, it is indeed possible to exploit the benefits of Big Data technologies. By leveraging its powerful capabilities, it is possible to minimize the amount of data which the SIEM solution has to store and, therefore, analyze [44].

5. Alerts and Correlation Rules

In this section, alerts and correlation rules are built according to the security events emanating from the different components of an IoT ecosystem. In our idea, this task must help the security administrator of the IoT infrastructure to better understand the existing correlations between the events generated within the IoT system. Correlation rules support a prevention, detection, and reaction security strategy, as they identify anomalous or suspicious situations in an IoT ecosystem evidencing that something potentially harmful happened, is happening now, or will happen soon.

In turn, the IoT security events depend on the nature of the IoT device and are related to the IoT architecture components [11] (see Figure 2). Regarding the application and business components, there are security events related to the code and the operations performed by the IoT applications including local, web, cloud, and mobile operations. Examples can include events like input or output of data, execution of functions, and abnormally reported or measured values. In the middleware component, events are related to the database access, component/user security, and privacy and interoperation. Examples can include user or things authentication, database privileges, and inconsistencies between applications and platforms reported. In the network and transmission components we can find security events related to the communication between devices and through a server located in the Internet using specific network protocols. Examples of events are validation of data in the destination, transmission errors, and communication method. In the perception or device component we find events related to the electronic components that conform the IoT device. Examples of events could include the validation of integrity of the device using a *Trusted Platform Module* (TPM), which leverages the concept of root of trust [45]. Specifically in this Section, 3 security events will be further detailed.

Regarding the generation of correlation rules, favorably a number of different algorithms have been presented so far, which can be categorized according to their characteristics [46]: (i) similarity-based, where events are compared based on their similarity and grouped over time, (ii) knowledge-based, where events are compared against a database of attack patterns, and (iii) statistical-based, where statistical attributes in attacks are analyzed, specially their frequency of occurrence with respect to past statistical data.

Correlation rules are usually created in a manual way applying some of the aforementioned algorithms, and in practice different threat intelligence providers offer feed services to guarantee a regular update of rules. In order to help the security administrators to generate correlation rules, this paper proposes a mapping between events, vulnerabilities, and attack surfaces for IoT ecosystem (see Section 4). The approach proposed here is a knowledge-based algorithm, since the mapping is actually a taxonomy of events, vulnerabilities, and attack surfaces, constituting the baseline for an attack pattern. The generation of the correlation rules in the following subsections was straightforward for the security administrator thanks to the existence of such proposed mapping.

The rule generation process firstly consists of conducting a threat analysis for a specific IoT ecosystem. This analysis is made by reading the mapping as shown in Table 1, i.e., starting from an security event stemming from an IoT ecosystem, belonging to an event category, and indicating that the exploitation of an IoT vulnerability has occurred over a specific attack surface. So, the security administrator must generate at least one correlation rule for such situation capturing as well the identified security events under the affected attack surface, and additionally preparing an immediate remediation over the identified IoT vulnerabilities.

The threat analysis can also be made reading Table 1 in the following way: the IoT ecosystem exposes some attack surfaces comprising one or more vulnerabilities, whose exploitation will generate events belonging to one or more event categories. So, the security administrator must generate at least one correlation rule for such a situation that has an attack surface as a starting point and then define all the possible security events, with their corresponding event categories that can come from there. Additionally, the security administrator must also define the remediation to be enforced in order to patch the related IoT vulnerability.

In any case, the algorithms to generate correlation rules require the existence of a knowledge base with suspicious activities. Such dubious activities can be recorded and marked individually as alerts, and the correlation of alerts, done through a correlation rule in the SIEM, can proclaim an attack.

When the SIEM is running in a state where all incoming events are analyzed against the set of active correlation rules, and if one of the events matches some previously defined conditions for a rule, the SIEM will register it until a new related event appears. This situation is done until a threshold is reached confirming an attack scenario, so the system administrator is informed and an incident response as part of a defensive safety strategy is triggered. These algorithms can be very efficient for their accuracy, ability to avoid false alerts, and capacity to detect multistage attacks. However, these correlation rules can be difficult to define, since their

```
1  initialize bufferOfEvents to zero
2  initialize threshold to maximumTolerable
3  while bufferOfEvents is not empty
4    Get newEvent from bufferOfEvents
5    Analize newEvent against correlationRules
6    If newEvent matches a correlationRule
7      matchedRuleCounter = matchedRuleCounter + 1
8      If matchedRuleCounter >= threshold
9        print "Event confirms an attack in progress"
10       Inform the system Admin
11       Launch an Incident Response
12     else
13       print "Event matches a rule but is not even an attack"
14   else
15     print "Event did not match any rule"
```

Listing 1: Reception and processing of security events by a SIEM.

arguments must be set in a precise way and are also deficient against new attacks (see Listing 1).

