10-2019

ESDRA: An efficient and secure distributed remote attestation scheme for IoT swarms

Boyu KUANG

Anmin FU

Shui YU

Guomin YANG
Singapore Management University, gmyang@smu.edu.sg

Mang SU

See next page for additional authors

Citation

KUANG, Boyu; FU, Anmin; YU, Shui; YANG, Guomin; SU, Mang; and ZHANG, Yuqing. ESDRA: An efficient and secure distributed remote attestation scheme for IoT swarms. (2019). *IEEE Internet of Things Journal*. 6, (5), 8372-8383.
Available at: https://ink.library.smu.edu.sg/sis_research/7327

Author

Boyu KUANG, Anmin FU, Shui YU, Guomin YANG, Mang SU, and Yuqing ZHANG

ESDRA: An Efficient and Secure Distributed Remote Attestation Scheme for IoT Swarms

Boyu Kuang, Anmin Fu, *Member, IEEE*, Shui Yu, *Senior Member, IEEE*, Guomin Yang, *Senior Member, IEEE*, Mang Su, and Yuqing Zhang

***Abstract*—An Internet of Things (IoT) system generally contains thousands of heterogeneous devices which often operate in swarms—large, dynamic, and self-organizing networks. Remote attestation is an important cornerstone for the security of these IoT swarms, as it ensures the software integrity of swarm devices and protects them from attacks. However, current attestation schemes suffer from single point of failure verifier. In this paper, we propose an Efficient and Secure Distributed Remote Attestation (ESDRA) scheme for IoT swarms. We present the first many-to-one attestation scheme for device swarms, which reduces the possibility of single point of failure verifier. Moreover, we utilize distributed attestation to verify the integrity of each node and apply accusation mechanism to report the invaded nodes, which makes ESDRA much easier to feedback the certain compromised nodes and reduces the run-time of attestation. We analyze the security of ESDRA and do some simulation experiments to show its practicality and efficiency. Especially, ESDRA can significantly reduce the attestation time and has a better performance in the energy consumption comparing with list-based attestation schemes.**

***Index Terms*—Remote attestation, reputation management, single point of failure, swarms.**

I. Introduction

INTERNET of Things (IoT) devices have been widely used in industrial production, medical, household, smart office, city construction, and daily wear [1]–[3]. Nowadays, many IoT systems contain numerous heterogeneous devices which often operate in swarms—large, dynamic, and self-organizing networks. For example, smart factory control systems where numerous connected IoT devices work together to monitor and control critical processes [4], and self-organizing dynamic networks where a large number of devices form collectively intelligent systems (e.g., robot used for oil or gas search) [5]. Different from other computing devices, IoT devices are often limited in energy and size [6], [7]. However, it is still important to ensure the security of those devices and protect them from attacks [8]. Because those devices often store a large amount of users' private information and affect the operation of the entire system [9]–[11]. Usually, a compromised node may lead to the collapse of the entire network in a contagious manner. Besides, there are a lot of attacks against the normal operation of the devices [12], [13]. Therefore, how to ensure the integrity (or correctness) of software running on various IoT devices becomes a core security issue that needs to be solved urgently [14]–[16].

Remote attestation is a quite suitable way that allows a trusted entity called *verifier* to remotely verify the software integrity of other entities called *provers*.

Since remote attestation can provide many advantages, including flexibility, scalability, implementation simplicity, and cost savings, many remote attestation methods [17]–[26] have been proposed. However, those traditional remote attestation schemes are not fully applicable to the environment of IoT because of the size and features of swarms. On the one hand, swarms contain a large number of devices. Traditional remote attestation schemes can only attest all the *provers* one by one, rather than simultaneously. Therefore, they cost a great deal of time for swarm attestation. On the other hand, the devices in swarms are usually heterogeneous in hardware and software structure, but the traditional remote attestation methods can only be used for the same type of device [27].

Therefore, in 2015 Asokan *et al.* [27] proposed the first IoT swarms attestation scheme SEDA. SEDA makes overall attestation for the whole swarm, rather than individually attesting each device. It determined the integrity of swarm by spanning tree and recursive calculations. Afterwards, Ambrosin *et al.* [28] proposed SANA based on SEDA which improved the security of the scheme by untrusted aggregator and public verifiability. In addition, Ibrahim *et al.* [29] proposed a noninteractive attestation scheme for denial-of-service (DoS) attacks, called SeED. It allowed the *provers* to periodically trigger attestation, reducing the time and energy consumption. But noninteractive attestation protocols needed

Manuscript received December 31, 2018; revised March 27, 2019 and April 29, 2019; accepted May 11, 2019. Date of publication May 16, 2019; date of current version October 8, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61572255, Grant U1836210, Grant 61572460, and Grant 61702266, in part by the National Key Research and Development Program of China under Grant 2016YFB0800703, and in part by the Six Talent Peaks Project of Jiangsu Province, China, under Grant XYDXXJS-032. *(Corresponding author: Anmin Fu.)*

B. Kuang and M. Su are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: 516106001766@njust.edu.cn; sumang@njust.edu.cn).

A. Fu is with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China, and also with the School of Computing and Information Technology, Institute of Cybersecurity and Cryptology, University of Wollongong, Wollongong, NSW 2522, Australia (e-mail: fuam@njust.edu.cn).

S. Yu is with the School of Software, University of Technology Sydney, Sydney, NSW 2007, Australia (e-mail: shui.yu@uts.edu.au).

G. Yang is with the School of Computing and Information Technology, Institute of Cybersecurity and Cryptology, University of Wollongong, Wollongong, NSW 2522, Australia (e-mail: gyang@uow.edu.au).

Y. Zhang is with the National Computer Network Intrusion Protection Center, University of Chinese Academy of Sciences, Beijing 100000, China (e-mail: zhangyq@ucas.ac.cn).

Digital Object Identifier 10.1109/JIOT.2019.2917223

synchronized clocks for all devices. Gong *et al.* [30] presented a formal description of senor nodes and proposed a remote attestation mechanism, which can track nodes in real time. Carpent *et al.* [31] proposed a new metric of swarm attestation called QoSA and two attestation schemes LISAa and LISAb.

In 2018, SALAD [32] focused on attestation of dynamic networks. It utilized the self-attestation to propagate and accumulate attestation results in the swarm, enabling attestation no longer limited by network topology. But SALAD required that any two devices in the swarm share a unique symmetric key, which incurred inestimable consumption.

Motivation: The existing swarm attestation schemes are all based on one-to-one [17]–[19] or one-to-many [27]–[32] models, i.e., a single *verifier* attests a single or multiple *provers*. They ensure swarm security through periodical attestation. Unfortunately, they still have some drawbacks. First, as the *verifier* is fixed, the attackers can easily crash the entire swarm by compromising the *verifier*, so-called single point of failure verifier. Furthermore, since current swarm attestation schemes attest all devices simultaneously, they should feedback the status of the whole swarms. But if they only collect the number of compromised and regular nodes, it is obviously not enough for the network owner to do some further operations. If they transfer specific information of each node, it introduces too much overhead. Moreover, there is usually multihop information transmission between *verifier* and *prover*, and each hop is affected by many factors, so it is difficult to ascertain the response time of attestation. On the one hand, it gives attackers a great chance to forge or modify the results of the validation. On the other hand, if one comprised node dose not return the attestation result or a result is lost due to network congestion or other reasons, the attestation time of the entire swarm will be prolonged [12].

Our Contributions: This paper proposes an Efficient and Secure Distributed Remote Attestation (ESDRA) scheme for IoT swarms. The main contributions of this paper are as follows.

1) We provide the first many-to-one attestation scheme for device swarms which eliminates the fixed *verifier*, reduces the possibility of single point of failure verifier, and also suits heterogeneous devices by trustworthy code certificates.
2) We propose a new device swarm attestation model that utilizes distributed attestation to verify the integrity of each node and applies accusation mechanism to report the invaded nodes. In this case, ESDRA is much easier to feedback the certain compromised nodes, apply to half-dynamic networks, and reduces the run-time of attestation.
3) We introduce the reputation mechanism into our scheme and utilize neighbor nodes to directly attest *provers*, which makes it much easier to estimate the time of attestation.
4) We demonstrate the security of our scheme. In addition, the simulation experiments indicate that ESDRA is practical and efficient. Especially, it can significantly reduce the attestation time and it has lower energy consumption comparing with other list-based attestation schemes.

Roadmap: The rest of this paper is organized as follows. Section II presents an overview of attestation schemes. Section III introduces the system model, objectives, and assumptions of our scheme. Section IV describes the detailed protocol. We analyze the security of ESDRA in Section V, and report the performance results in Section VI. Finally, Section VII concludes this paper.

II. Related Work

Many remote attestation methods have been proposed up to now. They can be divided into the following three categories: 1) software-based; 2) hardware-based; and 3) hybrid attestation model. Software-based remote attestation schemes [17]–[19] are generally based on strict time control and some strong but unreasonable assumptions, such as restricting attackers during attestation [27]. Thus, there are almost no software-based schemes available nowadays.

