A Fully Authenticated Diffie-Hellman Protocol and Its Application in WSNs

Fajun Sun, Selena He, *Member, IEEE*, Xiaotong Zhang, Jun Zhang, Qingan Li, and Yanxiang He

***Abstract*— The secure authenticated key establishment between nodes in Wireless Sensor Networks (WSNs) has not been fully solved in the existing schemes. It's a good idea to apply the Diffie-Hellman protocol to address it perfectly, but the existing authenticated Diffie-Hellman (ADH) protocols are not perfect because their authentication are partial or delayed. In this paper, we first present a concept of full authentication and propose a new fully authenticated Diffie-Hellman (FADH) prototype with light-certificate-based authentication. And then based on the theory of elliptic curve cryptography, we construct the TinyADH (Tiny Authenticated Diffie-Hellman) protocol with applying the FADH in WSNs. Compared with the existing similar solutions, TinyADH has lower communication overload, is easier to implement into existing standards, and more secure under equivalent computational complexity. The experimental results show that using this scheme for a successful key agreement between two nodes averagely takes about 54 seconds on TelosB. Moreover, the simulation results indicate that repeated key agreement can improve the secure connectivity rate. However, considering the cost performance ratio, it is advisable to take 2 runs of the negotiation.**

***Index Terms*— Authenticated key agreement, key establishment, key management, key confirmation, TinyADH.**

I. Introduction

COMPARED with the traditional network, the sensor network has the characteristics of unified deployment, unattended care, self-organization and limited resources [1], [2]. In sensor networks, whether its topology is a distributed plane or a hierarchical cluster-style level (heterogeneous or homogeneous), as an important part of security is to establish a secure connection between adjacent nodes after the nodes were deployed (bootstrapping security), that is, to establish a pairwise key for any two nodes within their communication range [3]. Therefore, the establishment of pairwise keys has attracted a great deal of attention since the beginning of research on sensor network security [3]–[8]. Early research [3]–[5] on the key establishment had been mainly focused on the establishment with symmetric cryptography. With the feasibility of asymmetric cryptography had been certified [2], [9], since there are some inevitable flaws in the key establishment with symmetric cryptography, the more attentions have been turned to the application of key management and node access control with public keys [6], [7]. Usually key establishment based on public key cryptography has two modes of transport and agreement [10]. Because of the low communication overhead and perfect forward secrecy [11]–[13] of the Diffie-Hellman protocol [14]–[16], in this paper, we mainly focus on how to apply it to securely establish pairwise key for WSNs (Wireless Sensor Networks). Prior literature [2], [17] show that the original Diffie-Hellman model is adopted in the existing protocols based on Diffie-Hellman for WSNs. However, there is a well-known authentication problem [15] in the basic Diffie-Hellman scheme (abbreviated as DH), that is to say, unauthenticated DH is prone to impersonation attacks, man-in-the-middle attacks, forgery attacks and so on. This problem has gotten many discussions in the key agreement research based on traditional networks [10], [15], [18]–[31]. But unfortunately, the authentication in many existing schemes is either partial or delayed (as described in Section III), and even a few schemes have no authentication. Therefore, the authentication problem in the AKA (authenticated key agreement) protocols of WSNs needs to be further solved. Furthermore, these protocols designed for traditional network do not take the characteristics of sensor networks into account, and are not suitable for ad hoc, wireless and open sensor networks. For example, many schemes [24], [27], [28], [31] assume that the public key is obtained through PKI (Public Key Infrastructure) or other trusted public facilities, so they do not exchange public keys to each other in the process of authentication negotiation, even the public key is often not verified, however it is usually necessary to transmit the public keys to each other in sensor networks. In addition, the existing scheme does not consider the problem of overlong agreement package as described in [32]. For more similar requirements of establishing authenticated session keys in sensor networks, one can refer to our another work [33].

In view of the above problems, we have done the work of this paper, and the significant contributions are summarized as follows:

1). It is pointed out that the main problem of the existing authenticated Diffie-Hellman (ADH) protocols is that the two aspects of authentication (including the entity authentication and message authentication) are not completely considered.

Manuscript received September 1, 2021; revised January 31, 2022 and March 19, 2022; accepted April 25, 2022. Date of publication May 9, 2022; date of current version May 26, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61972293 and Grant 61502346; in part by the Research Foundation of Education Bureau of Hunan Province, China, under Grant 19B450; and in part by the Science and Technology Project of the Education Department in Jiangxi Province, China, under Grant GJJ150605. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Nele Mentens. *(Corresponding author: Yanxiang He.)*

Fajun Sun, Xiaotong Zhang, Qingan Li, and Yanxiang He are with the School of Computer Science, Wuhan University, Wuhan 430072, China (e-mail: sunfajun@whu.edu.cn; icyghostzxt@qq.com; qingan@whu.edu.cn; yxhe@whu.edu.cn).

Selena He is with the Department of Computer Science, Kennesaw State University, Marietta, GA 30060 USA (e-mail: she4@kennesaw.edu).

Jun Zhang is with the School of Information Engineering, East China University of Technology, Nanchang 330000, China (e-mail: zhangjun_whu@whu.edu.cn).

Digital Object Identifier 10.1109/TIFS.2022.3173536

Therefore, the fully authenticated Diffie-Hellman (FADH) protocol and related concepts are defined and a FADH prototype is constructed.

2). Based on the definition of full authentication, AKA protocols are considered to fall into three categories: FAKA (fully AKA), PAKA (partially AKA), DAKA (delayed AKA), and the existing ADH protocols are divided into another three categories: partial authentication, delayed authentication, no authentication.

3). Some important issues in the application of ADH protocols to WSNs are discussed. Two FADH protocols (collectively referred to as TinyADH), namely TinyADH-NCR (non Challenge-Response) and TinyADH-CR, are built with the idea of FADH. The proposed protocols have lower communication overload, are easier to implement into existing standards, and more secure under equivalent computational complexity.

4). TinyADH is implemented on TinyOS in nesC. Based on the implementation of TinyADH, the retransmission simulation experiments of TinyADH further verify that two rounds of key agreement is appropriate for improving the secure connectivity rate [33].

The remainder of the paper is organized as follows. Section II discusses the related work. Section III describes preliminaries of this work including notations, definitions, assumptions, security requirements and design principles of AKA. Section IV gives the prototype and the concrete protocol in WSNs—TinyADH and related background information. Section V proves the correctness and security of TinyADH, and the performance is analyzed. Section VI describes the implementation and experiments on TelosB and simulations on MICAz. The evaluation of TinyADH is also presented in this section. Section VII concludes this paper.

II. Related Work

There are many authenticated Diffie-Hellman schemes in the traditional networks, the state-of-the-art research can be traced to the Diffie's work [15], where Diffie proposed many design principles for a good AKA protocol. A definition of secure protocol has been proposed from the view of run process and some desirable characteristics of secure protocols are discussed in his paper. A STS protocol is introduced in the literature and its security flaws are exposed, but there is no secure authenticated key agreement scheme and concrete agreement protocol. Then in 1998, Law *et al.* [10] proposed an efficient two-pass protocol for authenticated key agreement—MQV, which is based on Diffie-Hellman key agreement protocol and can be modified to work in an arbitrary finite group. Blake-Wilson *et al.* [18] reviewed the ADH protocols before 1998, and restated the security goals, desirable security attributes of a key agreement protocol. In this literature, the static and dynamic DH protocols, MTI/C0, KEA, 1-pass MQV, 3-pass MQV, AKC, UC-AKC and so many protocols are analyzed and compared. At last they think that MQV with key confirmation (KC) is a better scheme. Subsequently in 2003, Hugo Krawczyk [19] presented a protocol, namely SIGMA, which provides perfect forward secrecy via a DH protocol and uses digital signatures for authentication. He puts forward some ideas which are used in this article, but the authentication is not used in every message, at least the first message g^x is not authenticated with digital signature. And the entity authentication is not divided into authentication on public key and authentication on private key. In 2005, Maurizio Adriano Strangio [34] proposed a new two-party mutual authenticated key agreement protocol, namely ECKE-1, based on the former researches, such as MTI/A0, UM, MQV, LLK, SK, SSEB. Krawczyk [16] also proposed an improved protocol on MQV in the same year. They first analyzed the MQV protocol in CK formal model of key exchange and found that none of the stated security goals of MQV is actually satisfied. And then they proposed the HMQV, means "hashed variant" MQV which was believed providing with the same superb performance and functionality of the original protocol but for which all the MQV's security goals can be formally proven to hold in the random oracle model under the computational Diffie-Hellman assumption [16]. Then Sarr *et al.* [35] proposed a fully hashed MQV protocol which preserves the remarkable performance of HMQV and gets more secure. However, the protocols mentioned above are all partially authenticated. Similarly the protocols presented in Yao and Zhao [25] and Yasmin *et al.* [26] are also partially authenticated. Recently, certificateless mechanism has been favored by many scholars, but we find that the two-party key agreement protocols based on certificateless mechanism [24], [27], [28], [30], [31] have a similar defect, that is, they are prone to resources exhausting attack [36] due to their delayed authentication.

As for WSNs, early researches based on symmetric cryptography expose many problems, such as the master-key-shared schemes are usually based on the assumption that there is no node capture in a threshold of safe time [37]. And the schemes based on random key pre-distribution have the shortcomings that compromising a small number of nodes captures a large portion of the key pool and the storage size of the key ring increases rapidly with the increasing of connection rate [3]–[5]. After the feasibility of the public key cryptography in WSNs is verified [9], there are some state-of-the-art applications [2], [17] of Diffie-Hellman-based protocols in Sensor Networks, but they are based on the original Diffie-Hellman protocol, which has some security flaws due to the lack of authentication. Afterwards, Mišić [22] investigated network traffic and energy consumption effects of public key cryptography and found that the introduction of public key cryptography needs to consider the limitation of frame length in IEEE802.15.4. She proposed a scaled SSL (Secure Socket Layer) scheme for WSNs. But the Challenge-Response (CR) mechanism doesn't effect, for the challenge value hasn't been added into the ephemeral public key value packets (*i.e.* packets $k_n * Gk_{sn}$ and $k_c * Gk_{sc}$), then these two packets can be replayed. And the representation of negotiated packets is not completely correct. Recently there are some variant Diffie-Hellman schemes, such as identity-based TinyPBC [38], certificateless CL-EKM [7], AKAIoTs [29] and so on. Among them, TinyPBC adopts the identity of nodes as the public key, which enables anyone to impersonate a node with its identity. It can be avoided by a key confirmation, but at

least it leads to a delayed authentication. CL-EKM provide an effective key management scheme whose functions include key establishment, key update, adding and deleting node, etc. Now it seems that CL-EKM is a perfect solution. However the public key of H-sensors in CL-EKM is not authenticated and only after received a key confirmation L-sensor can know whether the H-sensor is valid. So the authentication on H-sensor is a delayed authentication, which may suffer from such DoS attacks [29] as energy exhausting attack [36]. AKAIoTs is a fully authenticated key agreement scheme, but there are some flaws in it, such as ephemeral key leakage attack, unable to broadcast due to binding ID, key escrow, and so on. Although Avoine *et al.* [39] provided a good PFS (Perfect Forward Secrecy) solution based on symmetric key mechanism, their solution is still based on pre-shared key, which is exactly what this paper wants to avoid.