Next, we will analyze three scenarios that are of special interest in IoT ecosystems, namely, geofencing, brute force, and command injection. In each scenario, we study some representative security events to define correlation rules implemented in a SIEM, which will aid the IoT architecture administrators in their security labors. The rules have been implemented on the Open-Source Security Information and Event Management System (OSSIM) which is one of the most popular open-source SIEMs, with a strong supporting community which in turn could be helpful to solve emerging issues.

5.1. Scenario 1: Geofence Attack in IoT Devices with Geolocation Functions. The first scenario is given by an IoT device moving within a delimited zone called *safe zone*. A device under this condition is called *geofenced*. Geofencing refers to a "virtual barrier or geographical border around a single point with a predefined set of boundaries on a geographical area mapped either with GPS or RFID". In this case, a safe zone has been defined for an IoT device and if the device exits the safe zone the SIEM will generate an alert.

As an example, a malicious user (attacker) takes possession of an IoT device (e.g., smart bike or vehicle) and intends to steal it. Within this scenario, the IoT device constantly validates its current position within the established geofence and when the device gets out of the safe zone, it sends a Device Access event to the SIEM informing about the theft. The Device Access event will be reported to the SIEM containing a correlation rule which validates the received information and generates an alert, so the theft can be prevented. In this scenario, the vulnerability exploited is Denial of Service, found in the attack surface device physical interface.

Hence, the correlation rule created for this scenario can be observed in Table 2, named, "IoT Geofencing Directive", comprising the following: (i) a "User data" defined in "GE1" which is the keyword used to notify the SIEM about a theft, (ii) a "Reliability" of 8 (in a scale up to 10, used internally to calculate the risk), (iii) a no defined "Timeout" (meaning that the frequency of the event in a time window is irrelevant), and (iv) an "Occurrence" of 1 (meaning that one occurrence of a Device Access event with a "User data" in "GE1" is enough to confirm the theft). This rule uses an OSSIM specially crafted plug-in that implements a regular expression to receive Device Access events from all the IoT devices being monitored.

5.2. Scenario 2: Brute Force Attack upon Failed Login in IoT Devices with Authentication Model N:N. The second scenario is represented by an attacker who attempts to perform a brute force attack against an application in an IoT device. The *N*:*N* authentication model is based on the interaction between an IoT device with more than one user and a user that could interact with a lot of components. The concurrent accesses or multiple accesses could affect the availability of an IoT device. Thus, a successful DoS may be achieved by the attacker trying multiple combinations of usernames and passwords through a brute force attack, until a valid username and password combination is found. It is important to avoid this type of attack because once the attacker enters the system as a valid user, she could exploit new vulnerabilities and would be considered as a trusted user within the system. The brute force attack should be detected through a correlation rule.

To this end, an authentication exception event is generated every time an invalid login is attempted. If any of these events constantly appears within a very short period of time, the correlation engine should generate an alert. In this scenario, the authentication exception events will be the evidence that any of the following vulnerabilities has been exploited: username enumeration, use of weak passwords, account lockout, or two-factor authentication. Additionally, these vulnerabilities can be applicable in the following attack surfaces here: administrative interface, device web interface, cloud interface, and mobile application.

Therefore, the correlation rule created for this scenario can be observed in Table 2 and is named "IoT rule", composed of (i) a "Reliability" of 1 (in a scale up to 10, used internally to calculate the risk), (ii) a no defined "Timeout"

Table 2: Correlation rules for different attacks scenarios implemented in OSSIM.