Hardware-based remote attestation schemes [20]–[22] are implemented on some secure hardware, such as trusted platform module (TPM) or physical unclonable function (PUF) hardware. But for some embedded devices and mobile devices, it is too expensive to conduct these schemes, which also results in the impracticality of hardware-based solutions. Therefore, there are a few hardware-based schemes in recent years. Wang *et al.* [33] proposed a secure attestation scheme base on the Intel software guard extension (SGX). Tan *et al.* [34] proposed MTRA for swarm attestation enabling the devices equipped with TPM to play more powerful roles in the attestation phase. In addition, Dessouky *et al.* [35] proposed a runtime attestation scheme LO-FAT in hardware, which attested the running process of *provers* based on the control flow graph (CFG) of programs. Noticing the memory bank attacks and the static integrity of devices, Zeitouni *et al.* [36] proposed another runtime attestation scheme ATRIUM.

Most of the current remote attestation schemes are based on a hybrid hardware and software setting. The basic idea of this attestation model is “minimum hardware overhead.” They only need the necessary basic hardware infrastructure. These schemes are always implemented on SMART [23], TrustLite [24], and TyTAN [25]. SMART is the first hybrid attestation architecture which mainly consists of a memory protection unit (MPU) that controls the read/write access to the storage area and a read-only memory (ROM) that stores some unrecoverable information. TrustLite is also an attestation platform for low-end embedded systems. It realizes the code isolation and the communication between protected modules by hardware. Besides, it ensures the integrity of the code and data by secure boot, which can verify the integrity of the device itself at the time of loading. Moreover, it utilizes the execution-aware MPU (EA-MPU) to provide a more secure read/write mechanism in memory access control, which not only considers the data access but also validates the current instruction pointer during execution. Furthermore, TyTAN is quite similar to TrustLite, except that it provides real-time guarantees and dynamic configuration for some important programs.

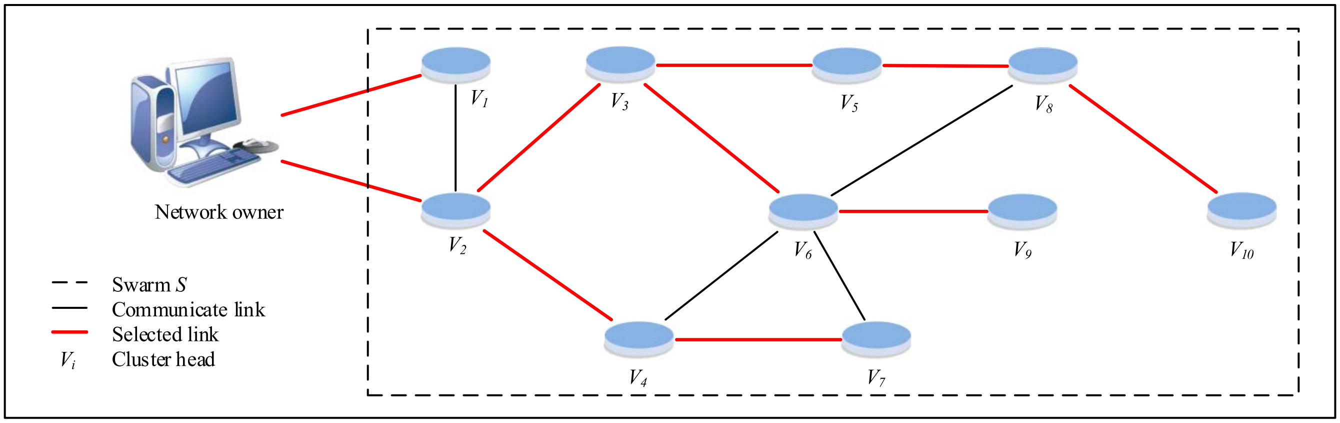


Fig. 1. Swarm attestation in a ten cluster head network.

SEDA [27] described its implementation based on SMART and TrustLite. SANA [28] introduced an aggregation signature technology to encrypt the attestation information and the transmitted integrity evidence to *verifier* through the spanning tree based on the implementation of TyTAN. SeED [29] was based on SMART and TrustLite, while it added a real-time clock to the basic architecture. Besides, DARPA [37], the first attestation schemes resilient to physical attacks, also realized its implementation based on SMART. It required devices to send heartbeat information periodically to prove their existence according to the assumption that devices would be offline for a period of time when a physical attack occurs.

In 2017, Eldefrawy *et al.* [38] proposed another hybrid attestation structure HYDRA based on verified seL4 microkernel, which ensures the memory isolation and controls the memory access. Then, Carpent *et al.* [39] proposed a lightweight attestation scheme defense against roving malware base on HYDRA.

In addition, some attestation schemes were based on other hybrid structures. C-FLAT [40], the first dynamic attestation withstanding runtime attacks, was based on Raspberry Pi 2. It recorded the real time running process through code segmentation and utilized the ARM TrustZone as the trusted execution environment (TEE). Kohnhäuser *et al.* proposed an attestation protocol SCAPI [41] to detect the physical attacks based on Stellaris EK-LM4F120XL microcontrollers (an embedded platform from Texas Instrument). SALAD [32] was also based on Stellaris LM4F120H5QR microcontrollers.

III. System Model and Assumption

A. System Model

A swarm S contains a large number of heterogeneous devices, which are usually mobile or embedded devices. In theory, any device can communicate with the nodes in its communication range, but the communication energy consumption will rise sharply with the increase of the communication distance dis. The relationship between them is presented as follows:

$$E = q \times dis^u \quad (1)$$

where q is a constant coefficient, which is determined by device models, and u is affected by many factors, such as obstacles and antenna quality [42]. Therefore, we divide S into several clusters according to the communication distance, reducing the communication within the cluster to an acceptable range. Note that majority of attestation networks [27], [29], [32] do not have head nodes, while few of the rest [30], [34] have a connected graph of head nodes. We build a network with cluster heads in this paper, because it facilitates the construction of distributed attestation and is further described in Section IV-C. As shown in Fig. 1, S consists of ten cluster heads. Each cluster head represents a cluster, which contains a large number of normal nodes. Considering the size of the graph, we do not illustrate these normal nodes in the diagram. In Section IV, we will introduce the structure within the cluster in detail.

B. Objectives

Taking into account the existing attacks, a secure and efficient attestation protocol for swarms must have the following properties.

1) *Authenticity and Freshness:* It is very important to ensure that the responses of the *provers* are true and trustworthy, as well as the responses follow a certain timeliness.
2) *Atomicity:* In order to prevent the attacker from modifying the attestation process or attestation results, the protocols cannot be interrupted.
3) *Unpredictability and Unforgeability:* The adversary neither can predict the specific information of the challenge in advance, nor can forge a response of a challenge.
4) *Heterogeneity:* The attestation scheme is suitable with different software and hardware.
5) *Scalability:* The attestation scheme is also suitable as the scale of the network grows.
6) *Determinism:* For different challenges, the *prover* cannot return the same response by the attestation procedure. A compromised node cannot be regarded as a benign node after attestation process.

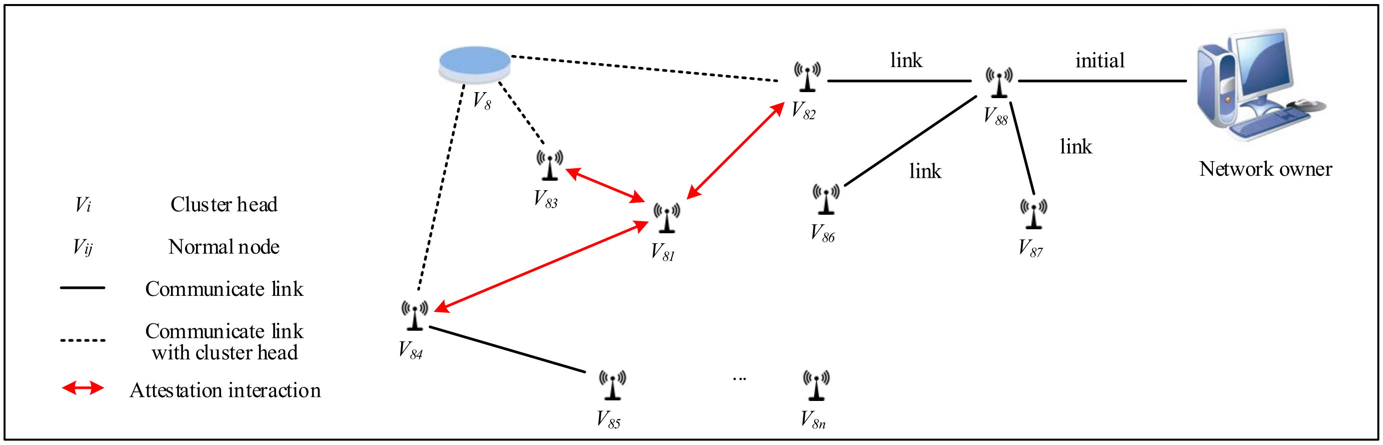


Fig. 2. Swarm attestation in a cluster.

C. Assumption

In our attestation scheme, we make four necessary assumptions.