In addition, in the formal proof of security protocols, Ran and Krawczyk [40] presented a formalism for the analysis of key-exchange protocols, this model is called as CK-model by later researchers, which is widely used in the proof of provable security. Bresson *et al.* [41] presented a proof method of Provably-Secure Authenticated Group Diffie-Hellman Key Exchange. Rosario Gennaro *et al.* [23] presented a concept of fully authenticated Diffie-Hellman and proved that the mOT and SSF protocols holds some secure attributes, but we find their authentication is delayed. That is, the entities *B* and *A* cannot directly verify α and β after receiving them respectively. They will not be able to verify the legitimacy of α and β until they exchange data using the session key. As a result, the packets α and β can be counterfeited as any random messages in a short period of time. Furthermore, their scheme is based on RSA, which is considered not suitable for sensor networks. The model eCK [42] is another more widely used model for the security proof of AKA schemes.

The above literature can be classified into Table I.

TABLE I
SOME NON-FADH PROTOCOLS

Type	Schemes
Partial Authentication	[6, 10, 15, 16, 19, 23, 25, 26, 34, 35]
Delayed Authentication	[7, 23, 27-29, 31, 38]
No Authentication	[2, 14, 17]

TABLE II
LIST OF NOTATIONS

Notations	Comments
l	$l \in Z^+$ is a security parameter large enough.
$(\cdot)_k$	Use the key k to encrypt / sign the plain text, the operation represents signing under the condition that the key k is a private key.
$\{\cdot\}_k$	Use the key k to sign the hashed plain text
$\mathcal{O}$	Infinity point on an elliptic curve E
$\mathbb{G}_1, \mathbb{G}_2$	Two generating cyclic groups, used by all relevant nodes or users
g, P	They are the generators of groups $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively.
$\mathbb{Z}_q^*$	A finite field of order q
$\mathscr{P}$	An authenticated key agreement (AKA) protocol
(d_i, Q_i)	A pair of private / public keys, d_i represents for the private key of some i, Q_i represents for the public key of i.
N_i, N_j	Sensor nodes i and j, respectively
ID_i	The identification details, or attributes, of node i
$\mathbb{F}$	A finite field
$Cert_i$	The certificate of node or user i
K_{ij}	The pairwise key established by entity i and j
$Nonce_i$	A temporary value for entity i that is used to keep freshness and resist the replay attack.
r_i	A random number used as the ephemeral private key of node or user i
v_i	A version number for the session sub key of ith phase
$\|\|$	An operator that connects text
$\in_R$	Randomly taken from
h, H	Two cryptographic secure hash functions
t, t_i	Some polynomial times
$\varepsilon, \varepsilon_i$	Some negligible advantages
SM	Scalar Multiplication operation
MAC	Message Authentication Code function
$HMAC$	Hash-based MAC

III. PRELIMINARIES

A. Notations, Intractable Problems and Assumptions

1) Notations: For the convenient explanation of problems, according to [10], we have defined the notations used in this article and list most of them in Table II. Other unlisted notations are explained where they appear.

2) Intractable Problems: Let q be a prime with length of l bits and $\mathbb{G}_1$ be a cyclic group of order q generated by g over some finite field $\mathbb{F}$.

Definition 1 (Discrete Logarithm Problem (DLP)): Given $\beta \in \mathbb{G}_1$, compute integer number $x \in \mathbb{F}$ in equation $\beta = g^x$, ie. compute $x = log_g \beta$.

Definition 2 (Computational Diffie-Hellman Problem (CDHP)): Given $g^x, g^y \in \mathbb{G}_1$ ($x, y \in_R \mathbb{F}$ is unknown), compute g^{xy}.

When we select Elliptic Curve Cryptography operations as the asymmetric cryptography primitives, $\mathbb{G}_2$ should be a cyclic group of order q generated by P over finite field $\mathbb{Z}_q^*$, where P is a point on elliptic curve E.

Definition 3 (Elliptic Curve Discrete Logarithm Problem (ECDLP)): Given $R, Q \in \mathbb{G}_2$, compute integer $a \in \mathbb{Z}_q^*$, so that the equation $Q = aR$ holds.

Definition 4 (Computational Diffie-Hellman Problem Based on Elliptic Curve (EC-CDHP)): Given $Q, aQ, bQ \in \mathbb{G}_2$ ($a, b \in_R \mathbb{Z}_q^*$ is unkown), compute $abQ \in \mathbb{G}_2$.

3) Assumptions: In this paper, we assume that the attacker can eavesdrop on, intercept, inject, store, tamper with communication packets, and have the ability of mathematical reasoning and computing in polynomial time. In addition, we define a group of assumptions as follows.

Assumption 1: DLP, ECDLP, CDHP and EC-CDHP are computationally intractable. At least the probability ε that any of these problems can be solved in polynomial time t is

negligible. Or for short, DLP, ECDLP, CDHP and EC-CDHP are (t, ε)*-intractable problems.*

Assumption 2: The primitive used to encrypt in the protocol is secure. Formally speaking in theory of indistinguishability, encryption algorithm is secure means the algorithm can achieve (t, ε)*-IND-CCA2 security [33].*

Assumption 3: The signature generated in the AKA protocol is unforgeable (in polynomial time). Formally speaking, the signature algorithm is secure means the algorithm can achieve (t, ε)*-EUF-ACMA (Existentially UnForgeable - Adaptive Chosen-Message Attack) security [33].*

Assumption 4: The ephemeral private key is secret, at least (t, ε)*-secret (as defined in Section III-B).*

Assumption 5: The private key of Certificate Authority (CA) is secret, at least (t, ε)*-secret.*

Assumption 6: The private key of entity is secret, at least (t, ε)*-secret.*

Assumption 7: It is assumed that the holder of the private key is the owner of the private key before the key agreement is completed.

For the unreliable communication, we will discuss it as a special case later, here we focus on the reliable communication. Thus the channel can be assumed to be reliable, that is, both sides of the communication entities will receive the other party's package correctly except for the interference of attackers. As for the interference of attackers, due to the Assumption 3, there are only following possibilities in a FAKA protocol: the adversary damaged the negotiation message and was detected by the receiver, or the attacker blocked back the negotiation message. Obviously we can treat the above interference as a channel interference, which is an unreliable problem that will be discussed later.

B. Security Requirements

Many of the existing studies [10], [15], [16], [18] have proposed that security requirements (or targets), such as KKS (Known-key Security) / KSK-R (Resilience to Known Session Keys), PFS, KCI-R (Resistance to Key Compromise Impersonation), NUKS (No Unknown Key Sharing), NKC (Non Key Control) [16], should be considered when we design an AKA protocol. Learning from previous experiences, we think that an AKA protocol should implement the **authentication** and **confidentiality**. In order to formally explain confidentiality and authentication, we do the following two formal definitions:

Definition 5 ((t, ε)*-Authenticated): When we say that a user (entity) or a message (string) is* (t, ε)*-authenticated, it means that the probability that the user / entity can be impersonated in polynomial time* t *is* ε*, or the probability that the message can be forged in polynomial time* t *is* ε*, and* ε *is negligible.*

The corresponding authentication operation is denoted as (t, ε)-authenticate or (t, ε)-authentication.

Definition 6 ((t, ε)*-Secret): When we say that a message or string is* (t, ε)*-secret, it means that the probability that the message or string can be computationally revealed from the public data in polynomial time* t *is* ε*, and* ε *is negligible.*

1) Authentication: Through authentication, each entity can ensure that the entity with which he / she has established secure connection is a valid entity that holds the private key certified by the KGC (or CA). Here we call this valid entity as an **interior entity** and, with Assumption 7, the private key can be the only authentication of the entity's legal identity. In addition, through message authentication, the entity can ensure that the message received is from an interior entity. In a word, authentication can make sure that the entity connected to the network and the exchanged message or data are not a fake. As a result, we think that an authentication for key agreement should include two parts: **entity authentication** and **message authentication**.

- Authentication on Entity, namely Entity Authentication: Entities participating in key negotiation should authenticate their identity to each other. It may be **unilateral (single side) authentication** in some special applications, but we think it should be **bilateral (mutual) authentication** in general, especially in the open wireless environments such as WSNs. It should include **authentication on public key** and **authentication on private key** when we use the public key primitives to implement the authentication. Preferably, in order to reduce the loss of resource exhausting attack, authentication on public and private key should not be integrated.
- Authentication on Message, namely Message Authentication: We can implement this function with a message authentication code (MAC) based on symmetric cryptography or a digital signature based on asymmetric cryptography. But during the process of key exchange, it is unfeasible to employ MAC since the session key is just being negotiated. Certainly we can use the pre-shared main key, but it is believed that only one node / user captured can compromise the whole network. So we'd better use the digital signature based on asymmetric cryptography to implement the message authentication. Generally speaking, the authentication on private key of an entity is implied in the message authentication. Of course, we may not need to authenticate the whole message or all messages, but at least we need to authenticate the public data implied secret data, such as the part implied secret values or ephemeral keys, because we usually use it to generate the session key.

For authentication, we have the following definitions.

Definition 7 (Authenticated Channel): Given a channel of two-party communication model shown in Fig. 1, if the two parties can authenticate to each other and the messages transmitted in it can be authenticated by the receiver. Then, we call it an Authenticated Channel (consisted of A, B and the channel). Formally speaking, in polynomial time t*, the probability* ε *that the entity is impersonated or the message sent by the entity is forged is negligible. We call this channel* (t, ε)*-Authenticated Channel.*

With an Authenticated Channel, there are some good attributes as follows.