Scenario	Rule name	Reliability	Timeout [sec]	Occ	Security event	User data	Vulnerability exploited	Attack surface related
1	IoT GeoFencing Directive	8	None	1	Device Access event	GE1	Denial of Service	Device physical interface
2	IoT rule	1	None	1	Authentication Exception event	BF1	Username enumeration, Use of weak passwords, Account lockout or two-factor authentication	Administrative interface, Device web interface, Cloud interface and Mobile application
	Brute Attack	3	5	5				
3	IoT Command injection	3	None	1	Command Injection Exceptions event	AE1	Encryption mechanisms impl. fails, Remote update is done without security controls or Storage location is writable	Device network services and Update mechanism

(meaning that the frequency of the event in a time window is meaningless), and (iii) an "Occurrence" of 1 (meaning that one occurrence of an authentication exception event with a "User data" in "BF1" is enough to confirm the alarm). This rule uses an OSSIM specially crafted plug-in that implements a regular expression to receive authentication exception events from all the IoT devices being monitored.

Additionally, there is a second nested rule called "brute force attack" which is similar to the "IoT rule", where the reliability will increase from 1 to 3 when at least 5 events are received (occurrence) in a time window of 5 seconds (timeout). A bigger reliability means that the associated risk will be higher.

5.3. Scenario 3: Command Injection Attack in IoT Devices Deployed in Hostile Environments. In the third scenario, the attacker wants to access or modify data stored within an IoT device through a command injection attack, where a malicious code is interpreted and executed over the administrative or device web interface. The injected code would allow modifying, stealing, or even eliminating data. Even a DoS attack could be provoked by the injection leaving the resource inaccessible.

Here, a command injection exceptions event is generated as a consequence of the recognition of unusual characters, e.g., SQL sentences embedded in the data that the IoT device receives. In this scenario, this event will be an evidence that any of the following vulnerabilities have been exploited: fails in the implementation of the encryption mechanisms, remote update being done without security controls, or storage location for updates files being writable. Finally, the aforementioned vulnerabilities can be applicable in the following attack surfaces: device network services and update mechanism.

Thus, the correlation rule created for this scenario can be observed in Table 2 and is named "IoT command injection", including (i) a "User data" defined in "AE1" which is the keyword used to notify the SIEM about an injection on the IoT device, (ii) a "Reliability" of 3 (in a scale up to 10, used internally to calculate the risk), (iii) a no defined "Timeout" (meaning that the frequency of the event in a time window is irrelevant), and (iv) an "Occurrence" of 1 (meaning that one occurrence of a command injection exceptions event with a "User data" in "AE1" is enough to confirm the alarm). This rule uses an OSSIM specially crafted plug-in that implements a regular expression to receive command injection exceptions events from all the IoT devices being monitored.

6. Incident Responses and Defensive Safety

It is clear that managing large amounts of events is a burden in the work of system administrators and that using a SIEM facilitates their work, especially when correlation rules and countermeasures can not be easily elicited by a nonexpert professional. Hence, this section deals with the incident response and defensive security based on the threats identified in the previous section. A security incident is defined as "a violation or imminent threat of violation of informatics security policies, acceptable use policies, or standard security practices" [47]. The NIST's (National Institute of Standards and Technology) Incident Management Guide establishes four phases: (i) preparation; (ii) detection and analysis; (iii) containment, eradication, and recovery; and (iv) postincident activity. Assuming a correct detection and analysis phase, the security incident must be detected and consequently an optimal reaction against it must be triggered [48]. This reaction must balance the inherent trade-off between cost of the counteraction and its negative impact on the system [49].

There are five types of security controls which treat the risk associated with an attack [50]: (i) *preventive*, to avoid the incident, (ii) *detective*, to discover an incident in progress, (iii) *corrective*, to decrease the impact, (iv) *deterrent*, to discourage the incident, and (v) *compensatory*, to place controls instead of more desirable controls as an alternative safeguard. Additionally, security controls also have three kinds of implementation [50]: (i) *administrative*, based on procedures or policies, (ii) *technical*, based on hardware or software components, and (iii) *physical*, based on elements supporting physical security. To this extent, SIEM systems can help in the detection and analysis phase through correlation rules and alarms, but they can also support the containment, eradication, and recovery phase

through the generation of responses as part of a defensive security strategy. Finally, SIEM systems can also contribute in the postincident activity phase based on its Digital Forensics capabilities [51].

Generally, the SIEM systems are equipped with a response-focused component that is able to execute actions triggered by a security incident (e.g., a confirmed intrusion is detected). These actions can be clustered in (i) active responses (technical controls intended to neutralize the attack) and (ii) passive responses (technical controls intended to alert the administrator about the detected incident) [52]. Therefore, a SIEM system could be used to support an incident response and defensive security strategy for an IoT ecosystem having the following considerations: IoT devices are generally implemented over computationally constrained hardware; there can exist transactions between IoT devices and also between the IoT device and an IoT platform remaining generally in the cloud, the SIEM system must be able to analyze trillions of IoT devices, get evidence from any kind of formats, scan different kinds of networks such as RFID or sensor networks, and handle exabytes of data, among others.