1) We assume that all the nodes satisfy the minimal hardware requirements, such as ROM and MPU, which ensures that the attestation protocols are stored in a trusted and nontamperable space, the execution of the protocols cannot be interrupted, and the access of some important parameters are safely controlled.
2) Since our protocol is designed for swarms, we assume that each node has at least three neighbors to ensure the reliability of the attestation result.
3) Because the physical attacks are often expensive and difficult to detect, we only consider the software attacks like other swarm attestation schemes [27]–[32]. The adversary can modify the attestation result and evade the attestation protocol via various methods, such as replaying, forging, substituting, and eavesdropping.
4) Since DoS attacks are almost impossible to be completely resisted, we also rule out the DoS attacks like other schemes [27], [28], [30]–[32].

IV. Protocol Description

ESDRA is a many-to-one attestation scheme. It utilizes the neighbor devices to determine the *prover*'s integrity. Each compromised node is reported to the network owner O using accusation mechanism. ESDRA can be divided into four phases: 1) device initialization; 2) device linking; 3) network construction; and 4) device attestation. The device initialization phase generates the basic information needed by the device. Then, a node links its neighbors at device linking phase. The network construction phase sets up the network framework. The device attestation phase is the actual attestation. Table I summarizes the variables and parameters.

A. Device Initialization

As shown in Fig. 2, V_{88} is a new device that is introduced into the swarm. The network owner O executes an initialization protocol to generate some basic configuration information for the node, including private key sk_{88}, public key pk_{88}, identity certificate cert(pk_{88}), code certificate cert(h_{88}), initial credit value w_{88}, maximum attestation time t_{88}, the time of last attestation $t_{a\text{-}88}$, the valid time of the attestation $t_{e\text{-}88}$, and a unique device identifier d_{88}. In our scheme, we use the asymmetric cryptography based on elliptic curve cryptography (ECC) to generate sk and pk. Since we set the size of sk to 255 bits, pk which is a point on the curve could be represented by 256 bits using compressed representation [43]. The code certificate cert(h) contains a trustworthy hash code h of the device and some common information of certificates (e.g., the signature algorithm and the issuer), which provides a reliable criterion in attestation phase. At the time of $(t_a + t_e)$, the node will be attested immediately. In this way, we can control the number of concurrent attestation devices. Note that reputation mechanism is widely used in many networks of multiple nodes to evaluate their trustworthiness, where the reputations of nodes always depend on their previous behaviors (e.g., forwarding behaviors) [44]–[46]. In our schemes, the nodes' credit values depend on each attestation routine they are involved in. The reputation mechanism allows the more trusted nodes to play more important roles in the generation of final attestation results. But since a node do not have the authority to report reputation values for other, the attacks that only target the reputation mechanism cannot compromise the security of ESDRA, such as badmouthing, ballot-stuffing, topology-based, on-off attacks, and so on. In order to prevent some high-credit nodes from being invaded and unable to be discovered in time, it is necessary to set an upper limit of the credit value $w_{\max}$. It also keeps some nodes with high credit values from affecting the final attestation results. Once a node is considered to be untrustworthy, its credit value will be set to $-w_{\max}$. More formally

$$\begin{aligned}&\text{initial}\left(h_i, 1^{\ell}\right)\\&\quad\to (sk_i, pk_i, \text{cert}(pk_i), \text{cert}(h_i), w_i, t_i, t_{a-i}, t_{e-i}, d_i)\end{aligned} \quad (2)$$

where 1^{ℓ} indicates a bit string of length ℓ. Then, O will use its private key to generate incompressible noise to fill the blank program storage area, as shown in Fig. 3. Without noise padding [Fig. 3(a)], adversaries can easily store the original code in the blank area and do not change the hash value of the

TABLE I
VARIABLE AND PARAMETERS

Entities		
O	Network owner	
V_i	Cluster head of the ith cluster	
V_{ij}	Normal node of the ith cluster	
Swarm parameters		**Unit/Length**
s	Total number of devices in S	Integer
n	Total number of cluster heads in S	Integer
n_i	Total number of devices in cluster I	Integer
m_i	Total number of neighbors of the *prover*	Integer
Device parameters		
sk	Private key	255 bit
pk	Public key	256 bit
h	Platform software configuration	256 bit
$cert(pk_i\,/\,pk_{ij})$	Identify certificate of V_i or V_{ij} (issued by O)	-
$cert(h_i\,/\,h_{ij})$	Code certificate of V_i or V_{ij} (issued by O)	-
t_a	Last attestation time	String
t_e	Valid attestation time	String
k	Symmetric key	256 bit
w	Credit value	Float
t	Maximum attestation time	String
d	Unique device identifier	Integer
Protocol parameters		
b	Attestation result by neighbor	Integer
a	Nonce	256 bit
T	Threshold of credit value	Float
e	Confidence level	Float
g	Probability parameter	Float
f	Final attestation result	Integer
w_{max}	Maximum credit value	Float
w_{rwd}	Credit value of reward	Float
w_{pns}	Credit value of punishment	Float
Procedures		
HMAC()	Creating a HMAC	
VerMac()	Verification of a HMAC	
Attestation()	Attestation by neighbor nodes	
Judge()	Calculation of the final attestation result	
CreditCalculate()	Calculation of the *prover*'s new credit value	

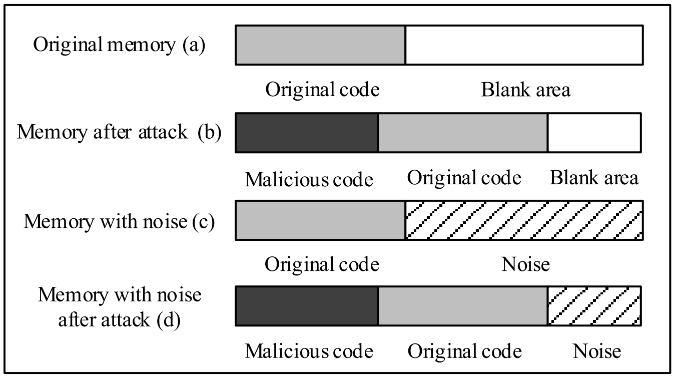


Fig. 3. Memory layout after and before attack.

program, thus having the opportunity to defraud the attestation program, as shown in Fig. 3(b). But if the blank memory area is filled with incompressible noise [Fig. 3(c)], attackers can only delete part of the noise area to store the original program code, which will inevitably change the hash value of the program and be detected by the attestation program, as shown in Fig. 3(d).

Remote attestation always follows a challenge-response mechanism. The *verifier* usually stores a trusted and expected memory content of the *prover*. In attestation process, the *verifier* sends a challenge to the *prover*. The *prover* will generate a response based on the challenge and its current memory status. At last, the *verifier* will determine the *prover*'s software integrity via comparing the response with the expected memory content. In order to save communication and storage overhead of *verifier*, instead of transferring the entire memory content, our remote attestation scheme transmits a code checksum (e.g., hash value) of it. Usually, the memory we detected is the program memory, as the data memory is nonexecutable and unpredictable.