- Entities can confirm each other who is the other and whether the other is an interior entity;
- A entity (such as A) can confirm the messages received from the channel is just from the other entity (such as B).

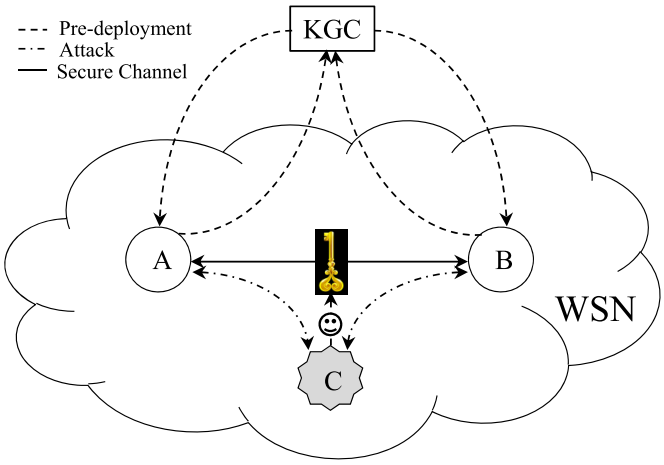


Fig. 1. A two-party communication model.

Definition 8 (AKA Channel): Given a channel of two-party communication model shown in Fig. 1, if the two parties can authenticate to each other and the public data used to generate session key (e.g. ephemeral public keys) can be authenticated by the receiver. Then, we call it an Authenticated Key Agreement Channel (consisted of A, B and the channel). Formally speaking, in polynomial time t, the probability ε that the entity is impersonated or the public data sent by the entity is forged is negligible. We call this channel (t, ε)-AKA Channel.

With an AKA channel, there are some good attributes as follows.

- Entities can confirm each other who is the other and whether the other is an interior entity;
- A entity (such as A) can confirm the public data (implied secret values or ephemeral keys and used to generate session keys, the same below) is just from the other entity (such as B), that is, the public data is not fake.

Obviously, an Authenticated Channel implies an AKA channel.

Definition 9 (Computational Leakage): For a protocol or algorithm, if only through mathematical deduction, the relevant secret data such as private keys, ephemeral keys, session keys, and so on cannot be calculated from the public data of the protocol, we say there is no Computational Leakage, otherwise, there is a Computational Leakage in the protocol or algorithm. Formally speaking, in polynomial time t, the probability ε that the above secret data is calculated from the public data of the protocol is not negligible, then we say there is a (t, ε)-Computational Leakage.

Definition 10 (FA & FAKA Protocol): For an AKA protocol $\mathscr{P}$, if every participant of $\mathscr{P}$ is authenticated (including Entity Authentication and Message Authentication):

***i**. The public data implied secret values or ephemeral keys is authenticated, or,*

***ii**. even every agreement message is authenticated on the base of entity authentication.*

*Then, we call this authentication **full authentication** (FA) and the corresponding protocol $\mathscr{P}$ a **FAKA** protocol. Formally speaking, if the authentication is a (t, ε)-authentication, we call the authentication is a (t, ε)-full authentication and the corresponding protocol $\mathscr{P}$ a (t, ε)-**fully authenticated key agreement** protocol.*

Similarly, in the above definition, since the public data is a part of the message, item **ii** implies item **i**.

*Definition 11 (2-Party FAKA Protocol): For a two-party communication model shown in Fig. 1, we call a FAKA protocol running on it a **2-party FAKA** protocol, whose authentications should be mutual.*

Obviously for a FAKA protocol with public key cryptography (PKC), the public key of the entity should be authenticated at first, and then we can use the public key to authenticate the private key and exchanged messages or data. In this paper we mainly discuss the 2-party FAKA protocol with PKC, so if there is no special explanation, the later FAKA protocol refers to the 2-party FAKA protocol with PKC.

At the same time, a protocol only with partial authentication is called **PAKA** protocol, such as the protocols with unilateral authentication and the protocols with only entity authentication. We call an authentication as a **delayed authentication** when this authentication is not completed in time (before the next message is processed) after the current message is received, the corresponding key agreement protocol is called **DAKA** protocol. Partial authentication is feasible in some scenarios, for example, the authentication of a broadcasting can be unilateral since the entity only needs to authenticate that the message is from a security center. But in sensor networks with an open environment, we think it need a full authentication for an authenticated key agreement protocol.

2) Confidentiality: That is, we should make sure that the secret data (*e.g.* keys) is only known by both parties, in theory regardless of when and where. As for AKA protocols, at least the session key should be (t, ε)-secret. In practical application, we should make it be computationally undiscovered (such as IND-CCA2 [7] security).

There are two levels of confidentiality for AKA: secrecy, perfect forward secrecy.

- Secrecy: The key established in agreement (we call it session key) is secret at least before the private key is revealed. Here the confidentiality of session key only depend on the long-term and ephemeral secrecy data or key.
- Perfect Forward Secrecy (PFS): The session key is secret even if the private key is revealed after the key agreement. In this case, the confidentiality of the session key only depend on the ephemeral secrecy data or key and the ephemeral secrecy data is known only by the entity itself regardless of when and where. Sometimes even a party involved in the negotiation does not know the ephemeral secrecy data of the other entity.

Here, secret should be (t, ε)-secret in practice.

According to the existing research, Diffie-Hellman protocol can make the key-dependent temporary security data only known by the entity that generated it while negotiating the key. So Diffie-Hellman holds the characteristic of perfect forward secrecy. We should use the protocol like Diffie-Hellman to negotiate the session key as we can as possible.

Definition 12: If an AKA protocol $\mathscr{P}$ is a FAKA protocol and the session key is established with Diffie-Hellman

protocol or its variants, we call $\mathscr{P}$ a fully authenticated Diffie-Hellman (FADH) protocol. Formally speaking, here if $\mathscr{P}$ is a (t, ε)-FAKA protocol and the session key is established with Diffie-Hellman protocol or its variants, we call $\mathscr{P}$ a (t, ε)-FADH protocol.

At last, we define the leveled secure AKA protocols as follows.

Definition 13 (Unconditional-AKA-Secure): Given an AKA protocol $\mathscr{P}$, after a run of it under the reliable communication, the two parties (like the entities A and B as shown in Fig. 1) of $\mathscr{P}$ can be sure that: ***i****. They have established a session key with which interior entity;* ***ii****. The session key is a secrecy (only known by A and B);* ***iii****. There is no computational leakage, at least there is no computational leakage of private key. Then $\mathscr{P}$ is an unconditional secure AKA protocol, in other words, $\mathscr{P}$ is Unconditional-AKA-Secure.*

Definition 14 ((t, ε)-AKA-Secure): Given an AKA protocol $\mathscr{P}$, under the Assumption 1-7, after a run of $\mathscr{P}$ under the reliable communication, if the advantage ε that the attacker gets the session key or one of private keys in polynomial time t is negligible. Then $\mathscr{P}$ is a (t, ε)-secure AKA protocol, in other words, $\mathscr{P}$ is (t, ε)-AKA-Secure.

Definition 15 (Weakly-(t, ε)-AKA-Secure): Given an AKA protocol $\mathscr{P}$, under the Assumption 1-7, after a run of $\mathscr{P}$ under the reliable communication, if the advantage ε that the attacker gets the session key in polynomial time t is negligible. Then $\mathscr{P}$ is a weakly (t, ε)-secure AKA protocol, in other words, $\mathscr{P}$ is Weakly-(t, ε)-AKA-Secure.

Definition 16 (PFS-$(t, \varepsilon +)$-AKA-Secure): Given an AKA protocol $\mathscr{P}$, under the Assumption 1-4 and with Assumption 5-7 before the key agreement is finished (PFS), after a run of $\mathscr{P}$ under the reliable communication, if the advantage ε that the attacker gets the session key or one of private keys in polynomial time t is negligible. Then $\mathscr{P}$ is a (t, ε)-secure AKA protocol with PFS, in other words, $\mathscr{P}$ is PFS-(t, ε)-AKA-Secure.

In addition, to avoid replay attack, we need a Challenge-Response [16] or time-stamp mechanism, but the precise time synchronization is expensive for the sensor networks and the loose time synchronization is unsuitable for a time-stamp mechanism to keep higher security. So the Challenge-Response mechanism is the first choice for the sensor networks.

Definition 17 (RRA-(t, ε)-AKA-Secure): Given an AKA protocol $\mathscr{P}$, if it is (t, ε)-AKA-Secure and can resist replay attacks. Then $\mathscr{P}$ is a (t, ε)-secure AKA protocol with resistance of replay attacks (RRA), in other words, $\mathscr{P}$ is RRA-(t, ε)-AKA-Secure.

We can define more levels about secure AKA protocols.

C. Design Principles

To design a perfect AKA protocol, Diffie [15], Menezes [43], Blake-Wilson [18], Law [10], *et al.* have proposed many desirable design principles. Based on the results of predecessors, we put forward the following design principles.

1) Security: Ensure full authentication (including entity authentication and message authentication), secrecy (best to achieve FS, even PFS), asymmetry (adding the identity to the message for preventing the attacks such as reflection attack, etc.), and Challenge-Response mechanism.

2) Implementation:

- Symmetry: mainly role symmetry, the messages transmitted have the same structure, especially for the peer-to-peer networks.
- In line with existing standards: such as the limitation of IEEE802.15.4 on frame length [22] etc.
- Other: reference to [18], such as anonymity, non-interactiveness, non-reliance on encryption, non-reliance on hash functions and time-stamping.

3) Performance: Keep the overhead as low as possible, such as low computation, communication, storage overhead, energy consumption etc. The messages should be short, less, and best to be broadcast.

Certainly, there is no unified principle here, we should adapt the different principles to different application requirements. Obviously the principles suitable to the high security would be different from the principles suitable to high performance.

IV. Proposed Protocols

A. FADH Prototype

At first, we assume the public key of CA, the pair of public / private keys and certificate of entity and other public data have been distributed into every entity, as is feasible in WSNs with unified deployment. As for traditional network, we believe that there must exist a trust channel to distribute these secure data, such as the identity certificate authority of government. So this assumption is reasonable. If there is no explanation, the ADH mentioned later refers to FADH.