Additionally, a SIEM must secure all the gathered data as it could contain sensitive information about citizens, industrial processes, buildings, transportation means, etc. [53]. According to the types of technical security controls mentioned previously, some of them are defined for each one of the scenarios defined in the previous section. For each case, the affected IoT device or platform and the scenario are also described.

6.1. Scenario 1: Geofence Attack in IoT Devices with Geolocation Functions. As part of the defensive security strategy for this scenario, several technical security controls can be defined, as shown next.

(i) **Preventive**: create a correlation rule in the SIEM that identifies when the location of a device is suspiciously close to the boundary of the geofence, so an alarm can be triggered to avoid a possible theft.

(ii) **Detective**: implement a correlation rule in the SIEM that detects when the IoT device is out of the geofence and warn the security manager, security personal, or the owner to avoid the device theft.

(iii) **Corrective**: implement an action in the SIEM that turns off the device through a command, to avoid the use of the IoT device. Alternatively, implement an action that sends the current device location to its legitimate user and/or the system administrator to avoid the theft.

(iv) **Deterrent**: implement an action in the SIEM system that sends a command to the device to emit a sound that alerts the device theft and deters the thief from continuing with the robbery.

(v) **Compensating**: if the SIEM detects that the theft could not be avoided, execute an action to delete all the information, including a deep erase, so the configuration or operation information cannot be compromised.

6.2. Scenario 2: Brute Force Attack upon Failed Login in IoT Devices with Authentication Model N:N. In this scenario, we propose the following technical security controls following the taxonomy proposed in [50]:

(i) **Preventive**: create a correlation rule in the SIEM that identifies when a brute force attack is in progress, due to the unusual characters being received, so an alarm can be triggered to avoid the attack being successful.

(ii) **Detective**: implement a correlation rule in the SIEM that detects when a brute force attack in progress discovers a set of user credentials. Warn immediately the security manager to avoid that the user account can be exploited.

(iii) **Corrective**: implement an action in the SIEM that enables the resolution of a captcha or the application of a two-step authentication process for new user authentication attempts [54].

(iv) **Deterrent**: implement an action in the SIEM system that, through a command, takes a picture with the front face camera (to photograph the attacker) and shows it in the device's display with a message asking to stop the attack. This control could be valid in case that the attack was performed locally from the IoT device, not remotely.

(v) **Compensating**: if other controls do not work, the SIEM could execute a script to enable rules in a WAF (Web Application Firewall) available in the network that avoids the brute force attack.

Any of these responses could be accompanied by an email to the IoT device owner, notifying about the possible attack, the device information, and where the attack comes from.

6.3. Scenario 3: Command Injection Attack in IoT Devices Deployed in Hostile Environments. As part of the defensive security strategy, different technical security controls can be defined for this scenario, as shown next.

(i) **Preventive**: create a correlation rule in the SIEM that identifies when a command injection attack is in progress, detecting that unusual characters are being received in a user input field, so an alarm can be triggered to avoid the attack being successful.

(ii) **Detective**: implement a correlation rule in the SIEM that detects when a command injection attack in progress manages to execute a special command in the device of the victim. Warn immediately the security manager to avoid malicious operations.

(iii) **Corrective**: implement an action in the SIEM that increases the level for the security policies running in the operative system of the victim, so special commands cannot be run immediately.

(iv) **Deterrent**: when unusual characters are detected in the security events coming to the SIEM, it can start an action that sends a warning message to the attacker informing that the attack will be reported and its consequences.

(v) **Compensating**: if other controls do not work, the SIEM could execute a script to enable rules in a WAF (Web Application Firewall) available in the network that avoid the command injection attack.