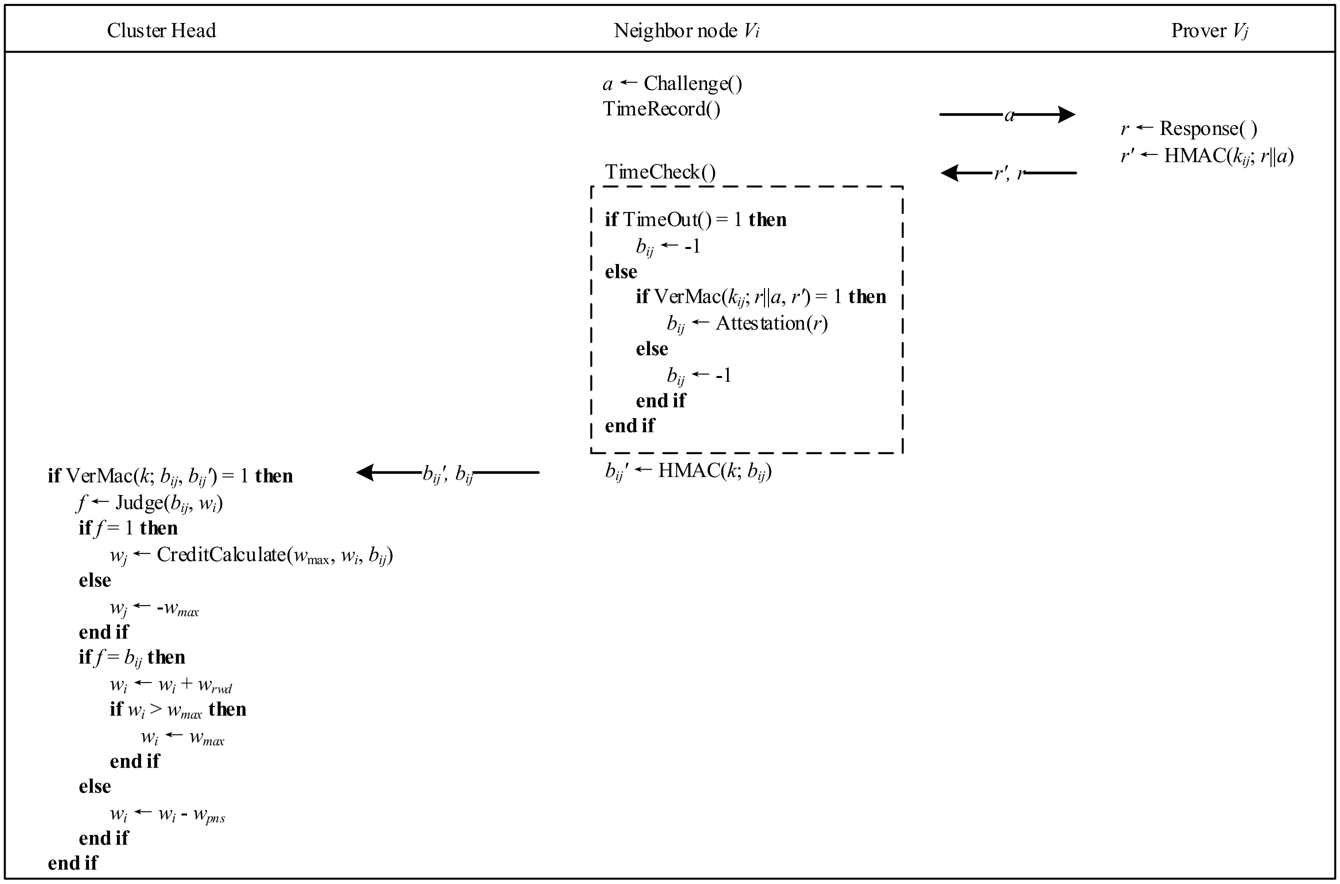
B. Device Linking

After the device V_{ij} is initialized or updated, it will execute the link protocol to complete basic information exchange with neighbor nodes. As shown in Fig. 2, V_{88} executes the linking protocol with V_{82}, V_{86}, and V_{87}. They exchange each other's code certificate cert(h_i), identity certificate cert(pk_i), device identifier d_i, maximum attestation time t_i, the last attestation time $t_{a\text{-}i}$, valid attestation time $t_{e\text{-}i}$, and current credit value w_i, which prevents the nodes that have been proven untrusted from rejoining the swarm. Note that the linking protocol and this information is protected by hardware, which means that a malicious node cannot report a bogus value here. A session key k is generated for subsequent communication based on the secret keys and the identity certificates, which can be achieved using an authenticated key agreement protocol. In our scheme, we use the ECC cofactor Diffie–Hellman (CDH) key-agreement schemes from NIST SP 800-56A [44]. In attestation phase, the temporary attestation results are calculated based on this exchanged information. In this way, the nodes can determine the integrity of heterogeneous neighbor devices, which guarantees the scalability of ESDRA. Formally,

$$\begin{aligned}&\mathrm{link}\big[V_i : sk_i, V_j : sk_j, * : \mathrm{cert}(pk_i), \mathrm{cert}\big(pk_j\big)\\ &\quad \mathrm{cert}(h_i), \mathrm{cert}\big(h_j\big), t_i, t_j, d_i, d_j, w_i, w_j\big]\\ &\quad \rightarrow \big[V_i : k_{ij}, V_j : k_{ij}\big].\end{aligned} \tag{3}$$

C. Network Construction

O periodically selects the nodes with highest credit value in each zone of swarm as the cluster heads. The number of cluster heads should be determined by the size of S. The distance between cluster heads should be taken into account in the selection process, making the cluster heads distribute as evenly as possible in S. It is noted that the cluster head selection algorithm is outside the scope of this paper since many sophisticated cluster head selection algorithms [48], [49] have



```
Neighbor node V_i:
$a \leftarrow$ Challenge()
TimeRecord()

TimeCheck()
if TimeOut() = 1 then
    $b_{ij} \leftarrow -1$
else
    if VerMac($k_{ij}$; $r\|a$, $r'$) = 1 then
        $b_{ij} \leftarrow$ Attestation($r$)
    else
        $b_{ij} \leftarrow -1$
    end if
end if
$b_{ij}' \leftarrow$ HMAC($k$; $b_{ij}$)
```

```
Prover V_j:
$r \leftarrow$ Response( )
$r' \leftarrow$ HMAC($k_{ij}$; $r\|a$)
```

```
Cluster Head:
if VerMac($k$; $b_{ij}$, $b_{ij}'$) = 1 then
    $f \leftarrow$ Judge($b_{ij}$, $w_i$)
    if $f = 1$ then
        $w_j \leftarrow$ CreditCalculate($w_{\max}$, $w_i$, $b_{ij}$)
    else
        $w_j \leftarrow -w_{max}$
    end if
    if $f = b_{ij}$ then
        $w_i \leftarrow w_i + w_{rwd}$
        if $w_i > w_{max}$ then
            $w_i \leftarrow w_{max}$
        end if
    else
        $w_i \leftarrow w_i - w_{pns}$
    end if
end if
```

Fig. 4. ESDRA protocol.

been proposed and all of them can be used in our mechanism with minor modifications.

The cluster heads form a tree structure with O through signal transmission [27], as shown in Fig. 1. The remaining nodes form clusters based on the proximity principle. In this phase, the normal nodes will send their cert(pk_i), current credit values w_i, and d_i to cluster heads as well as get cluster heads' cert(pk_i) and a session key k. Note that O holds the credit values of all nodes. After each attestation routine, O will receive the new credit values of the corresponding nodes. The credit values can indicate the health status of the deployed devices, since a node's credit value will be set to $-w_{\max}$ when it is detected to be untrustworthy.

In addition, when the cluster head discovers that a node is untrusted, it will send the node's w_i and d_i to O through the spanning tree. Hence, O can remove or maintain the corresponding node according to d_i. The spanning-tree network can more easily support parallel computing and save transmission costs at the same time. Besides, as we present in Section III, the energy costs of nodes will raise dramatically with the increase of communication distance. Therefore, we reduce energy consumption by using multihop information transmission and making the transfer distance of each hop within a certain range.

In particular, if the devices of a swarm are concentrated in a certain area, there may be only one cluster. Then, no multihop information transmission is needed.

D. Device Attestation

Once the credit value of a device is below the threshold T or the attestation time is up, the neighbor nodes will challenge it and record the challenge time immediately. As shown in Fig. 4, the challenge contains a nonce a, which can resist replay attacks effectively for its unpredictability. After generating the responses r, the *prover* first performs a predesigned calculation on a and r (e.g., a logical operation), and then encapsulates the output with the symmetric key to generate a message authentication code (MAC) r'. When receiving r and r' from the prover, the neighbor nodes verify the MAC and the response by comparing r with the knowledge obtained during the linking phase, and then generate temporary attestation results b. Note that MAC algorithms generally contain the hash-based MAC (HMAC) and cipher-based MAC (CMAC). In order to maintain the consistency of this paper, we use HMAC uniformly. If the response time exceeds the maximum attestation time t, the neighbor nodes will directly regard the *prover* as compromised. Formally,

$$\text{Attestation}\left[V_i : t_j, a, \text{cert}(h_j), k_{ij}, V_j : h_j', k_{ij}, * : -\right] \rightarrow \left[V_i : r, V_j : -\right]. \quad (4)$$

All temporary attestation results b will be aggregated to their corresponding cluster head. The cluster head calculates the final attestation result f according to each b and w. The formulas are as follows:

$$e = \frac{\sum_{i=0}^{n}\left(b_{ij} \times w_i\right)}{g \cdot \sum_{i=0}^{n} w_i} \tag{5}$$

$$f = \begin{cases} 1, & e \geq 1 \\ -1, & e < 1 \end{cases} \tag{6}$$

where the temporary attestation result b is equal to 1 or -1. If b equals 1, it indicates that the temporary attestation is successful, and otherwise representing that the neighbor node believes the *prover* is compromised. In addition, the parameter $g \in (0, 1]$ is a probability parameter denoting the evaluation criteria. If g approaches 0, it means the node is trustworthy in any case. If g equals 1, it means the software configuration of the *prover* is integrated only when all the temporary attestations success. As the value of g increases, the credibility of the result gets higher, but a node is more stringent to be able to pass the attestation. In the strict network environment and attestation process, we should set the value of g as 1. However, in a real network environment, the neighbor nodes may incorrectly judge the *prover*'s integrity due to network delay, network congestion, and some other external reasons. Therefore, we utilize g to increase the fault tolerance of protocols, thereby g should be chosen according to the probability of false positives and the strictness of the attestation procedure. The parameter e is the level of confidence of the *prover*. If e is equal to 1, the cluster head directly will set the final attestation result f to 1. Otherwise, the cluster head will calculate f using (6). Similar to the temporary attestation result b, if f equals 1, it indicates that the *prover* is trustworthy, otherwise incredible.

If $f = 1$, the cluster head node will recalculate the *prover*'s credit value according to (7), otherwise it will be set to $-w_{\max}$. Equation (7) guarantees the new credit value is larger than the threshold T, avoiding the redundant attestation. The parameter $w_{\max}$ is the maximum credit value

$$w = (w_{\max} - T) \cdot \frac{\sum_{i=0}^{n}\left(b_{ij} \times w_i\right)}{\sum_{i=0}^{n} w_i} + T. \tag{7}$$

After getting f, the cluster head will compare f with each b_i. If they are not equal, the cluster head node will reduce the credit value of the corresponding node. Otherwise, the node's credit value will be increased unless w is equal to $w_{\max}$.