According to the users' security requirements, our schemes can be divided into two kinds: ADH with CR (abbreviated as ADH-CR) and ADH without CR (abbreviated as ADH-NCR). The first scheme provides an ADH protocol with Challenge-Response mechanism, which is suitable for the scenario with high security, while the second scheme provides an ADH protocol without Challenge-Response mechanism, which is suitable for the scenario with low security. In the scheme without CR mechanism, an attacker can use the replay attack to create a session key between legitimate nodes that are not within the scope of communication. With the transmission of the data subsequently, this situation will be gradually detected. So it may be a feasible scheme in the traditional network with enough energy and storage. In ADH-CR mode, we can only unicast agreement packets, while in ADH-NCR mode, we can broadcast agreement packets. Usually, with key confirmation, we can compose them in the following two ways:

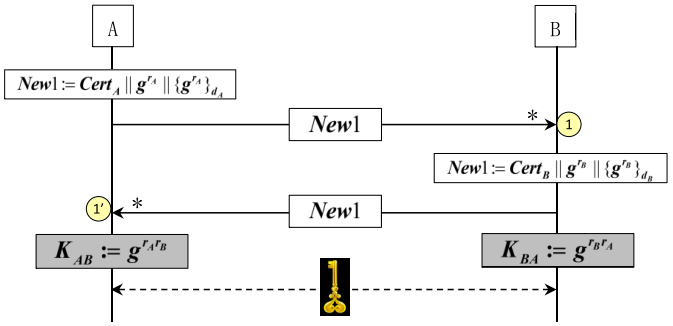
i. CR + unicast + KC-Piggyback;

ii. NCR + broadcast + KC.

Here, KC-Piggyback means the key confirmation will be piggybacked in the next data transmission.

The two schemes we have designed are shown in Fig. 2 and Fig. 3.

1) ADH-NCR Scheme: The ADH-NCR scheme consists of two phases: the agreement phase and the key generation phase. At first, entities A and B select separately a random



Step 1 or 1': Upon receipt of the negotiation packet New1, it is necessary to verify the certificate and signature, the pairwise key will be calculated and stored when the verification is passed.

Fig. 2. ADH-NCR scheme.

number $r_A, r_B \in_R \mathbb{Z}_q^*$. In the agreement phase there is only one exchanged agreement packet (two-pass). Because of the removal of the CR mechanism, this agreement packet can be transmitted by energy-saving broadcast. Here we define the exchanged message as follows:

$$New1 := Cert_A || g^{r_A} || \{g^{r_A}\}_{d_A}.$$

After the entity, such as A, received the exchanged message New1 from B, he / she first does the following two verification sequentially:

- Verify the certificate and public key of B: Verify with the public key of CA whether the certificate and public key of B are valid.
- Verify the exchanged message and private key: When the previous verification is passed, verify with the public key of B whether the exchanged message g^{r_B} and private key of B is valid.

If all of above verifications are passed, entity A can compute the session key as follows:

$$K_{AB} := g^{r_A r_B}$$

In the same time, the entity B can finish the above verification and computation process and gets the same session key $g^{r_A r_B}$.

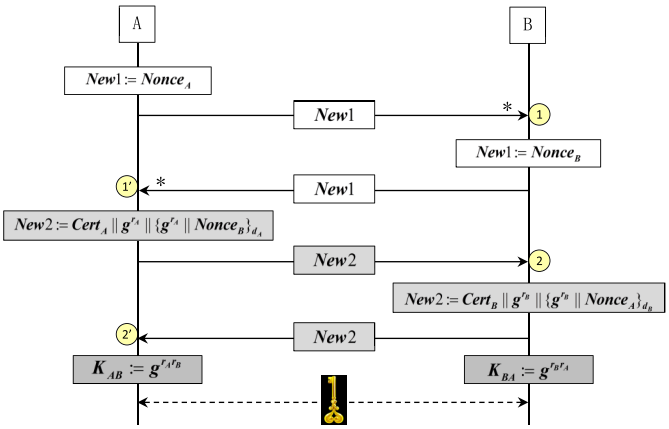
2) ADH-CR Scheme: The ADH-CR scheme consists of about three phases: broadcasting nonce, sending certificate and key agreement packet and the key generation phase. But each phase is not isolated, and they can be crossed properly. At the first phase, each entity broadcasts a New1 packet as follows:

$$New1 := Nonce_A.$$

In the second phase, once the entity (such as B) receives a New1 packet from the other, he / she stores the Nonce information into temporary space, then selects a random number $r_B \in_R \mathbb{Z}_q^*$ and encapsulates the New2 packet as follows:

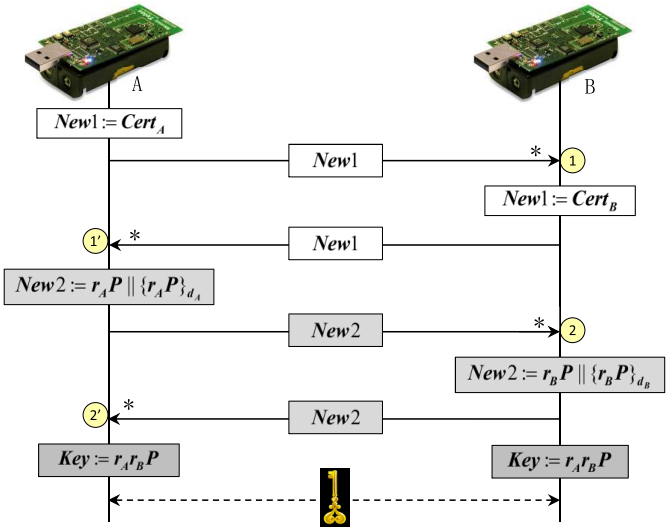
$$New2 := Cert_B || g^{r_B} || \{g^{r_B} || Nonce_A\}_{d_B}.$$

And then the entity B sends the New2 packet to the other entity (such as A). At last, in the third phase, once the entity (such as A) received the New2 packet, he / she verifies the certificate of the other party (such as B) at first. When the certificate is valid and belong to B indeed, the entity will continue to verify whether the signature of nonce and agreement message is



Step 1 or 1': Upon receipt of the negotiation packet New1, it is necessary to register the Nonce information; Step 2 or 2': Upon receipt of the negotiation packet New2, the certificate and signature will be verified, the Nonce values are compared, and then the pairwise key will be calculated and stored with the security information such as certificate when the verification is passed.

Fig. 3. ADH-CR scheme.



Step 1 or 1': Upon receipt of the negotiation packet New1, it is necessary to verify the certificate and register the security information when the verification is passed; Step 2 or 2': Upon receipt of the negotiation packet New2, the signature will be verified, and the pairwise key will be calculated and stored when passed.

Fig. 4. TinyADH without challenge-response.

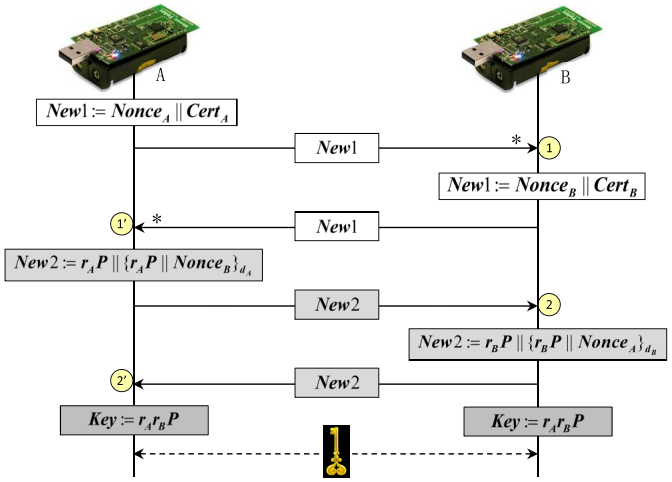
valid. When the verification is passed and the Nonce values are matched, the entity will compute the session key as follows:

$$K_{AB} := g^{r_A r_B}.$$

The complete process is shown as Fig. 3.

B. TinyADH

Based on the above FADH prototype, two ADH protocols suitable for sensor networks, namely TinyADH-NCR and TinyADH-CR (collectively referred to as TinyADH), are designed on the characteristics of sensor networks. Because of the limitations of computation, communication, storage and energy, we cut the ADH protocol slightly for sensor networks. Here we assume the AKA protocol is based on PKC and selects the ECC as the primitives. At first, according to the



Step 1 or 1': Upon receipt of the negotiation packet New1, it is necessary to verify the certificate and register the security information when the verification is passed; Step 2 or 2': Upon receipt of the negotiation packet New2, the signature will be verified, and the pairwise key will be calculated and stored when passed.

Fig. 5. TinyADH with challenge-response.

standard IEEE802.15.4 and our test on practical sensor node TelosB in TinyOS 2.1.1, the length of the packet should be limited to 114B or less. However, if the compression mechanism is not used, the length of the New1 package will reach $82 + 40 + 40 = 162$B. Considering a compression operation requires a high amount of computation and decentralized verification is more conducive to the prevention of DoS attacks, instead of using only one transmission packet like ADH-NCR, we divide it into two packets in TinyADH without CR (TinyADH-NCR). Finally, we designed the two ADH protocols suitable for sensor networks — TinyADH-NCR and TinyADH-CR as shown in Fig. 4 and Fig. 5. TinyADH-NCR is a FADH protocol without Challenge-Response and TinyADH-CR is a FADH protocol with Challenge-Response. Since there is no CR mechanism in TinyADH-NCR, it is not necessary to target a node's messages at a specific node. So we can broadcast the New1 and New2 packets, which can not only save energy but also improve the secure connection rate. In the meanwhile, without the CR mechanism, the agreement package can be set shorter and the protocol can be more concise. In TinyADH-NCR, we define the New1 and New2 packets (taking node A as an example) as follows:

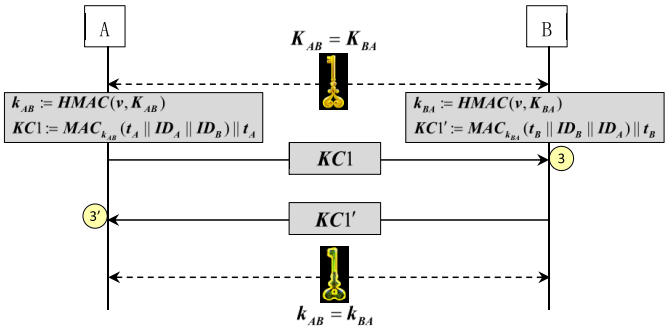
$$New1 := Cert_A,$$
$$New2 := r_A P || \{r_A P\}_{d_A}.$$

And we define the New1 and New2 packets for TinyADH-CR (also taking node A as an example) as follows:

$$New1 := Nonce_A || Cert_A,$$
$$New2 := r_A P || \{r_A P || Nonce_B\}_{d_A}.$$

Finally, when the New1 and New2 packets are correctly received and verified successfully, the session key of the two nodes can be computed as follows:

$$Key := r_A r_B P.$$



Step 3 or 3': Upon receipt of the confirmation packet KC1 or KC1', t_A or t_B will calculate the MAC value by MAC function and shared subkey. And then the calculated MAC value is compared with the received MAC value. If the MAC values are matched, the entity can confirm that the other party and himself have successfully constructed the session key.