6.4. Evaluation. The evaluation of our proposal was conducted through the emulation of an IoT-secure device built under the premise of security by designed. Our IoT-secure device, running Ubuntu IoT OS, implements a number of security functions through the incorporation of some security modules, specifically (i) a module to detect security anomalies around the Ethernet traffic, i.e., Suricata IoT IDS (https://github.com/decanio/suricata-IoT), (ii) a module to detect anomalies in wireless access networks like implementations based on 802.11 specifications, i.e., Kismet IDS (https://www.kismetwireless.net/), and (iii) a vulnerability scanning engine that builds and sends vulnerabilities reports, i.e., OpenVAS (http://www.openvas.org/). Each one of these modules is able to send security events toward a SIEM server implemented using OSSIM, which has a set of correlation rules, designed and implemented leveraging the proposed security architecture and multirelations mapping described in Section 4. Security events are sent using the Syslog standard.

Suricata IoT required 3 configurations, namely, definition of a common identity for all security events reported by the IoT device, definition of internal/external network addresses, and definition of IDS rules to be used (ours use just emerging-dos and emerging-webserver rules). Additionally, on the server side, an specific plug-in was set containing a regular expression for OSSIM to receive and understand the security events coming from Suricata. The security events received by the OSSIM contain the following fields: malicious source IP address, malicious source port, malicious destination IP address, malicious destination port, matched Suricata rule ID, and rule priority.

In turn, Kismet IDS required a wireless antenna able to work on monitor mode (TP-LINK WN722N). Kismet configuration required to set the wireless adapter in monitor mode and define the name of the interface that scan the media. Furthermore, it was required to implement a client to forward to OSSIM all messages generated and stored locally by Kismet IDS. Besides, on the server side, a specific plug-in was set containing a regular expression for OSSIM to receive and understand the security events coming from Kismet. The security events received by OSSIM contain the following fields: malicious source IP address, malicious source MAC address, malicious destination MAC address, wireless channel, and matched Kismet rule ID.

In the case of OpenVAS, it required the development of a client sending all the vulnerability reports toward OSSIM. This client was implemented using the API OpenVAS Management Protocol (OMP). Moreover, on the server side, a specific plug-in was set containing a regular expression for OSSIM to receive and understand security events coming from OpenVAS. The security events received for the OSSIM contain the following fields: vulnerability ID, IP address, vulnerability severity, and the Common Vulnerabilities and Exposures (CVE) code.

The IoT-secure device was a Raspberry Pi3 (1 GB RAM, Quad Core 1.2 GHz Broadcom BCM2837, 100 base Ethernet, 10 GB ROM) with an integrated wireless antenna TP-LINK WN722N. On the other hand, the SIEM server was implemented with OSSIM 5.4.1 (8 GB RAM, Intel(R) Xeon(R) CPU E5-2620 v3, 2.40GHz, 8 cores, 1 GG Ethernet, 100 GB ROM) with a total set of 16 plug-ins. OSSIM does not have an initial restriction regarding the creation of correlation rules. Yet, it does not have directives like DoS, network, and SCADA, which are only available for the commercial version. Correlation rules were implemented as indicated in the scenarios described in Section 5. Last but not least, the OSSIM server was successful to receive the security events form the IoT device, perform a match with the correlation rules, and generate an alert.

7. Conclusions and Future Work

The IoT world represents unquestionably a great opportunity to benefit our everyday life [55]. Several applications have been already developed to integrate IoT devices within the existing network infrastructures.

Along this paper, a comprehensive study around security aspects in IoT devices has been conducted, reviewing the most common security events, vulnerabilities, and attack surfaces. Specifically, we proposed a multirelations mapping among these three categories, which we believe may be helpful to support the security administrator to better understand the causes, symptoms, and attack vectors of security incidents. Specifically, security events have been used to generate correlation rules which are used by a SIEM system to detect security incidents.

Defensive security has also been explored describing different possible actions that could be implemented from a SIEM in order to decrease the impact of an incident (if it has already occurred) or avoid the progress of the incident (if it has been detected that an adversary is trying to exploit a vulnerability).

Future works include the definition of new techniques that allow the generation of correlation rules in a more automatic way, so the SIEM system may be able to face different combinations of events, vulnerabilities, and threats.

Automatic rules generation can be grounded on techniques like semantic web to build and maintain a complete taxonomy applicable for different components of an IoT ecosystem. This initiative would require a complete characterization of the IoT ecosystem to identify all the applicable attack surfaces and guarantee a proper security event generation coming from IoT devices.

Moreover, a concrete evaluation of the overhead needed to effectively introduce a SIEM solution within the IoT ecosystem is worth of investigating. Finally, we are currently working in the deployment of a SIEM solution over an IoT

device thinking in situations where it is necessary to have a security solution just inside the IoT ecosystem.

