Adapting TLS Handshake Protocol for Heterogenous IP-Based WSN using Identity Based Cryptography

Rania Mzid[1], Manel Boujelben[1], Habib Youssef[2], and Mohamed Abid[1]
[1]CES Research Unit, ENIS, University of Sfax, Tunisia
[2]PRINCE Research Unit, ISITCom Hammam Sousse, University of Sousse, Tunisia

Abstract—**IP-based Wireless Sensor Networks (IP-based WSNs) combine IPv6 technology with WSNs to create a global sensor network infrastructure. However, wireless radio access and Internet connectivity make end-to-end security urgently needed by security critical WSN applications. Transport Layer Security (TLS) is considered as a suitable solution to ensure such security. However, the certificate-based mechanism used by the TLS handshake protocol has a complex certificate management overhead and long handshake latency. Identity Based Cryptography (IBC) provides a viable alternative to the use of certificates. In this paper, we propose two improved TLS handshake protocols for IP-based WSNs using IBC. The first uses IBC and Elliptic curve Diffie Hellman (ECDH) protocol for key exchange and agreement while the second uses a variant of IBC based on Elliptic Curve Cryptography (ECC) and bilinear pairing. Security analysis shows that improved TLS ensures security requirements for IP-based WSN. AVISPA tool is used to validate the proposed improvements. In addition, performance analysis shows that TLS handshake protocol using IBC gives better performance in terms of latency and energy consumption.**

Keywords–IP-based WSN; 6lowPAN; end-to-end security; TLS handshake protocol; Identity Based Cryptography; bilinear pairing

I. INTRODUCTION

Wireless Sensor Networks (WSNs) [1] are a variant of Low Power Wireless Personal Area Networks (LowPANs). Considerable efforts are being deployed to integrate LowPANs with other wired and wireless IP networks, in order to make use of pervasive nature and existing infrastructure associated with IP technologies. The Internet Engineering Task Force (IETF) is standardizing the deployment of IPv6 over LowPANs through a working group known as 6lowPAN.

6lowPAN has introduced the notion of IP-based Wireless Sensor Networks (IP-based WSNs). Indeed, the emergence of IP-based WSNs makes the need to define security mechanisms more and more important. Although IEEE 802.15.4 provides link level security, application level security is also needed for IP-based WSNs[2]. However, currently, IP-based WSNs does not provide application level security because it is commonly recognized that it results in too much overhead. Transport layer security (TLS) has been identified as a good solution to ensure such security. However, due to its reliance on public-key cryptography and certificate use, TLS is considered too "heavy-weight" for highly constrained embedded devices like the WSN motes.

In this paper, we aim to take advantage of security services offered by TLS while making it more adequate for IP-based WSN. We suggest two improvements to TLS handshake protocol using Identity Based Cryptography (IBC). The first uses IBC instead of certificate and Elliptic Curve Diffie Hellman (ECDH) for key exchange and agreement, while the second uses a variant of IBC based on Elliptic Curve Cryptography (ECC) and bilinear pairing. The introduction of IBC eliminates the need for the generation and transport of certificates, thus inducing better performance in terms of latency and energy consumption. It also provides basic security requirements such as confidentiality, authentication and integrity.

The remainder of this paper is organized as follows. Section II gives an overview of TLS handshake protocol and 6lowPAN. Related work is presented in Section III. In section IV we present two improved TLS handshake protocols using IBC. Security and performance analysis results are discussed in Section V. Finally, Section VI concludes the paper.

II. AN OVERVIEW OF 6LOWPAN AND TLS

A. 6lowPAN

1) 6lowPAN adaptation layer: 6lowPAN is a working group within IETF concerned with the specification of transmitting IPv6 packets over IEEE 802.15.4 networks. Currently, there are two 6lowPAN internet drafts,[2] and[3]. The former gives an overview, motivation and a problem statement while the latter dives into technical details and defines the frame format for the transmission of IPv6 packets over 802.15.4 networks. As IPv6 requires support of packet sizes larger than the maximum 802.15.4 frame size, 6lowPAN adaptation layer allows fragmentation/reassembly. This layer provides also additional functionality beyond just fragmentation. Mechanisms supporting mesh networking are defined and a dispatch value before the actual payload allows for header compression in higher layers by indicating what type of datagram follows.