Fig. 6. Key confirmation.

C. Key Confirmation

Key confirmation means that the participants of AKA protocol confirm whether the other party has established the same session key as themselves at the end of key agreement period. Combined with the existing experiences [7], [39], [43], [44], we adopt the key confirmation method as shown in Fig. 6, where k_{AB} or k_{BA} is a **phased session key** which can be used to encrypt or sign data with symmetric cryptography in a session phase. Here, to save code space, MAC can also be replaced by HMAC.

In the figure, υ is the version of a sub key, K_{AB} and K_{BA} are the master session keys generated from the key agreement. The value of υ is taken with a hash chain as follows:

$$\begin{cases} \upsilon_0 = h(K_{AB}), \\ \upsilon_i = h(\upsilon_{i-1}). \end{cases}$$

The subscript number of υ is initialized into 0, and can be synchronized with plain text or cipher text (encrypted with the current phased session key) at the start of every data transporting period. Usually υ_0 is used for key confirmation. When the protocol runs in the non-CR mode, it is necessary to explicitly adopt the key confirmation for resisting replay attacks. While the protocol runs in the CR mode, we can use the implicit key confirmation mechanism of data piggyback in the protocol. Certainly we can also extract the technique of generating phased session keys to use in the CR mode.

V. Proofs and Analyses

A. Proofs

Theorem 1: If an AKA protocol $\mathscr{P}$ is a FAKA protocol based on two-party communication model (as shown in Fig. 1), there exists an AKA channel in it. Formally speaking, if the protocol is a (t, ε)-FAKA protocol, the channel is a (t, ε)-AKA Channel.

Proof: Because $\mathscr{P}$ is a FAKA protocol, by Definition 10, every participant of $\mathscr{P}$ is authenticated, that is, the two parties can authenticate to each other. And item **ii** in the Definition 10 implies item **i**, so the public data is authenticated on the base of entity authentication. Therefore, there exists an AKA channel in it.

Formally speaking, if the protocol is a (t, ε)-FAKA protocol, the protocol $\mathscr{P}$ is (t, ε)-authenticated. By Definition 8, the channel is a (t, ε)-AKA Channel. ■

Lemma 1: Given entities of a FAKA protocol $\mathscr{P}$, they can confirm that they have established a session key with which interior entity.

Proof: Because $\mathscr{P}$ is a FAKA protocol, by Theorem 1 and Definition 8, we can get, the entities of $\mathscr{P}$ can confirm that they have established a session key with which interior entity. ■

Theorem 2: A (t, ε)-FADH protocol is at least Weakly-(t, ε)-AKA-Secure.

Proof: Assume $\mathscr{P}$ is such a FADH protocol and the advantage that the attacker gets the session key in polynomial time t is ε. The attacker can only get the advantage ε in polynomial time t from the following three cases: ①, the attacker guesses and gets the session key. Assume the advantage that the attacker gets from this case is ε_1. ②, the attacker impersonates as an interior entity to negotiate the session key with another interior entity. Assume the advantage that the attacker gets from this case is ε_2. ③, the attacker calculates out the session key from the protocol $\mathscr{P}$. Assume the advantage that the attacker gets from this case is ε_3.

The above three cases are discussed below.

Case ①, $\varepsilon_1 = \frac{1}{2^l}$, where l is a security parameter large enough. Obviously it is negligible.

Case ②, by the definition of (t, ε)-FADH (Definition 12) and Theory 1, there exists an AKA channel in $\mathscr{P}$ based on two-party communication model. Then the probability ε_2 that the interior entity is impersonated is negligible.

Case ③, by the definition of (t, ε)-FADH (Definition 12), the session key is established with Diffie-Hellman protocol or its variants. Moreover, the parties establishing the session key are interior entities. By the PFS of Diffie-Hellman protocol, the advantage ε_3 that the attacker gets the session key from the Diffie-Hellman protocol is negligible unless the attacker can sovle CDHP or DLP. But by Assumption 1, this probability ε_3 is negligible.

In summary, by the analysis of cases ①, ②, and ③, $\varepsilon = \varepsilon_1 + \varepsilon_2 + \varepsilon_3$ is negligible. Therefore, by our definition of Weakly-(t, ε)-AKA-Secure (Definition 15), a FADH protocol is at least Weakly-(t, ε)-AKA-Secure. ■

Theorem 3: ADH-NCR is (t, ε)-AKA-Secure.

Proof: Assume the advantage that the attacker gets the session key or one of private keys in polynomial time t is ε. The attacker can only get the advantage ε in polynomial time t from the following four cases: ①, the attacker guesses and gets one of these keys. Assume the advantage that the attacker gets from this case is ε_1. ②, the attacker impersonates as an interior entity to negotiate the session key with another interior entity. Assume the advantage that the attacker gets from this case is ε_2. ③, the attacker calculates out the session key from the protocol ADH-NCR. Assume the advantage that the attacker gets from this case is ε_3. ④, the attacker calculates out one of private keys from the protocol ADH-NCR. Assume the advantage that the attacker gets from this case is ε_4.

The above four cases are discussed below.

Case ①, $\varepsilon_1 = \frac{1}{2^l}$, where l is a security parameter large enough. Obviously it is negligible.

Case ②, in ADH-NCR, the authentication on the public key is implemented with $Cert_A$ and $Cert_B$ in $New1$, and the authentication on the private key and public data is implemented with $\{g^{r_A}\}_{d_A}$ and $\{g^{r_B}\}_{d_B}$, so we have: $\varepsilon_2 \leq \varepsilon_{21} + \varepsilon_{22} + \varepsilon_{23} + \varepsilon_{24}$, where ε_{21}, ε_{22}, ε_{23}, ε_{24} are separately the advantages that the attacker forges $Cert_A$, $Cert_B$, $\{g^{r_A}\}_{d_A}$, and $\{g^{r_B}\}_{d_B}$ successfully. By Assumption 1, 3 and 7, ε_{21}, ε_{22}, ε_{23}, ε_{24} are negligible. So ε_2 is negligible.

Case ③, g^{r_A} and g^{r_B} is public. By Assumption 4, we have r_A and r_B are at least (t, ε_{31})-secret and (t, ε_{32})-secret. Furthermore, according to Definition 2 and Assumption 1, we have: $\varepsilon_3 \leq \varepsilon_{31} + \varepsilon_{32} + \varepsilon_{33} + \varepsilon_{34} + \varepsilon_{35}$, where ε_{33}, ε_{34}, ε_{35} are separately the advantages that the attacker solve DLP to get the keys r_A and r_B, solve CDH to get $g^{r_A r_B}$. And ε_{31}, ε_{32}, ε_{33}, ε_{34}, ε_{35} are negligible. So ε_3 is negligible.

Case ④, in the process of agreement, only $Cert_A$, g^{r_A}, $\{g^{r_A}\}_{d_A}$, $Cert_B$, g^{r_B}, $\{g^{r_B}\}_{d_B}$, $g^{r_{KDC}}$ are sent in the public channel. But according to Assumption 1 and 2, we have $\varepsilon_4 \leq \varepsilon_{41} + \varepsilon_{42} + \varepsilon_{43} + \varepsilon_{44} + \varepsilon_{45} + \varepsilon_{46} + \varepsilon_{47}$, where $\varepsilon_{41} - \varepsilon_{47}$ are separately the advantages that the attacker solve above CDHP or DLP to get the private keys.

In summary, by the analysis of cases ①, ②, ③, and ④, $\varepsilon = \varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \varepsilon_4$ is negligible. Therefore, by Definition 14, we have that ADH-NCR is (t, ε)-AKA-Secure. ■

Theorem 4: TinyADH-CR is not only RRA-(t, ε)-AKA-Secure but also PFS-(t, ε)-AKA-Secure.

Proof: Here, we need to prove the following three statements:

i. TinyADH-CR is (t, ε)-AKA-Secure.

ii. TinyADH-CR is RRA.

iii. TinyADH-CR holds PFS.

For the statement **i**, assume the advantage that the attacker gets the session key or one of private keys in polynomial time t is ε. The attacker can only get the advantage ε in polynomial time t from the following four cases: ①, the attacker guesses and gets one of these keys. Assume the advantage that the attacker gets from this case is ε_1. ②, the attacker impersonates as an interior entity to negotiate the session key with another interior entity. Assume the advantage that the attacker gets from this case is ε_2. ③, the attacker calculates out the session key from the protocol TinyADH-CR. Assume the advantage that the attacker gets from this case is ε_3. ④, the attacker calculates out one of private keys from the protocol TinyADH-CR. Assume the advantage that the attacker gets from this case is ε_4.

The above four cases are discussed below.

Case ①, $\varepsilon_1 = \frac{1}{2^l}$, where l is a security parameter large enough. Obviously it is negligible.

Case ②, in TinyADH-CR, the authentication on the public key is implemented with $Cert_A$ and $Cert_B$ in New1, and the authentication on the private key and public data is implemented with $\{r_A P || Nonce_B\}_{d_A}$ and $\{r_B P || Nonce_A\}_{d_B}$, so we have: $\varepsilon_2 \leq \varepsilon_{21} + \varepsilon_{22} + \varepsilon_{23} + \varepsilon_{24}$, where ε_{21}, ε_{22}, ε_{23}, ε_{24} are separately the advantages that the attacker forges $Cert_A$, $Cert_B$, $\{r_A P || Nonce_B\}_{d_A}$ and $\{r_B P || Nonce_A\}_{d_B}$ successfully.

By Assumption 1, 3 and 7, ε_{21}, ε_{22}, ε_{23}, ε_{24} are negligible. So ε_2 is negligible.

Case ③, $r_A P$ and $r_B P$ is public. By Assumption 4, we have r_A and r_B are at least (t, ε_{31})-secret and (t, ε_{32})-secret. Furthermore, according to Definition 4 and Assumption 1, we have: $\varepsilon_3 \leq \varepsilon_{31} + \varepsilon_{32} + \varepsilon_{33} + \varepsilon_{34} + \varepsilon_{35}$, where ε_{33}, ε_{34}, ε_{35} are separately the advantages that the attacker solve ECDLP to get key r_A and r_B, solve EC-CDHP to get $r_A r_B P$. And $\varepsilon_{31}, \varepsilon_{32}$, ε_{33}, ε_{34}, ε_{35} are negligible. So ε_3 is negligible.