Data Availability

The IoT vulnerabilities, attack surfaces, and event categories data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work has been supported by the European Commission Horizon 2020 Programme under Grant Agreement no. H2020-ICT-2014-2/671672-SELFNET (*Framework for Self-Organized Network Management in Virtualized and Software Defined Networks*), by a Ramón y Cajal Research Contract (RYC-2015-18210) granted by the MINECO (Spain) and cofunded by the European Social Fund, by a Leonardo Grant 2017 for Researchers and Cultural Creators awarded by the BBVA Foundation, and by the Colombian School of Engineering Julio Garavito (Colombia) through the project "Developing of a SIEM solution (Security Information and Event Management System) for information asset protection", approved by the Internal Research Opening 2016.

References

[1] K. Yelamarthi, M. S. Aman, and A. Abdelgawad, "An application-driven modular IoT architecture," *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 1350929, 16 pages, 2017.

[2] Gartner, "Gartner's 2016 Hype Cycle for Emerging Technologies," 2016. [Online]. Available: https://www.gartner.com/newsroom/id/3412017.

[3] S. Li, L. D. Xu, and S. Zhao, "The internet of things: a survey," *Information Systems Frontiers*, vol. 17, no. 2, pp. 243–259, 2015.

[4] S. Haller, S. Karnouskos, and C. Schroth, "The Internet of Things in an Enterprise Context," in *Future Internet – FIS 2008*, vol. 5468 of *Lecture Notes in Computer Science*, pp. 14–28, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[5] A. Abdelgawad and K. Yelamarthi, "Internet of things (IoT) platform for structure health monitoring," *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 6560797, 2017.

[6] B. R. Stojkoska, K. Trivodaliev, and D. Davcev, "Internet of things framework for home care systems," *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 8323646, 2017.

[7] B. Gomes, L. Muniz, F. J. da Silva e Silva, L. E. Rios, and M. Endler, "A comprehensive cloud-based IoT software infrastructure for Ambient Assisted Living," in *Proceedings of the 2015 International Conference on Cloud Technologies and Applications (CloudTech)*, pp. 1–8, Marrakech, Morocco, June 2015.

[8] S. Charmonman and P. Mongkhonvanit, "Special consideration for Big Data in IoE or Internet of Everything," in *Proceedings of the 13th International Conference on ICT and Knowledge Engineering, ICT and KE 2015*, pp. 147–150, Thailand, November 2015.

[9] J. Granjal, E. Monteiro, and J. Sá Silva, "Security for the internet of things: a survey of existing protocols and open research issues," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1294–1312, 2015.

[10] A. Zanella, N. Bui, A. P. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.

[11] R. Khan, S. U. Khan, and R. Zaheer, "Future internet: the internet of things architecture, possible applications and key challenges," in *Proceedings of the 10th International Conference on Frontiers of Information Technology (FIT' 12)*, pp. 257–260, December 2012.

[12] V. Beltran, A. F. Skarmeta, and P. M. Ruiz, "An ARM-Compliant Architecture for User Privacy in Smart Cities: SMARTIE—Quality by Design in the IoT," *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 3859836, 13 pages, 2017.

[13] Y. H. Hwang, "IoT security & privacy: Threats and challenges," in *Proceedings of the 1st ACM Workshop on IoT Privacy, Trust, and Security, IoTPTS 2015*, p. 1, Singapore.

[14] F. Gómez Mármol, M. Gil Pérez, and G. Martínez Pérez, "I Don't Trust ICT: Research Challenges in Cyber Security," in *Trust Management X*, vol. 473 of *IFIP Advances in Information and Communication Technology*, pp. 129–136, Springer International Publishing, Cham, 2016.

[15] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: the road ahead," *Computer Networks*, vol. 76, pp. 146–164, 2015.

[16] I. Yaqoob, E. Ahmed, M. H. U. Rehman et al., "The rise of ransomware and emerging security challenges in the Internet of Things," *Computer Networks*, vol. 129, pp. 444–458, 2017.

[17] D. S. Lavrova, "An approach to developing the SIEM system for the Internet of Things," *Automatic Control and Computer Sciences*, vol. 50, no. 8, pp. 673–681, 2016.

[18] P. Zegzhda, D. Zegzhda, M. Kalinin, A. Pechenkin, A. Minin, and D. Lavrova, "Safe integration of SIEM systems with Internet of Things: Data aggregation, integrity control, and bioinspired safe routing," in *Proceedings of the 9th International Conference on Security of Information and Networks, SIN 2016*, pp. 81–87, USA, July 2016.