Furthermore, the cluster head will send the device identifiers and the new credit values to O and all nodes in the cluster. If the final attestation fails, all the neighbors will disconnect from the *prover*. O could remove or maintain the corresponding device according to the identifier.

Remark: Generally, the swarm attestation schemes guarantee their scalability by self-measurement. In our ESDRA, every device holds a trustworthy code certificate cert(h_{ij}). After the device linking phase, neighbor nodes exchange their basic information, including the code certificate. In this way, the neighbor devices do not need to know the detailed information of *provers* and can verify their software integrity. Therefore, our scheme can be applied to various heterogeneous swarms devices.

As for the connectivity of nodes, since the attestation is time-constrained, most normal nodes do not affect the process of attestation when they are disconnected. For other key nodes (e.g., cluster heads), this problem can be solved by relinking or resending.

V. Security Analysis

The purpose of swarm attestation is to ensure that all devices in S are running under an untampered software environment. We formalize this process as a safety experiment $\mathrm{EXP}_{\mathcal{ADV}}$. An attacker $\mathcal{ADV}$ can interact with all the devices in S. $\mathcal{ADV}$ modifies the software configuration of at least one device V_i. $\mathcal{ADV}$ can tamper with, eavesdrop on or delete all the information transmitted via V_i. After a polynomial number (in terms of the security parameters ℓ_a, ℓ_{sign}, and ℓ_{hmac}) of steps, V_i is attested and a final attestation result f is generated by the cluster head. If $f = 1$, it indicates that the attestation is successful, otherwise the attestation fails. A secure swarm attestation scheme is defined as follows.

Definition 1 (Secure Swarm Attestation [27], [28]): F is a polynomial function of ℓ_a, ℓ_{sign}, and ℓ_{hmac}. If the probability of a compromised device passing the attestation $\Pr[f = 1|\mathrm{EXP}_{\mathcal{ADV}}(1^{\ell}) = b]$ is considered to be negligible in $\ell = F(\ell_a, \ell_{\text{sign}}, \ell_{\text{hmac}})$, the corresponding swarm attestation scheme is secure.

Theorem 1: If the underlying signature and HMAC scheme are selective forgery resistant, ESDRA is a secure swarm attestation scheme.

Proof: When the final attestation result $f_i = 1$, V_i is considered credible. For each temporary attestation, an attacker wants to make $b_{ji} = 1$ and $\mathrm{VerMac}(k_{ij}; r||a, r') = 1$. Thus, with aim to pass the attestation, the attacker may use the following strategies: 1) using a previous HMAC r'_{old} based on the original code; 2) forging a new challenge-based HMAC r'; and 3) invading several neighbor nodes to influence the final attestation result.

For the first case, according to Section IV, a is a new random value sent by the neighbor node and $r' = \mathrm{HMAC}(k_{ij}; r||a)$, so only when $a = a_{\text{old}}$, the attestation is successful. Note that HMAC scheme is selective forgery resistant. Therefore, the probability of $a = a_{\text{old}}$ is $2^{-\ell_a}$, where ℓ_a represents the length of nonce a. Thus, the temporary attestation result b is highly unlikely equal to 1. Also, f_i can hardly be equal to 1.

For the second case, r represents the current software configuration of the node. Hence, the neighbor nodes can easily identify that r' is different from the code hash in cert(h). The attestation cannot success.

For the third case, as we discuss above, the probability that a single compromised node does not be detected is $\Pr[f = 1|\mathrm{EXP}_{\mathcal{ADV}}(1^{\ell}) = b]$. So the probability that n compromised neighbor nodes do not be detected satisfies $\Pr|E| \leq (\Pr[f = 1|\mathrm{EXP}_{\mathcal{ADV}}(1^{\ell}) = b])^n$. For the probability $\Pr[f = 1|\mathrm{EXP}_{\mathcal{ADV}}(1^{\ell}) = b]$ is negligible, $\Pr|E|$ tends to 0, which indicates that ESDRA can effectively resist collusion attacks.

Therefore, we can conclude that for ℓ_a, ℓ_{sign}, and ℓ_{hmac}, the probability of $f = 1$ is negligible, where f is the final attestation result of a comprised device. According to Definition 1, ESDRA is a secure swarm attestation scheme. ■

TABLE II
COMPARISON OF DIFFERENT SCHEMES

Scheme	Interaction Pattern	Comprised nodes	Timing	Topology	Run-time(s)	Average Energy(J)	Result
SEDA	One2many	Collection	Loose	Static	$t_{total} \leq (448 + 264Ht + \sum_{i=0}^{Ht} m_i)t_{tr} + (1 + Ht)t_{ca}$ $(2 + 4Ht + \sum_{i=0}^{Ht} m_i)t_{\text{hmac}} + (1 + Ht)t_{prng} + t_{sign}$	$E \leq (80 + 104m_i)E_{\text{send}} + (104 + 80m_i)E_{\text{recv}}$ $+ m_i E_{prng} + (3 + 3m_i)E_{\text{hmac}}$	Count
SeED	One2many	Collection	Loose	Static	$t_{total} \leq (320 + 160Ht)t_{tr} + (1 + Ht)t_{ca}$ $+ (1 + 2Ht)t_{\text{hmac}} + (1 + Ht)t_{prng}$	$E \leq 160E_{\text{send}} + 160m_i E_{\text{recv}} + E_{prng} + E_{\text{hmac}}$	Count
SALAD	One2many	Collection	Loose	Dynamic	$t_{total} \leq ((132 + 36n)Ht + \sum_{i=0}^{Ht} m_i')t_{tr} + (1 + Ht)t_{ca}$ $(m_i' + 2m_i'Ht + \sum_{i=0}^{Ht} m_i')t_{\text{hmac}} + t_{prng}$	$E \leq 100m_i'E_{\text{send}} + (36n + 32)m_i'E_{\text{recv}}$ $+ E_{prng} + 2m_i'E_{\text{hmac}}$	List
ESDRA	Many2one	Accusation	Strict	Half-dynamic	$t_{total} \leq (72 + (8m_i + 8)Ht)t_{tr} + (2 + Ht)t_{ca}$ $+ (2 + 2n_i + 4m_i)t_{\text{hmac}} + t_{prng}$	$E \leq 132E_{\text{send}} + (176 + 8m_i + 136 / m_i)E_{\text{recv}}$ $+ E_{prng} + (5 + 1 / m_i)E_{\text{hmac}}$	List

VI. PERFORMANCE EVALUATION

In this section, we evaluate the efficiency of our ESDRA through theoretical analyzes, which includes five aspects: 1) computation cost; 2) communication cost; 3) memory cost; 4) run-time; and 5) energy cost. Moreover, we simulate our scheme by comparing with the state-of-the-art schemes [27], [29], [32]. Following previous work [27]–[29], we only consider the costs in attestation phase. This is because the frequency of other phases is quiet low, and they may only execute once. But the attestation phase is periodically repeated. Moreover, usually, according to SEDA [27], these phases are performed offline and the cost of initialization phase comes mainly from the network owner O with nonlimitation of energy. Thus, the main overhead of devices is generated during the attestation phase.

A. Theoretical Evaluation

Computation Cost: The main computational overhead of nodes lies in some cryptographic operations. During attestation phase, the *prover* needs to compute m_i HMACs and verify one HMAC, where m_i indicates the number of neighbor nodes. Neighbor nodes verify two HMACs and compute one HMAC. The cluster head verifies m_i HMACs and calculate $(n_i + 1)$ HMACs, where n_i is the number of normal nodes in cluster, since the cluster head also needs to send the t_a, t_e, the new w_i and the corresponding d_i to intracluster nodes as well as O or the cluster head at the upper level.

Communication Cost: We implement HMAC with HMAC_SHA-256, the key agreement protocol with ECC CDH, and the signature scheme with ECDSA. In addition, we use the X.509 certificates in our scheme. We utilize l_x to denote the length of a bit string x. Thus, $\ell_{\text{hmac}} = 256$ and $\ell_{\text{sign}} = 512$. Besides, $\ell_a = 256$, $\ell_r = 256$, $\ell_w = 32$, $\ell_d = 32$, $\ell_t = 32$, and $\ell_b = 32$. It means that the credit value, the maximum attestation time, the temporary attestation result, and the device identifier are four bytes. Signature is 64 bytes. Nonce, responses, signing, and verification keys are 32 bytes. Since the size of the certificate is mainly determined by the size of key and signature, we use $32 + 64 = 96$ bytes to approximate the size of certificate. Therefore, the *prover* needs to receive $(136 + 8m_i)$ bytes and send $64m_i$ bytes. Neighbor nodes need to receive at most $(168 + 8m_i)$ bytes and send 68 bytes. The cluster head receives $36m_i$ bytes and sends at most $(104 + 8m_i)(n_i + 1)$ bytes.