Case ④, in the process of agreement, only $Cert_A$, $r_A P$, $\{r_A P || Nonce_B\}_{d_A}$, $Cert_B$, $r_B P$, $\{r_B P || Nonce_A\}_{d_B}$, $r_{KDC} P$ are sent in the public channel. But according to Assumption 1 and 2, we have $\varepsilon_4 \leq \varepsilon_{41} + \varepsilon_{42} + \varepsilon_{43} + \varepsilon_{44} + \varepsilon_{45} + \varepsilon_{46} + \varepsilon_{47}$, where $\varepsilon_{41} - \varepsilon_{47}$ are separately the advantages that the attacker solve above EC-CDHP or ECDLP problems to get the private keys.

In summary, by the analysis of cases ①, ②, ③, and ④, $\varepsilon = \varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \varepsilon_4$ is negligible. Therefore, by Definition 14, we have TinyADH-CR is (t, ε)-AKA-Secure.

For the statement **ii**, in TinyADH-CR, the Challenge-Response mechanism is implemented in the packets New1 and New2. The challenge is sent through New1 and authenticated through digital signature in New2 to realize the response. The initiator of the challenge can determine whether it is a replay attack by round trip time and Nonce value. If the Nonce value received in New2 is not the same as the sent New1 or the round trip time (RTT) is bigger than a maximal RTT (including the time for one signature verification and one scalar point multiplication), the packet can be judged as a replay. In this way, a replay attack can be avoided. So TinyADH-CR is RRA.

For the statement **iii**, because the Diffie-Hellman scheme is encapsulated in TinyADH-CR, and the Diffie-Hellman scheme has been believed to hold perfect forward secrecy under the condition that the entities are not impersonated [11]. By the statement **i**, the entities are not impersonated. So TinyADH-CR holds PFS.

Combined with the above three statements, we have, TinyADH-CH is not only RRA-(t, ε)-AKA-Secure but also PFS-(t, ε)-AKA-Secure, denoted as RRA & PFS-(t, ε)-AKA-Secure. ■

With the similarly evidence, we can also prove:

1). ADH-CR is a RRA & PFS-(t, ε)-AKA-Secure protocol;

2). TinyADH-NCR is PFS-(t, ε)-AKA-Secure;

3). TinyADH-NCR with KC is a RRA & PFS-(t, ε)-AKA-Secure protocol;

4). ADH-NCR, ADH-CR, TinyADH-NCR, and TinyADH-CR are all FADH protocols.

These proofs will not be detailed here to avoid wordiness. In summary, ADH-NCR, ADH-CR, TinyADH-NCR, and TinyADH-CR are all the secure AKA protocols with different levels.

B. Analyses

1) Other Security Analyses:

- **Meeting the Traditional Security Goals**
  PFS: Obviously, All the FADH protocols proposed in this paper are based on the Diffie-Hellman protocol, which has been believed to hold perfect forward secrecy under the condition that the entities are not impersonated [11].
  KSK-R: Because the ephemeral security data used to generate the session key in our four protocols are the random nonce values and independent of each other, the adversary known some session keys can not derive out other session keys under Assumption 1. In probability, revealed the polynomial f_p session keys do not affect the security of generating new session keys from the key space with exponential size f_e. As long as the hash function is secure, that is, the function value is evenly distributed in the key space, the advantage $\frac{f_p}{f_e}$ that the attacker achieve successful attack is negligible. Thus, the four protocols are resilient to KSK (Known Session Keys) attack. As for the "parallel known session key attack (PKSK)" mentioned in [45], for ADH-CR, because its challenge response mechanism can resist replay attacks, PKSK attacks generated by replay are avoided. For ADH-NCR, we can use the following asymmetric way to resist it. That is, both parties calculate the session key as follows: $Key := H(ID_A, ID_B, r_A r_B P)$, where $H : \{0,1\}^* \times \{0,1\}^* \times G \longrightarrow Z_q^*$ is a cryptographic hash function and A is the initiator of the session. In this way, when the attacker attacks, he / she can only query and obtain the key $H(ID_A, ID_B, r_A r_B P)$, but he / she can't deduce $H(ID_B, ID_A, r_A r_B P)$ from $H(ID_A, ID_B, r_A r_B P)$ or vice versa, which prevents the attack. This method can also be used in ADH-CR to enhance security.
  KCI-R: Because the public / private key pairs of entities or nodes are different from each other, and they are bound one by one with their respective identities through certificates, the private key of an entity or node is disclosed that does not enable the adversary to impersonate other entities to the compromised entity or node.
  NUKS: According to Theorem 1, because an AKA channel has been built in the run of a FAKA protocol, any third party can be blocked out of the channel and the two ends of the channel, the unknown key shared would never happen in the FAKA protocols.
  NKC: Because the session key is generated from the ephemeral security data, which is randomly generated and independent of each other. If the random number generator is a true random number generator, the key would not be controlled.
  Other: In the same time, with full authentication, our protocol can resist MITM (Man-in-the-Middle), impersonation and packet forgery attacks. With

TABLE III
COMPUTATIONAL COMPLEXITY OF ECDH, ECDSA

Items	ECDH	ECDSA
A (sign)	2SM	1SM+1Hash+1Rev+2Mod+2M
B (verify)	2SM	2SM+1PA+1Rev+4Mod+2M+1Hash
Total	4SM	3SM+1PA+2Hash+2Rev+6Mod+4M

The abbreviations in the table have the same meaning as [33].

TABLE IV
COMPARING THE PERFORMANCE OF SEVERAL SCHEMES*

Scheme	Computation (SMs)	Communication (Bytes)		Authentication	AKA-Secure	KC	OL
		Sent	Received				
CL-EKM [7]	6d (4d)	53+213d	266d	DAKA	RRA & PFS	Y	Y
AKAIoTs [29]	6d	152d	152d	DAKA	PFS	N	Y
Scaled SSL [22]	7d	282d	282d	PAKA	PFS	Y	N
TinyAKE [33]#	8d	103+113d	226d	FAKA	RRA	N	N
TinyADH-CR	6d+1	103+93d	196d	FAKA	RRA & PFS	Y	N
TinyADH-NCR	5d+1	188	188d	FAKA	PFS	Y	N

*Parameters: ID: 2B, Point: 40B, Cert: 82B, Hash value: 20B, Nonce: 8B, Header+Tailer: 13B, d: average number of neighbors; the last column means whether the agreement packet is overlong; #: Our another key establishment protocol.

Challenge-Response and timing cleanup mechanisms, TinyADH can resist replay attacks.

- **Resilience to Node Capture**
  Because only the private key, session key and other secrets related to the node are loaded in the node, therefore, node capture will not expose any secret information of other nodes and KGC (or CA) in our protocols. With node capture and replica, attackers can only connect many nodes with the same identity into the network, but there are much literature about the detecting of the node capture attack to mitigate it [46].

2) Performance Analysis: Since this article focuses on the problem of key agreement for sensor networks, we mainly analyze the performance of TinyADH-NCR and TinyADH-CR. The analysis method of ADH and ADH-CR is similar, and it is no longer described here.

- **Analysis of Computational Complexity**
  In TinyADH-NCR, it is necessary for two parties to do the verification of certificate and public data, and a scalar multiplication of the packet New2, so according to Table III, the computational complexity per node for a two-party model is:

$$\begin{aligned}&(2\text{ECDSA.verify}+4\text{SM}+2\text{ECDSA})/2\\&=(2\text{SM}+1\text{PA}+1\text{Rev}+4\text{Mod}+2\text{M}+1\text{Hash})\\&\quad+2\text{SM}+(3\text{SM}+1\text{PA}+2\text{Hash}+2\text{Rev}+6\text{Mod}+4\text{M})\\&=7\text{SM}+2\text{PA}+3\text{Rev}+10\text{Mod}+6\text{M}+3\text{Hash}\\&\approx 7.2\ \text{SM}.\end{aligned}$$

  For the network with n nodes and e edges, the computational complexity per node with average d neighbors of TinyADH-NCR is:

$$\begin{aligned}&\text{SM}+\text{ECDSA.sign}+(n\cdot d\cdot\text{ECDSA.verify}\\&\quad+2\cdot e\cdot(\text{SM}+\text{ECDSA.verify}))/n\\&=\text{SM}+\text{ECDSA.sign}+d\cdot\text{ECDSA.verify}+d\cdot(\text{SM}\\&\quad+\text{ECDSA.verify})\\&=\text{SM}+\text{ECDSA.sign}+d\cdot(\text{ECDSA.verify}+\text{SM}\\&\quad+\text{ECDSA.verify})\\&\approx(2+5.2\text{d})\ \text{SM}.\end{aligned}$$

  The computational complexity per node of TinyADH-CR is:

$$\begin{aligned}&\text{SM}+(n\cdot d\cdot(\text{ECDSA.verify}+\text{ECDSA.sign})\\&\quad+2\cdot e\cdot(\text{SM}+\text{ECDSA.verify}))/n\\&=\text{SM}+d\cdot(\text{ECDSA.verify}+\text{ECDSA.sign})+d\cdot(\text{SM}\\&\quad+\text{ECDSA.verify})\\&=\text{SM}+d\cdot(\text{ECDSA.verify}+\text{ECDSA.sign}+\text{SM}\\&\quad+\text{ECDSA.verify})\\&\approx(1+6.2\text{d})\ \text{SM}.\end{aligned}$$

- **Communication Analysis**
  Assume the protocols work for ECC-160 and the length of header + tailer, ID, signature, Nonce are 13B, 2B, 40B, 8B respectively. Then the length of certificate, DH agreement data, New1, and New2 are 82B, 40B, 8 + 82B, and 40 + 40B respectively.
  For the two-party model, the total communication quantity of sending and receiving in TinyADH-NCR is:

$$2*(82+13)+2*(40+40+13)=376\ \text{(Bytes)}.$$

  For the two-party model, the total communication quantity of sending and receiving in TinyADH-CR is:

$$2*(8+82+13)+2*(40+40+13)=392\ \text{(Bytes)}.$$

  For the network with n nodes and e edges, the sending quantity per node with d neighbors of TinyADH-NCR is:

$$(n\cdot(82+13)+n\cdot(40+40+13))/n=188\ \text{(Bytes)}.$$

  The sending quantity per node of TinyADH-CR with d neighbors is:

$$(n\cdot(8+82+13)+2\cdot e\cdot(40+40+13))/n=103+93d\ \text{(Bytes)}.$$

  For the network with n nodes and e edges, the receiving quantity of TinyADH-NCR per node with d neighbors is:

$$(n\cdot d\cdot(82+13)+n\cdot d\cdot(40+40+13))/n=188d\ \text{(Bytes)}.$$

TABLE V
EXPERIMENTAL ENVIRONMENT

Items	Configurations
Node OS	TinyOS2.1.1
Primitives library	TinyECC2.0
KGC / CA	Intel i7-4790 3.6GHz 16G RAM
Hardware	
KGC / CA OS	Ubuntu 12.04
Sensor Node	TelosB
Simulation Node	MICAz

TABLE VI
COMPARING THE STORAGE SIZE OF SEVERAL PROTOCOLS

Project / Protocol	RAM (Bytes)	ROM (Bytes)
TinyADH-NCR*#	3032	33352
TinyADH-CR*	3074	33532
TinyAKE*$	4074	38136
EPKI [2]	1140	34342

$: Our another key establishment protocol;
*: Queue length=1, Neighbor number=1;
#: Store the public key rather than certificate.