2) 6lowPAN security analysis: 6lowPAN security analysis draft [4] discusses possible threats and security options for 6lowPAN. This document summarizes attacks and threats against user and data security in 6lowPAN caused by its wireless radio access and connectivity to the Internet. Although it does not provide any solution, several services from the real world were implemented using 6lowPAN technology.

B. TLS and handshake protocol

TLS comprises two main components; *handshake protocol* and *record protocol*. The former is responsible for negotiating and establishing secure connections, while the latter is responsible for securing data transmission. We present in Figure 1 the TLS protocol stack.

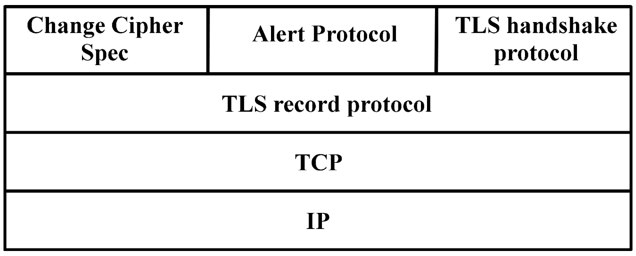

Figure 1. Architecture of TLS protocol

TLS handshake protocol is the core of TLS protocol. During the establishment of a TLS session, handshake protocol negotiates security parameters, and agrees on necessary session keys to protect the communication traffic. Figure 2 presents the message flow for a full handshake.

During a full handshake, the client sends a client hello message. The server must then respond with a server hello message to agree on security parameters, e.g. method for key exchange, cipher suit, etc.

Following the hello messages, the server will send its certificate. Additionally, a server key exchange message may be sent. If the server is authenticated, it may request a certificate from the client. Next, the server will send the server hello done message. This message indicates that the hello-message phase of the handshake is complete.

The server will then wait for a client response. If the server has sent a certificate request message, the client must send the certificate message.

The client key exchange message is now sent, and the content of that message will depend on the public key algorithm selected in the hello phase.

If the client has sent a certificate with signing ability, a digitally- signed certificate verify message is sent to explicitly verify the certificate.

At this point, a change cipher spec message is sent by the client, and the client copies the pending Cipher Spec into the current Cipher Spec.

The client then immediately sends the finished message under the new algorithms and keys. In response, the server will send its own change cipher spec message, transfer the pending to the current Cipher Spec, and send its finished message under the new Cipher Spec.

At this point, the handshake is complete, the session turns into the data transfer phase. The record protocol uses symmetric-key algorithms for bulk encryption. Thus, application data can be securely transported.

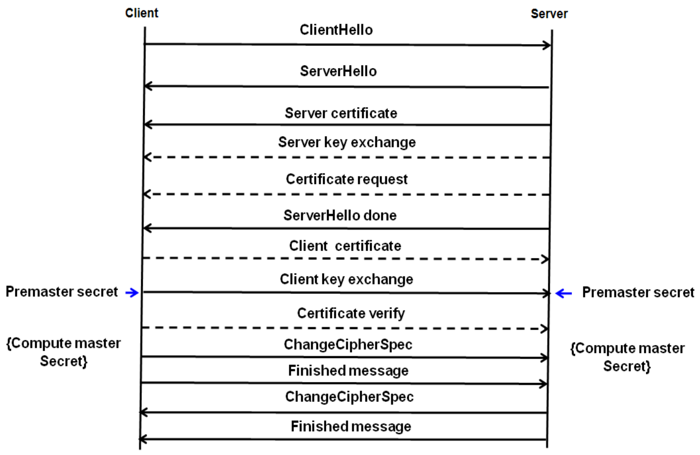

Figure 2. Message flow for a full handshake

III. Related Work

Recent progress on Elliptic Curve Cryptography (ECC) [5] provides new opportunities to utilize public-key cryptography in sensor networks. Comparing with RSA, ECC achieves equivalent security strength with smaller keys. For instance, 163-bit ECC is as secure as 1024-bit RSA [6]. ECC has been used in TLS in order to reduce the complexity of this protocol, and related protocols have been standardized [7].