Memory Cost: Each normal node needs to store at least: 1) its own key pair (sk, pk) and device identifier d; 2) the neighbor nodes' device identifiers d_i with their corresponding credit values w_i, maximum attestation times t_i, the time of last attestation t_a, the valid time of the attestation t_e, code certificates cert(h_i), and session keys k_i; and 3) session key with cluster head. The cluster head node also needs to store the device identifiers of all nodes in the cluster, the corresponding credit values and session keys. Hence, a normal node requires $(100 + 204m_i)$ bytes storage space and the cluster head node needs $(100 + 164m_i + 40n_i)$ bytes storage space.

Run-Time: We use t_{hmac}, t_{prng}, t_{ca}, and t_{tr} to denote the time to compute or verify an HMAC, generate a 32-bit random number, access a channel in one hop, and transmit one byte. Ht is the height of the spanning tree of the swarm. Thus, the total attestation time t_{total} is

$$t_{\text{total}} \leq (72 + (8m_i + 8)Ht)t_{tr} + (2 + Ht)t_{ca} + (2n_i + 4m_i + 2)t_{\text{hmac}} + t_{prng}. \quad (8)$$

Energy Cost: We use E_{send}, E_{recv}, E_{prng}, and E_{hmac} to, respectively, indicate the energy cost of sending one byte, receiving one byte, generating a 20-byte random number, and computing or verifying an HMAC. In an attestation process, the maximum energy consumption of the cluster head node is

$$E \leq (104 + 8m_i)(n_i + 1)E_{\text{send}} + 36m_i E_{\text{recv}} + (m_i + n_i + 1)E_{\text{hmac}}. \quad (9)$$

The maximum energy consumption of the *prover* is

$$E_i \leq 64m_i E_{\text{send}} + (168 + 8m_i)E_{\text{recv}} + (2m_i + 1)E_{\text{hmac}}. \quad (10)$$

The maximum energy cost of neighbor nodes is

$$E_j \leq 68E_{\text{send}} + (136 + 8m_i)E_{\text{recv}} + E_{prng} + 3E_{\text{hmac}}. \quad (11)$$

Functionality: As shown in Table II, ESDRA is a many-to-one swarm attestation scheme which eliminates the fixed *verifier*, reducing the possibility of single point of failure verifier. Besides, current swarm attestation schemes obtain the information of comprised nodes via collective mechanism. ESDRA utilizes an accusation mechanism which is much easier to feedback the certain compromised nodes. The *prover* in ESDRA is directly verified by its neighbors, thus it is not hard

TABLE III
RUN-TIME OF CRYPTOGRAPHIC FUNCTIONS

Function	Run-time (24 MHz) TrustLite [21] (ms)	
	Creat	Verify
HMAC(SHA-256)	0.4	0.4
ECDSA	347.2	-
PRNG(32 Bytes)	5.0	-

TABLE IV
ENERGY COST OF DIFFERENT FUNCTIONS

Function	Energy Cost
Transmit 1 bit	2.1 μJ
Receive 1 bit	2.3 μJ
Hmac()	0.2 mJ
VerMac()	0.2 mJ
PRNG(32 Bytes)	2.5 mJ
ECDSA sign	104 mJ
ECDSA verify	126 mJ

to estimate the attestation time. In other word, we can ensure that the attack time of the adversary is limited. As for the topology of the network, although ESDRA cannot be applied to highly dynamic networks, it also does not need to guarantee the whole network static at attestation phase, which is further discussed in Section VI-B. Furthermore, ESDRA has a huge advantage in run-time over existing protocols. Finally, the energy consumption of ESDRA is higher than that of count-based attestation schemes which only obtain the number of compromised devices in the swarm and do not present the specific state of each device. But ESDRA's energy consumption is lower than most of other list-based attestation proposals. Note that m_i' is average number of nodes in each device's communication range.

B. Simulation

We utilize the OMNeT++ [50] simulation tool to simulate run-time and energy consumption of ESDRA comparing with the SEDA [27], SeED [29], and SALAD [32]. As other schemes [27]–[29], we require the nodes in attestation procedure stay static and can only move at free time, so the dynamic of the network does not affect the overhead of devices and we simulate a static network. We implement our protocol at the application layer. Because most of the IoT devices are low-end mobile devices or embedded devices, and traditional secure components, like TPM, are too expensive and redundant in general, we prefer to use the lightweight architecture, i.e., TrustLite [24], in our simulation. We use delays to simulate the cryptographic operations in TrustLite. Table III presents the simulation setup parameters.

The assessment of energy costs is based on Raspberry Pi device with ARM architecture. Table IV introduces the parameters for simulation setup. Besides, we simulate 500 nodes in each cluster. We set the initial credit value to 3, the threshold T to 1, the credit value of reward w_{rwd} to 1, the credit value of punishment w_{pns} to 2, the credit value $w \in [0, 5]$, and the probability parameter g to 0.8. Note that the choice for reward/punishment values is important, which should be based on the actual application scenario, the initial credit value, the threshold T, as well as the credit value range. If the reward value is too large or the punishment value is too small, the comprised device may not be detected in time.

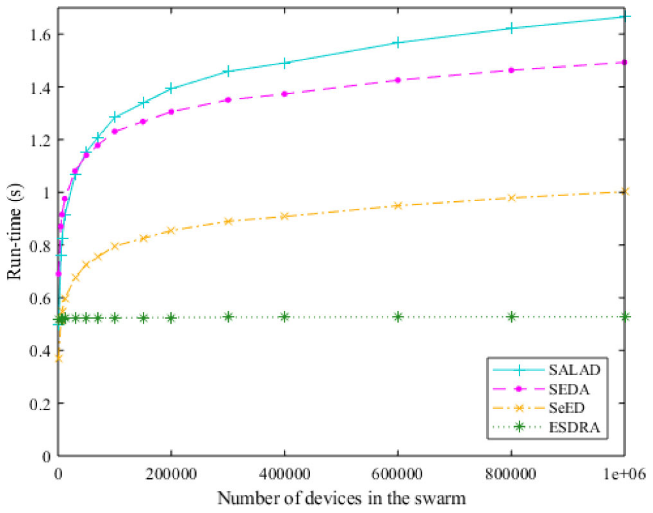


Fig. 5. Comparison of run-time for different number of devices.

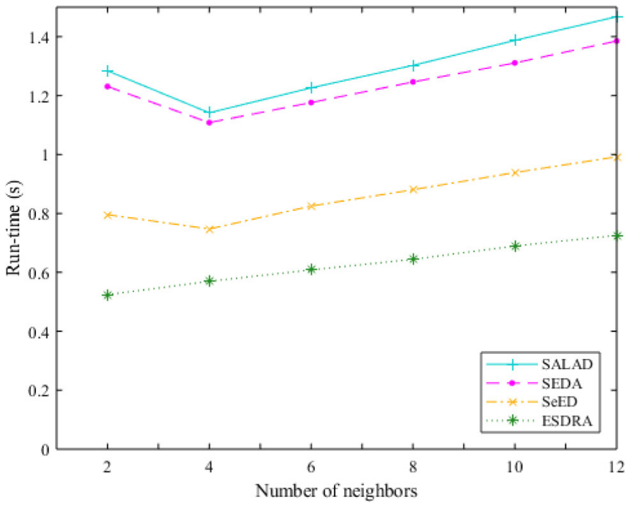


Fig. 6. Comparison of run-time with varying number of neighbors.

Fig. 5 shows the run-time for different number of devices by comparing ESDRA with SALAD, SEDA, and SeED. It is obvious that ESDRA has a significant advantage. Because other schemes require the up-level nodes to wait for the attestation result of the low-level nodes, the attestation time of them increases with the size of swarms. Conversely, the distributed attestation of each node in ESDRA executes independently and every attestation result is calculated in parallel. Besides, other swarm attestation solutions are more susceptible to the size of the size of swarm, since they need to compute and verify an HMAC for constantly collecting and updating the nodes' information during the information transmission of each hop. Conversely, ESDRA applies accusation mechanism to report the invaded nodes. The intermediate node only forwards information and does not perform other operations. Therefore, the size of swarm has less impact on our solution than other schemes.

Fig. 6 illustrates the run-time of SALAD, SEDA, SeED, and ESDRA with different number of neighbors when the

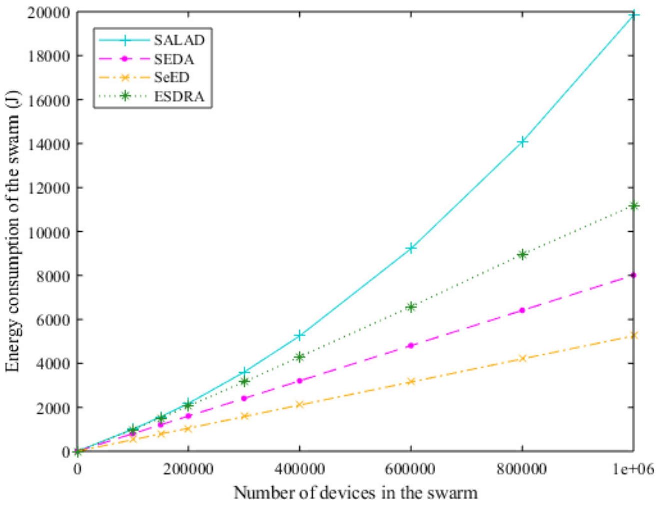


Fig. 7. Comparison of energy consumption.

total number of the swarm is 100 000. It is evident that our ESDRA shows obvious superiority over the run-time. In addition, SALAD, SEDA, and SeED present an obvious threshold when there are four neighbors for each device because it is affected by two factors. One is the depth of spanning tree and another is the runtime for individual devices. But the depth of spanning tree depends on the size of swarm which has limited influence on ESDRA as we discussed before.