The receiving quantity of TinyADH-CR per node with d neighbors is:

$$(n \cdot d \cdot (8+82+13)+n \cdot d \cdot (40+40+13))/n = 196d \text{ (Bytes)}.$$

- **Energy Consumption Analysis**
  The energy consumption includes the consumption of computation and communication. With the above quantity of computation and communication and the energy consumption of every unit in the literature [17], we can get the consumption very easy.

In order to investigate the performance such as feasibility and security of our scheme, we compare TinyADH with several existing related protocols, as shown in Table IV. Among them, CL-EKM is designed for heterogeneous dynamic wireless sensor networks, where node with decapsulation consumes 6 ECC point multiplications while node with encapsulation consumes 4 ECC point multiplications. As can be see from Table IV, TinyADH-CR has strong security and low communication capacity, and the negotiation packages do not exceed the maximum load of frame, while the computation amount is equivalent to the existing similar better scheme. We can also find that, TinyADH-NCR has lower computation and communication than TinyADH-CR, and its negotiation packages do not exceed the maximum load of frame too. TinyADH-NCR makes an appropriate trade-off between security and cost, which is more suitable for WSNs.

VI. IMPLEMENTATIONS AND EXPERIMENTS

A. Implementations

When a protocol is put forward, there are many problems in its concrete realization. Limited to space, here we only discuss several problems among them.

1) On the Storage of Neighbor Information: The public key or certificate that stores a neighbor will obviously affect the occupancy of the storage space, especially for the sensor network application with static allocation. To reduce storage, we should only store the public key which is passed in the verification.

2) About the Point Compress: As for the Elliptic Curve Cryptography, the point compress is a good operation for the reduce of storage and communication, but it is at the expense of expensive modular multiplication and modular square root. Maybe it is not good choice to use point compress in WSNs.

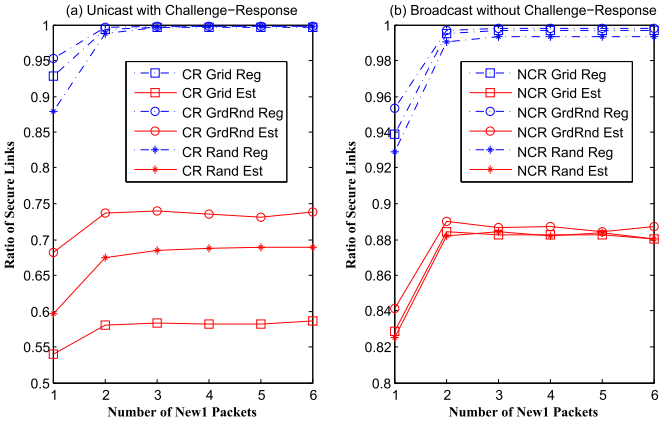


Fig. 7. The influence of re-agreement on secure link ratio.

3) Impact of Implementation on Security: The different implementation of a protocol would have very obvious impact on the security. Take the forward secrecy as an example, in fact it depends on whether the ephemeral secure data is deleted or overwritten thoroughly and the session key is cleared in time when the session is over. About other problems such as queue buffering mechanism, invalid packet cleanup, and so on, we have discussed them in our another work—TinyAKE [33], which would not be expanded here.

B. Experiments

1) Experimental Environment: The experimental environment parameters of hardware and software are listed in Table V.

2) Experiments for TinyADH: We implement TinyADH in an embedded programming language nesC [47] based on TinyOS. The total protocol is based on the primitives library TinyECC. The storage of some protocols (ECC-160) after compiled is shown in Table VI.

The key agreement experiments are finished on two TelosB nodes, the average establishing time of TinyADH-CR is about 52.7s and TinyADH-NCR about 56.5s for ECC-160.

3) Simulation on Retransmission: According to the existing work [33], [44], repeated negotiation is helpful to improve the

security connection rate. In order to study the influence of repeatedly broadcasting New1 packets on the establishment of secure links, we have carried out the simulation study of 6, 49 and 361 nodes in TOSSIM. The topology of 361 nodes is arranged into a 19 x19 mesh grid on a field of 450m x 450m. The distance between nodes is 25m and the noise model file is casino-lab.txt in TinyOS.

To find the optimal value of retransmissions, we designed 6 groups of experiments according to the number of repetitions. 20 experiments were repeated in each group, and the average was obtained after recording the results. The final result is listed in Fig. 7. As can be seen from Fig. 7, the second retransmission of the packet New1 lead to a 7.76% (6.66% for non-CR) increase in secure connection rates. But the effect of the third retransmission is not obvious, just 0.32% (0.16% for non-CR), relative to its cost can be ignored. Therefore, we think the 2 transmissions of New1 is suitable for the public-key-based key establishment. As can also be seen from Fig. 7, because of the simplicity of protocol and broadcasting mode, the secure connectivity of TinyADH-NCR is obviously higher than that of TinyADH-CR.

VII. Conclusion

In order to address the problem of secure key establishment in sensor networks, we first analyze the existing schemes in sensor networks and traditional networks and find the authentication of existing schemes are either partial or delayed. Then we present the idea of a fully authenticated key agreement (FAKA) and the relevant theories about FAKA are preliminarily constructed and proved. And then the idea of using FAKA and Diffie-Hellman to build a secure AKA protocol is proposed. Based on this idea, two kinds of schemes in the FADH prototype — ADH with Challenge-Response (ADH-CR) and ADH without Challenge-Response (ADH) are constructed. With the FADH protocol prototype, we construct two fully authenticated Diffie-Hellman protocols — TinyADH-NCR and TinyADH-CR, which are suitable for the characteristics of WSNs and implemented on TinyOS in nesC. Compared with the existing similar solutions, TinyADH has lower communication overload, is easier to implement into existing standards, and more secure under equivalent computational complexity. The experimental results show that the use of CR or non-CR has little influence on the time, the average time is about 54 seconds (CR: 52.7s, non-CR: 56.5s), however the simulation results show that the secure connectivity of TinyADH-NCR is obviously higher than that of TinyADH-CR. The simulation results on retransmission also show that sending 2 New1 packets can achieve higher secure connection rates and cost-effective. Currently, certificateless scheme is very attractive, so certificateless FAKA protocol will be our further work.

Acknowledgment

The authors are would like to thank all the reviewers for their insightful comments and kind guidances to improve the article. And also thanks to these free platforms such as TinyECC2.0 and TinyOS.

References

[1] K.-A. Shim, "A survey of public-key cryptographic primitives in wireless sensor networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 577–601, 1st Quart., 2016.

[2] D. J. Malan, M. Welsh, and M. D. Smith, "A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography," in *Proc. 1st IEEE Int. Conf. Sensor Ad Hoc Commun. Netw.*, Oct. 2004, pp. 71–80.

[3] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *Proc. 10th ACM Conf. Comput. Commun. Secur. (CCS)*, 2003, pp. 52–61.

[4] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Proc. 19th Int. Conf. Data Eng.*, May 2003, pp. 197–213.

[5] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A pairwise key predistribution scheme for wireless sensor networks," in *Proc. 10th ACM Conf. Comput. Commun. Secur. (CCS)*, Oct. 2003, pp. 42–51.

[6] H. Wang, B. Sheng, C. C. Tan, and Q. Li, "Public-key based access control in sensornet," *Wireless Netw.*, vol. 17, no. 5, pp. 1217–1234, Jul. 2011.

[7] S. H. Seo, J. Won, S. Sultana, and E. Bertino, "Effective key management in dynamic wireless sensor networks," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 2, pp. 371–383, Feb. 2015.

[8] I. Nadir, W. K. Zegeye, F. Moazzami, and Y. Astatke, "Establishing symmetric pairwise-keys using public-key cryptography in wireless sensor networks (WSN)," in *Proc. IEEE 7th Annu. Ubiquitous Comput., Electron. Mobile Commun. Conf. (UEMCON)*, Oct. 2016, pp. 1–6.

[9] N. Gura, A. Patel, A. Wander, H. Eberle, and S. C. Shantz, "Comparing elliptic curve cryptography and RSA on 8-bit CPUs," in *Proc. Cryptograph. Hardw. Embedded Syst. (CHES), Int. Workshop*, Cambridge, MA, USA, Aug. 2004, pp. 119–132.

[10] L. Law, A. Menezes, M. Qu, J. Solinas, and S. Vanstone, "An efficient protocol for authenticated key agreement," *Des., Codes Cryptogr.*, vol. 28, no. 2, pp. 119–134, 2003.

[11] C. G. Gnther, *An Identity-Based Key-Exchange Protocol*. Berlin, Germany: Springer, 1989.

[12] A. J. Menezes, S. A. Vanstone, and P. C. V. Oorschot, *Handbook of Applied Cryptography*. Boca Raton, FL, USA: CRC Press, 1997.

[13] D. G. Park, C. Boyd, and S. J. Moon, "Forward secrecy and its application to future mobile communications security," in *Proc. Int. Workshop Pract. Theory Public Key Cryptogr., Public Key Cryptography*, 2000, pp. 433–445.

[14] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. IT-22, no. 6, pp. 644–654, Nov. 1976.