[19] G. Ho, D. Leung, P. Mishra, A. Hosseini, D. Song, and D. Wagner, "Smart locks: Lessons for securing commodity internet of things devices," in *Proceedings of the 11th ACM Asia Conference on Computer and Communications Security, ASIA CCS 2016*, pp. 461–472, Xi'an, China, June 2016.

[20] M. Woschek, "Owasp cheat sheets," *pp*, vol. 315, p. 4, 2015, https://www.owasp.org/images/9/9a/OWASP_Cheatsheets_Book.pdf.

[21] ISO/IEC, "ISO/IEC 27032:2012 - Information technologyâ€"Security techniquesâ€"Guidelines for cybersecurity," https://www.iso.org/standard/44375.html, 2012.

[22] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: mirai and other botnets," *IEEE Computer Society*, vol. 50, no. 7, pp. 80–84, 2017.

[23] A.-R. Sadeghi, C. Wachsmann, and M. Waidner, "Security and privacy challenges in industrial internet of things," in *Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC '15)*, pp. 1–6, IEEE, San Francisco, Calif, USA, June 2015.

[24] A. O. Prokofiev, Y. S. Smirnova, and V. A. Surov, "A method to detect Internet of Things botnets," in *Proceedings of the 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, pp. 105–108, Moscow, January 2018.

[25] V. Adat and B. B. Gupta, "Security in Internet of Things: issues, challenges, taxonomy, and architecture," *Telecommunication Systems*, vol. 67, no. 3, pp. 423–441, 2018.

[26] ISO/IEC, "ISO/IEC 27000:2018 - Information technology-Security techniques-Information security management systems-Overview and vocabulary," https://www.iso.org/standard/73906.html, 2018.

[27] S. Gupta, B. S. Chaudhari, and B. Chakrabarty, "Vulnerable network analysis using war driving and Security intelligence," in *Proceedings of the 2016 International Conference on Inventive Computation Technologies, ICICT 2016*, India, August 2016.

[28] J. R Vacca, *Network and system security*, S. Elliot, Ed., Syngress - Elsevier, 2014.

[29] Y. Chahid, M. Benabdellah, and A. Azizi, "Internet of things security," in *Proceedings of the 2017 International Conference on Wireless Technologies, Embedded and Intelligent Systems, WITS 2017*, Morocco, April 2017.

[30] R. Van Rijswijk and E. Poll, "Using trusted execution environments in two–factor authentication: comparing approaches," ser. Lecture Notes in Informatics. 1em plus 0.5em minus 0.4em Bonn, Germany: Gesellschaft for Informatik, 9 2013, pp. 20–31.

[31] C. Doukas, I. Maglogiannis, V. Koufi, F. Malamateniou, and G. Vassilacopoulos, "Enabling data protection through PKI encryption in IoT m-Health devices," in *Proceedings of the 12th IEEE International Conference on BioInformatics and BioEngineering, BIBE 2012*, pp. 25–29, November 2012.

[32] W.-I. Bae and J. Kwak, "Smart card-based secure authentication protocol in multi-server IoT environment," *Multimedia Tools and Applications*, pp. 1–19, 2017.

[33] B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in Internet of Things," *Journal of Network and Computer Applications*, vol. 84, pp. 25–37, 2017.

[34] W. Meng, "Intrusion Detection in the Era of IoT: Building Trust via Traffic Filtering and Sampling," *The Computer Journal*, vol. 51, no. 7, pp. 36–43, 2018.

[35] I. Smith and D. Bailey, "IoT Security Guidelines for Endpoint Ecosystem," GSM Association, Tech. Rep., 2016. [Online]. Available: https://www.gsma.com/iot/wp-content/uploads/2016/02/CLP.13-v1.0.pdf.

[36] P. Kasinathan, G. Costamagna, H. Khaleel, C. Pastrone, and M. A. Spirito, "Demo: An ids framework for internet of things empowered by 6lowpan," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer; Communications Security*, pp. 1337–1340, 2013.

[37] P. Nespoli and F. Gómez Mármol, "e-Health Wireless IDS with SIEM integration," in *IEEE Wireless Communications and Networking Conference (WCNC'18)*, Barcelona, Spain, 2018.