Fig. 7 displays the energy consumption of our ESDRA compared with SALAD, SEDA, and SeED. As shown in Fig. 7, the energy consumption of ESDRA is larger than SEDA and SeED, since SEDA and SeED are count-based attestation schemes that only feedback the number of compromised nodes in the swarm. SeED is a noninteractive attestation protocol, thus its energy consumption is minimal. But it needs synchronized clocks for each device that bring more hardware overhead for all devices. ESDRA and SALAD are list-based attestation schemes. The network owner holds the specific status of each node. It is obvious that ESDRA's performance in energy consumption is better and the energy consumption of SALAD is more susceptible to the scale of the swarm, because each device in SALAD needs to forward the status list of the entire swarm. The more devices there are, the more information be conveyed, and the more overhead will be. Conversely, the devices in ESDRA only forward some information of *prover* and neighbors.

Moreover, the advantage of ESDRA in terms of attestation time indicates its adaptability to the dynamic topology. Most swarm attestation schemes [27]–[29] assume that the entire network topology remaining static during attestation phase. SALAD [29] proposed an attestation method which adapts to highly dynamic topologies. However, it incurs too much energy and memory cost, especially as it must maintain a unique channel key for every two devices in the swarm. ESDRA provides a new solution to these problems. Each attestation of ESDRA is limited to a smaller area and can be completed in a short time. Thus, it only needs to guarantee the connection of partial nodes at attestation phase.

VII. Conclusion

In this paper, we propose a new attestation scheme. We design the first many-to-one swarm attestation scheme to eliminate the fixed verifier and reduce the possibility of single point of failure verifier. Moreover, the distributed attestation mode makes ESDRA much easier to feedback the certain compromised nodes and apply to half-dynamic networks, dramatically reduces the run-time cost comparing with other swarm attestation methods, and reduces the energy consumption comparing with other list-based attestation schemes. In addition, we take advantage of the strict time control and reputation mechanism to enhance the security of our protocol. The multihop tree network structure and cluster design also make a great contribution to the efficiency of our scheme.

References

[1] D. Minoli, K. Sohraby, and B. Occhiogrosso, "IoT considerations, requirements, and architectures for smart buildings—Energy optimization and next generation building management systems," *IEEE Internet Things J.*, vol. 4, no. 1, pp. 269–283, Feb. 2017.

[2] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for smart cities," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 22–32, Feb. 2014.

[3] Y. Yu, Y. Li, J. Tian, and J. Liu, "Blockchain-based solutions to security and privacy issues in the Internet of Things," *IEEE Wireless Commun.*, vol. 25, no. 6, pp. 12–18, Dec. 2018.

[4] S. Wang, J. Wan, D. Zhang, D. Li, and C. Zhang, "Towards smart factory for Industry 4.0: A self-organized multi-agent system with big data based feedback and coordination," *Comput. Netw.*, vol. 101, pp. 158–168, Jun. 2016.

[5] M. Rubenstein, A. Cornejo, and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm," *Science*, vol. 345, no. 6198, pp. 795–799, Aug. 2014.

[6] Z. Sheng, S. Yang, Y. Yu, A. V. Vasilakos, J. A. Mccann, and K. K. Leung, "A survey on the IETF protocol suite for the Internet of Things: Standards, challenges, and opportunities," *IEEE Wireless Commun.*, vol. 20, no. 6, pp. 91–98, Dec. 2013.

[7] Y. Yu, L. Xue, Y. Li, X. Du, M. Guizani, and B. Yang, "Assured data deletion with fine-grained access control for fog-based industrial applications," *IEEE Trans. Ind. Inf.*, vol. 14, no. 10, pp. 4538–4547, Oct. 2018.

[8] A. Fu, J. Song, S. Li, G. Zhang, and Y. Zhang, "A privacy-preserving group authentication protocol for machine type communication in LTE/LTE—A networks," *Security Commun. Netw.*, vol. 9, no. 13, pp. 2002–2014, Sep. 2016.

[9] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the Internet of Things: Perspectives and challenges," *Wireless Netw.*, vol. 20, no. 8, pp. 2481–2501, Nov. 2014.

[10] A. Fu, S. Yu, Y. Zhang, H. Wang, and C. Huang, "NPP: A new privacy-aware public auditing scheme for cloud data sharing with group users," *IEEE Trans. Big Data*, to be published. doi: 10.1109/TBDATA.2017.2701347.

[11] A. Fu, N. Qin, Y. Wang, Q. Li, and G. Zhang, "Nframe: A privacy-preserving with non-frameability handover authentication protocol based on (t, n) secret sharing for LTE/LTE—A networks," *Wireless Netw.*, vol. 23, no. 7, pp. 2165–2176, Oct. 2017.

[12] R. V. Steiner and E. Lupu, "Attestation in wireless sensor networks: A survey," *ACM Comput. Surveys*, vol. 49, no. 3, pp. 51–82, Dec. 2016.

[13] A. Costin, J. Zaddach, A. Francillon, and D. Balzarotti, "A large-scale analysis of the security of embedded firmwares," in *Proc. USENIX Security*, San Diego, CA, USA, 2014, pp. 95–110.

[14] H. Ma, L. Liu, A. Zhou, and D. Zhao, "On networking of Internet of Things: Explorations and challenges," *IEEE Internet Things J.*, vol. 3, no. 4, pp. 441–452, Aug. 2016.

[15] A. Fu, Y. Li, S. Yu, Y. Yu, and G. Zhang, "DIPOR: An IDA-based dynamic proof of retrievability scheme for cloud storage systems," *J. Netw. Comput. Appl.*, vol. 104, pp. 97–106, Feb. 2018.

[16] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao, "A survey on security and privacy issues in Internet-of-Things," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1250–1258, Oct. 2017.

[17] A. Seshadri, A. Perrig, L. V. Doorn, and P. Khosla, "'SWATT: Software-based attestation for embedded devices," in *Proc. IEEE S&P*, Berkeley, CA, USA, 2004, pp. 272–282.

[18] D. Zhang and D. Liu, "DataGuard: Dynamic data attestation in wireless sensor networks," in *Proc. IEEE/IFIP Int. Conf. DSN*, Chicago, IL, USA, 2010, pp. 261–270.

[19] S. Kiyomoto and Y. Miyake, "Lightweight attestation scheme for wireless sensor network," *Int. J. Security Appl.*, vol. 8, no. 2, pp. 25–40, Mar. 2014.

[20] J. Kong, F. Koushanfar, P. K. Pendyala, A. R. Sadeghi, and C. Wachsmann, "PUFatt: Embedded platform attestation based on novel processor-based PUFs," in *Proc. 51st DAC*, San Francisco, CA, USA, 2014, pp. 1–6.

[21] H. Tan, W. Hu, and S. Jha, "A remote attestation protocol with trusted platform modules (TPMs) in wireless sensor networks," *Security Commun. Netw.*, vol. 8, no. 13, pp. 2171–2188, Sep. 2015.

[22] X. Kovah, C. Kallenberg, C. Weathers, A. Herzog, M. Albin, and J. Butterworth, "New results for timing-based attestation," in *Proc. IEEE S&P*, San Francisco, CA, USA, 2012, pp. 239–253.

[23] K. Eldefrawy, G. Tsudik, A. Francillon, and D. Perito, "SMART: Secure and minimal architecture for (establishing dynamic) root of trust," in *Proc. NDSS*, San Diego, CA, USA, 2012, pp. 1–15.

[24] P. Koeberl, S. Schulz, A.-R. Sadeghi, and V. Varadharajan, "TrustLite: A security architecture for tiny embedded devices," in *Proc. 9th EuroSys*, Amsterdam, The Netherlands, 2014, pp. 1–14.

[25] F. Brasser, B. E. Mahjoub, A. R. Sadeghi, C. Wachsmann, and P. Koeberl, "TyTAN: Tiny trust anchor for tiny devices," in *Proc. 52nd DAC*, San Francisco, CA, USA, 2015, pp. 1–6.

[26] Y. Yang, X. Wang, S. Zhu, and G. Cao, "Distributed software-based attestation for node compromise detection in sensor networks," in *Proc. 26th IEEE SRDS*, Beijing, China, 2007, pp. 219–230.

[27] N. Asokan *et al.*, "SEDA: Scalable embedded device attestation," in *Proc. 22nd ACM CCS*, Denver, CO, USA, 2015, pp. 964–975.

[28] M. Ambrosin, M. Conti, A. Ibrahim, G. Neven, A.-R. Sadeghi, and M. Schunter, "SANA: Secure and scalable aggregate network attestation," in *Proc. 23rd ACM CCS*, Vienna, Austria, 2016, pp. 731–742.