[15] W. Diffie, P. C. van Oorschot, and M. J. Wiener, "Authentication and authenticated key exchanges," *Des. Codes Cryptogr.*, vol. 2, no. 2, pp. 107–125, Jun. 1992.

[16] H. Krawczyk, "HMQV: A high-performance secure Diffie–Hellman protocol," in *Proc. Int. Conf. Adv. Cryptol.*, 2005, pp. 546–566.

[17] A. S. Wander, N. Gura, H. Eberle, V. Gupta, and S. Chang Shantz, "Energy analysis of public-key cryptography for wireless sensor networks," in *Proc. 3rd IEEE Int. Conf. Pervasive Comput. Commun.*, Mar. 2005, pp. 324–328.

[18] S. Blake-Wilson and A. Menezes, *Authenticated Diffie–Hellman Key Agreement Protocols*. Berlin, Germany: Springer, 1999.

[19] H. Krawczyk, "SIGMA: The 'SIGn-and-MAc' approach to authenticated Diffie–Hellman and its use in the IKE protocols," in *Proc. Annu. Int. Cryptol. Conf.*, vol. 2729, no. 2, 2003, pp. 400–425.

[20] L. Harn, W. J. Hsin, and M. Mehta, "Authenticated Diffie-Hellman key agreement protocol using a single cryptographic assumption," *IEE Proc. Commun.*, vol. 152, no. 4, pp. 404–410, Aug. 2005.

[21] E. J. Yoon, W. S. Lee, and K. Y. Yoo, *Improving Single-Assumption Authenticated Diffie–Hellman Key Agreement Protocols*. Berlin, Germany: Springer, 2007.

[22] J. Mišić, "Traffic and energy consumption of an IEEE 802.15.4 network in the presence of authenticated, ECC Diffie–Hellman ephemeral key exchange," *Comput. Netw.*, vol. 52, no. 11, pp. 2227–2236, Aug. 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S138912860800128X

[23] R. Gennaro, H. Krawczyk, and T. Rabin, *Okamoto-Tanaka Revisited: Fully Authenticated Diffie–Hellman with Minimal Overhead*. Berlin, Germany: Springer, 2010.

[24] D. He, S. Padhye, and J. Chen, "An efficient certificateless two-party authenticated key agreement protocol," *Comput. Math. Appl.*, vol. 64, no. 6, pp. 1914–1926, 2012.

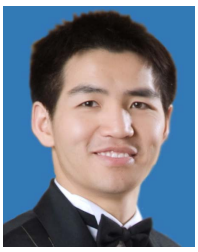
[25] A. C.-C. Yao and Y. Zhao, "OAKE: A new family of implicitly authenticated Diffie–Hellman protocols," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2013, pp. 1113–1128.
[26] R. Yasmin, G. Wang, and G. Wang, *Provable Security of a Pairing-Free One-Pass Authenticated Key Establishment Protocol for Wireless Sensor Networks*. Cham, Switzerland: Springer, 2014.
[27] H. Tu, N. Kumar, J. Kim, and J. Seo, "A strongly secure pairing-free certificateless authenticated key agreement protocol suitable for smart media and mobile environments," *Multimedia Tools Appl.*, vol. 74, no. 16, pp. 1–13, 2015.
[28] H. Sun, Q. Wen, and W. Li, "A strongly secure pairing-free certificateless authenticated key agreement protocol under the CDH assumption," *Sci. China Inf. Sci.*, vol. 59, no. 3, p. 32109, Mar. 2016.
[29] M. E. S. Saeed, Q.-Y. Liu, G. Y. Tian, B. Gao, and F. Li, "AKAIoTs: Authenticated key agreement for Internet of Things," *Wireless Netw.*, vol. 25, no. 6, pp. 3081–3101, Mar. 2018.
[30] F. Sun and Y. He, "An authenticated key agreement method based on cross interlock mechanism and its implementation," China Patent CN 110 971 401 A, Nov. 19, 2019, pp. 1–18. [Online]. Available: http://pss-system.cnipa.gov.cn/
[31] L. Deng and R. Gao, "Certificateless two-party authenticated key agreement scheme for smart grid," *Inf. Sci.*, vol. 543, pp. 143–156, Jan. 2021.
[32] F. Sun, Y. He, X. Zhang, and Q. Li, "An authentication key establishment method based on lightweight certificate and its implementation for sensor networks," China Patent CN 110 912 692 A, Nov. 19, 2019, pp. 1–12. [Online]. Available: http://pss-system.cnipa.gov.cn/
[33] F. Sun, S. He, X. Zhang, F. Shen, Q. Li, and Y. He, "TinyAKE: A more practicable and trustable scheme for authenticated key establishment in WSNs," 2021, *arXiv:2104.01907*.
[34] M. A. Strangio, "Efficient Diffie–Hellmann two-party key agreement protocols based on elliptic curves," in *Proc. ACM Symp. Appl. Comput. (SAC)*, 2005, pp. 324–331.
[35] A. P. Sarr, P. Elbaz-Vincent, and J. C. Bajard, *A Secure Efficient Authenticated Diffie–Hellman Protocol*. Berlin, Germany: Springer, 2009.
[36] C.-T. Hsueh, C.-Y. Wen, and Y.-C. Ouyang, "A secure scheme against power exhausting attacks in hierarchical wireless sensor networks," *IEEE Sensors J.*, vol. 15, no. 6, pp. 3590–3602, Jun. 2015.
[37] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient security mechanisms for large-scale distributed sensor networks," in *Proc. 10th ACM Conf. Comput. Commun. Secur. (CCS)*, 2003, pp. 62–72.
[38] L. B. Oliveira *et al.*, "TinyPBC: Pairings for authenticated identity-based non-interactive key distribution in sensor networks," *Comput. Commun.*, vol. 34, no. 3, pp. 485–493, Mar. 2011.
[39] G. Avoine, S. Canard, and L. Ferreira, "Symmetric-key authenticated key exchange (SAKE) with perfect forward secrecy," in *Proc. Cryptographers Track RSA Conf.* in Lecture Notes in Computer Science, vol. 12006, S. Jarecki, Ed. San Francisco, CA, USA: Springer, Feb. 2020, pp. 199–224, doi: 10.1007/978-3-030-40186-3_10.
[40] R. Canetti and H. Krawczyk, "Analysis of key-exchange protocols and their use for building secure channels," in *Proc. EUROCRYPT*, vol. 2045, 2001, pp. 453–474.
[41] E. Bresson, O. Chevassut, and D. Pointcheval, "Provably secure authenticated group Diffie–Hellman key exchange," *ACM Trans. Inf. Syst. Secur.*, vol. 10, no. 3, p. 10, Jul. 2007.
[42] B. LaMacchia, K. Lauter, and A. Mityagin, "Stronger security of authenticated key exchange," in *Provable Security*, W. Susilo, J. K. Liu, and Y. Mu, Eds. Berlin, Germany: Springer Berlin Heidelberg, 2007, pp. 1–16.
[43] A. J. Menezes, M. Qu, and S. A. Vanstone, "Some new key agreement protocols providing implicit authentication," in *Proc. Workshop Sel. Areas Cryptogr. (SAC)*, 1995, pp. 22–32.
[44] F. Sun, "Research on key management protocols for hierarchical wireless sensor networks," M.S. thesis, College Comput. Commun., Hunan Univ., Changsha, China, 2007.
[45] D. R. Stinson and M. B. Paterson, *Cryptography: Theory and Practice*, 4th ed. Boca Raton, FL, USA: CRC Press, 2017.
[46] P. Tague, M. Li, and R. Poovendran, "Mitigation of control channel jamming under node capture attacks," *IEEE Trans. Mobile Comput.*, vol. 8, no. 9, pp. 1221–1234, Sep. 2009.
[47] D. Gay, P. Levis, R. V. Behren, M. Welsh, E. Brewer, and D. Culler, "The nesC language: A holistic approach to networked embedded systems," in *Proc. ACM SIGPLAN Conf. Program. Lang. Design Implement.*, 2003, pp. 1–11.

Fajun Sun received the M.S. degree from the College of Computer and Communication, Hunan University, Changsha, China, in 2007. He is currently pursuing the Ph.D. degree with the School of Computer Science, Wuhan University, Wuhan, China. From 2007 to 2016, he was a Lecturer, then an Associate Professor with the School of Mathematics and Computing Science, Huaihua University, Huaihua, China. His main research interests include trusted sensor networks, distributed computing, and Internet of Things.

Selena He (Member, IEEE) received the B.S. degree in electronic engineering from the Wuhan Institute of Technology, Wuhan, China, in 2002, the M.S. and Ph.D. degrees in wireless networking from the Department of Computer Science, Georgia State University, Atlanta, GA, USA, in 2012, and the M.S. degree in computer science from Utah State University, Logan, UT, USA, in 2003. She is currently an Associate Professor with the Department of Computer Science at Kennesaw State University, Kennesaw, GA, USA. Her research interests include cyber physical systems, cyber security, wireless networking, and social networks.

Xiaotong Zhang received the B.S. and M.S. degrees in computer software and theory from Wuhan University, Wuhan, China, in 2017, where he is currently pursuing the Ph.D. degree with the School of Computer Science. His main research interests include network theory, artificial neural networks, and social networks.

Jun Zhang received the Ph.D. degree from the School of Computer Science, Wuhan University, China. He is currently an Associate Professor with the School of Information Engineering, East China University of Technology, China. He is also a Senior Member at Chinese Computer Federation. His current research interests include computer architecture, big data technology, and high performance computing.



Qingan Li received the B.S. and Ph.D. degrees from the School of Computer Science, Wuhan University, Wuhan, China, in 2008 and 2013, respectively. He is currently an Associate Professor with the School of Computer Science, Wuhan University. His current research interests include compiler optimization, program analysis, embedded systems, and the Internet of Things (IoT).

Yanxiang He received the B.S. and M.S. degrees from the Department of Mathematics, Wuhan University, Wuhan, China, in 1973 and 1975, respectively, the M.S. degree from the Department of Computer and Information Science, University of Oregon, Eugene, OR, USA, in 1986, and the Ph.D. degree from the Computer School, Wuhan University, in 1999. He is currently a Professor and a Ph.D. supervisor with the School of Computer Science, Wuhan University. He has published more than 200 journals and conference papers, five monographs, 12 textbooks, and teaching reference books. His research interests include trustworthy software, distributed parallel processing, and high performance computing. He received the Famous Teacher Award of China in 2009.