[38] A. Sforzin, F. G. Marmol, M. Conti, and J. Bohli, "RPiDS: Raspberry Pi IDS — A Fruitful Intrusion Detection System for IoT," in *Proceedings of the 2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, pp. 440–448, Toulouse, France, July 2016.

[39] I. Kotenko and A. Chechulin, "Computer attack modeling and security evaluation based on attack graphs," in *Proceedings of the 2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems, IDAACS 2013*, pp. 614–619, Germany, September 2013.

[40] I. Kotenko and A. Chechulin, "Common Framework for Attack Modeling and Security Evaluation in SIEM Systems," in *Proceedings of the 2012 IEEE International Conference on Green Computing and Communications (GreenCom)*, pp. 94–101, Besancon, France, November 2012.

[41] G. Kambourakis, C. Kolias, and A. Stavrou, "The Mirai botnet and the IoT Zombie Armies," in *Proceedings of the 2017 IEEE Military Communications Conference, MILCOM 2017*, pp. 267–272, USA, October 2017.

[42] D. Geneiatakis, I. Kounelis, R. Neisse, I. Nai-Fovino, G. Steri, and G. Baldini, "Security and privacy issues for an IoT based smart home," in *Proceedings of the 40th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2017*, pp. 1292–1297, Croatia, May 2017.

[43] Z.-K. Zhang, M. C. Y. Cho, C.-W. Wang, C.-W. Hsu, C.-K. Chen, and S. Shieh, "IoT security: ongoing challenges and research opportunities," in *Proceedings of the 7th IEEE International Conference on Service-Oriented Computing and Applications (SOCA '14)*, pp. 230–234, IEEE, Matsue, Japan, November 2014.

[44] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. Ullah Khan, "The rise of 'big data' on cloud computing: review and open research issues," *Information Systems*, vol. 47, pp. 98–115, 2015.

[45] G. Shpantzer, "Implementing hardware roots of trust: The trusted platform module comes of age," *SANS Whitepaper*, 2013. [Online]. Available: https://trustedcomputinggroup.org/wp-content/uploads/SANS-Implementing-Hardware-Roots-of-Trust.pdf.

[46] S. A. Mirheidari, S. Arshad, and R. Jalili, "Alert Correlation Algorithms: A Survey and Taxonomy," in *Cyberspace Safety and Security*, vol. 8300 of *Lecture Notes in Computer Science*, pp. 183–197, Springer International Publishing, Cham, 2013.

[47] P. Cichonski, T. Millar, T. Grance, and K. Scarfone, "Computer Security Incident Handling Guide : Recommendations of the National Institute of Standards and Technology," National Institute of Standards and Technology NIST SP 800-61r2, 2012.

[48] D. Díaz-López, G. Dólera-Tormo, F. Gómez-Mármol, and G. Martínez-Pérez, "Dynamic counter-measures for risk-based access control systems: An evolutive approach," *Future Generation Computer Systems*, vol. 55, pp. 321–335, 2016.

[49] P. Nespoli, D. Papamartzivanos, F. G. Marmol, and G. Kambourakis, "Optimal countermeasures selection against cyber attacks: A comprehensive survey on reaction frameworks," *IEEE Communications Surveys & Tutorials*, 2017.

[50] P. H. Gregory, *ISSP guide to security essentials*, vol. 12, Cengage Learning, 2014.

[51] Alienvault, "Insider's guide to Incident Response," https://www.alienvault.com/resource-center/ebook/insider-guide-to-incident-response-download, 2017.

[52] E. Tittle, J. M. Stewart, and M. Chapple, *CISSP: Certified Information Systems Security Professional Study Guide*, vol. 7, John Wiley Sons, 2012.

[53] S. Perumal, N. Md Norwawi, and V. Raman, "Internet of Things(IoT) digital forensic investigation model: Top-down forensic approach methodology," in *Proceedings of the 5th*

International Conference on Digital Information Processing and Communications, ICDIPC 2015, pp. 19–23, Switzerland, October 2015.

[54] RSA, "Two-Factor Authentication Is a Must for Mobile," 2016. [Online]. Available: https://www.rsa.com/en-us/blog/2016-06/two-factor-authentication-is-a-must-for-mobile.

[55] A. Soro, A. H. Ambe, and M. Brereton, "Minding the gap: Reconciling human and technical perspectives on the IoT for healthy ageing," *Wireless Communications and Mobile Computing*, vol. 2017, 2017.