[29] A. Ibrahim, A. R. Sadeghi, and S. Zeitouni, "SeED: Secure non-interactive attestation for embedded devices," in *Proc. 10th ACM WiSec*, Boston, MA, USA, 2017, pp. 64–74.

[30] B. Gong, Y. Zhang, and Y. Wang, "A remote attestation mechanism for the sensing layer nodes of the Internet of Things," *Future Gener. Comput. Syst.*, vol. 78, no. 3, pp. 867–886, Jan. 2018.

[31] X. Carpent, K. Eldefrawy, N. Rattanavipanon, and G. Tsudik, "Lightweight swarm attestation: A tale of two LISA-s," in *Proc. ASIACCS*, Abu Dhabi, UAE, 2017, pp. 84–100.

[32] F. Kohnhäuser, N. Büscher, and S. Katzenbeisser, "SALAD: Secure and lightweight attestation of highly dynamic and disruptive networks," in *Proc. ASIACCS*, Incheon, South Korea, 2018, pp. 329–342.

[33] J. Wang, Z. Hong, Y. Zhang, and Y. Jin, "Enabling security-enhanced attestation with Intel SGX for remote terminal and IoT," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 1, pp. 88–96, Jan. 2018.

[34] H. Tan, G. Tsudik, and S. Jha, "MTRA: Multiple-tier remote attestation in IoT networks," in *Proc. IEEE CNS*, Las Vegas, NV, USA, 2017, pp. 1–9.

[35] G. Dessouky *et al.*, "LO-FAT: Low-overhead control flow attestation in hardware," in *Proc. 54th DAC.*, Austin, TX, USA, 2017, pp. 1–7.

[36] S. Zeitouni *et al.*, "ATRIUM: Runtime attestation resilient under memory attacks," in *Proc. 36th ICCAD*, Irvine, CA, USA, 2017, pp. 384–391.

[37] A. Ibrahim, A. R. Sadeghi, G. Tsudik, and S. Zeitouni, "DARPA: Device attestation resilient to physical attacks," in *Proc. 9th ACM WiSec*, Darmstadt, Germany, 2016, pp. 171–182.

[38] K. Eldefrawy, N. Rattanavipanon, and G. Tsudik, "HYDRA: Hybrid design for remote attestation (using a formally verified microkernel)," in *Proc. 10th ACM WiSec*, Boston, MA, USA, 2017, pp. 99–110.

[39] X. Carpent, N. Rattanavipanon, and G. Tsudik, "Remote attestation of IoT devices via SMARM: Shuffled measurements against roving malware," in *Proc. IEEE Int. Symp. HOST*, Washington, DC, USA, 2018, pp. 9–16.

[40] T. Abera *et al.*, "C-FLAT: Control-flow attestation for embedded systems software," in *Proc. ACM SIGSAC*, Vienna, Austria, 2016, pp. 743–754.

[41] F. Kohnhäuser, N. Büscher, S. Gabmeyer, and S. Katzenbeisser, "SCAPI: A scalable attestation protocol to detect software and physical attacks," in *Proc. 10th WiSec*, Boston, MA, USA, 2017, pp. 75–86.

[42] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, "Energy-aware wireless microsensor networks," *IEEE Signal Process. Mag.*, vol. 19, no. 2, pp. 40–50, Aug. 2002.

[43] P. S. L. M. Barreto and M. Naehrig, *IEEE P1363.3 Submission: Pairing-Friendly Elliptic Curves of Prime Order With Embedding Degree 12*, Piscataway, NJ, USA: IEEE Stand. Assoc., 2006.

[44] S. Ganeriwal, L. K. Balzano, and M. B. Srivastava, "Reputation-based framework for high integrity sensor networks," *ACM Trans. Sensor Netw.*, vol. 4, no. 3, pp. 1–37, May 2008.

[45] G. Hatzivasilis, I. Papaefstathiou, and C. Manifavas, "SCOTRES: Secure routing for IoT and CPS," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 2129–2141, Dec. 2017.

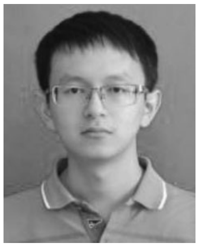
[46] G. Hatzivasilis, I. Papaefstathiou, and C. Manifavas, "ModConTR: A modular and configurable trust and reputation-based system for secure routing in ad-hoc networks," in *Proc. IEEE/ACS AICCSA*, Doha, Qatar, 2014, pp. 56–63.

[47] E. B. Barker, D. Johnson, and M. E. Smid, *Recommendation for PairWise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised)*, Nat. Inst. Stand. Technol., Gaithersburg, MD, USA, 2007. [Online]. Available: https://csrc.nist.gov/publications/detail/sp/800-56a/revised/archive/2007-03-14

[48] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Trans. Wireless Commun.*, vol. 1, no. 4, pp. 660–670, Dec. 2002.

[49] X. Fan and Y. Song, "Improvement on LEACH protocol of wireless sensor network," in *Proc. SENSORCOMM*, Valencia, Spain, 2007, pp. 260–264.

[50] *OMNeT++: Discrete Event Simulator*. Accessed: May 30, 2019. [Online]. Available: https://omnetpp.org/



Boyu Kuang received the bachelor's degree in computer science and technology from the Nanjing University of Science and Technology, Nanjing, China, in 2016, where he is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering.

His current research interests include Internet of Things security and privacy preserving.

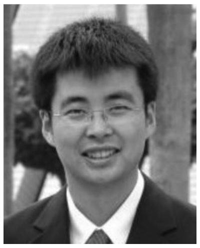
Anmin Fu (M'16) received the Ph.D. degree in information security from Xidian University, Xi'an, China, in 2011.

He is currently an Associate Professor and a Supervisor of Ph.D. students with the Nanjing University of Science and Technology, Nanjing, China. He is also a Visiting Research Fellow with the University of Wollongong, Wollongong, NSW, Australia. He has authored or coauthored over 50 technical papers, including being published in international journals and conferences, such as the IEEE Transactions on Vehicular Technology, the IEEE Transactions on Big Data, IEEE Communications Letters, the *Journal of Network and Computer Applications*, *Computers and Security*, *Cluster Computing*, *Security and Communication Networks*, IEEE ICC, and IEEE GLOBECOM. His current research interests include Internet of Things security, cloud computing security, and privacy preserving.

Shui Yu (SM'12) received the Ph.D. degree in information technology from Deakin University, Geelong, VIC, Australia, in 2004.

He is currently a Full Professor with the School of Software, University of Technology Sydney, Sydney, NSW, Australia. He has had 2 monographs published, he has edited 2 books and has authored or coauthored over 200 technical papers in top journals and conferences, such as the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, the IEEE TRANSACTIONS ON MOBILE COMPUTING, the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING, the IEEE/ACM TRANSACTIONS ON NETWORKING, and INFOCOM. He initiated the research field of networking for big data in 2013. He has an H-index of 32. His current research interests include security and privacy, networking, big data, and mathematical modeling.

Dr. Yu is a member of the AAAS and ACM. He actively serves his research communities in various roles. He served as an Associate Editor for the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS from 2013 to 2015. He is currently serving on the Editorial Boards of IEEE COMMUNICATIONS SURVEYS AND TUTORIALS (Exemplary Editor for 2014), IEEE ACCESS, the IEEE INTERNET OF THING JOURNAL, IEEE COMMUNICATIONS LETTERS (Exemplary Editor for 2016), and a number of other international journals. He has organized several special issues either on big data or cybersecurity. He has served over 70 international conferences as a member of their Organizing Committee, such as the Publication Chair for IEEE Globecom 2015 and IEEE INFOCOM 2016 and 2017, the TPC Co-Chair for IEEE BigDataService 2015 and IEEE ITNAC 2015, and the General Chair for ACSW 2017. He is the Vice Chair of the Technical Committee on Big Data of the IEEE Communication Society and a Distinguished Lecturer of the IEEE Communication Society.



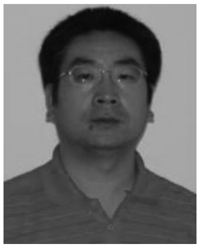
Guomin Yang (SM'17) received the Ph.D. degree from the Computer Science Department, City University of Hong Kong, Hong Kong, in 2009.

He was a Research Scientist with the Temasek Laboratories, National University of Singapore, Singapore. In 2012, he joined the University of Wollongong, Wollongong, NSW, Australia, where he is currently an Associate Professor with the School of Computing and Information Technology. His current research interests include cryptography and network security.

Dr. Yang was a recipient of the Australian Research Council Discovery Early Career Researcher Award in 2015.

Mang Su received the Ph.D. degree in cryptography from Xidian University, Xi'an, China, in 2014.

She is currently a Lecturer with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. Her current research interests include access control and cloud computing.



Yuqing Zhang received the B.S. and M.S. degrees in computer science and the Ph.D. degree in cryptography from Xidian University, Xi'an, China, in 1987, 1990, and 2000, respectively.

He is a Professor and a Supervisor of Ph.D. students with the Graduate University of Chinese Academy of Sciences, Beijing, China. His current research interests include cryptography and network security.