For WSNs, the idea of ensuring end-to-end security has been relatively widely considered in the literature. Sizzle[8] is an SSL-based end-to-end security architecture for the embedded Internet. It uses ECDH[9] for key exchange and the Elliptic Curve Digital Signature Algorithm (ECDSA) [10] for authentication. Sizzle has been proposed for WSNs. However, Sizzle adopts a proxy based architecture, where a TCP/IP connection terminates at the gateway. Further, a sensor network adopting Sizzle uses its own protocol. Sizzle has no additional resource load on IP-based WSN protocol stack, since it uses a lightweight TCP/IP stack and 6lowPAN adaptation layer. In [11], SSL-based Lightweight Security of IP-based WSNs has been proposed. Like Sizzle, the protocol proposed in [11] uses ECDH [9] for key exchange and ECDSA [10] for authentication. Furthermore, it ensures end-to-end security for IP-based WSNs and can operate over 6lowPAN protocol stack.

IV. Improved TLS handshake protocol for heterogenous IP-based WSN using IBC

In this section, we will present the network model and the proposed improvements to make TLS handshake protocol more adequate for IP-based WSNs. However, first we will introduce some basic concepts regarding elliptic curves [5], IBC and pairing function [12] to better understand our proposal.

A. *Elliptic curves, IBC and bilinear pairing*

1) Elliptic curves: Elliptic curves are usually defined over binary fields F_{2^m} (m ≥ 1), or over prime fields F_q, q>3. Let q be a big primitive number. An elliptic curve E_q defined in a finite field F_q, is given by:

$$\mathrm{E_q}: y^2 = x^3 + ax + b \qquad a, b \in \mathrm{F_q} \qquad (1)$$

$\mathrm{E_q}$ is then the set of point P(x, y) where $x, y \in \mathrm{F_q}$ verifying the equation (1). The security of elliptic curve cryptosystems is based on the difficulty of solving the Discrete Logarithm Problem (DLP) on the Elliptic curve group called ECDLP [13]. In fact, ECDLP defined in $\mathrm{E_q}$ is the following: If $P \in \mathrm{E_q}$ and $kP \in \mathrm{E_q}$, ECDLP problem results in the difficulty of determining k knowing P and kP.

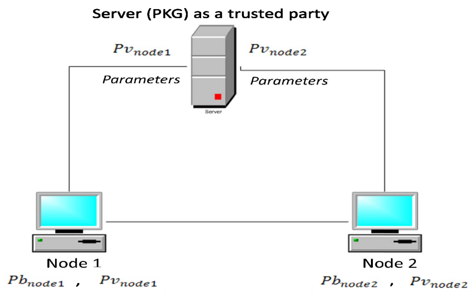
Server (PKG) as a trusted party
Pv_{node1}
Parameters
Pv_{node2}
Parameters
Server
Node 1
Pb_{node1} , Pv_{node1}
Node 2
Pb_{node2} , Pv_{node2}

Figure 3. IBC infrastructure

2) IBC and bilinear pairing

a) IBC: It was proposed by Shamir in 1984 [14]. This type of cryptography has been introduced to eliminate the certificate use. Hence, minimize the transmission of messages as it is the most expensive task for tiny devices in terms of energy. A Private Key Generator (PKG), as a trusted party, must be implemented for IBC instead of the certification authority necessary for public key infrastructure (PKI) *(*Figure 3). Its role is to generate the private keys for communicating nodes. Each node is pre-loaded with a unique identity and a unique secret key not shared with any other node.

b) Bilinear pairing: It was in 2001 that Boneh and Franklin proposed the first scheme using IBC [15]. This scheme is based on bilinear pairing. We introduce *pairing* briefly: Let G_1 denote a cyclic additive group of some large prime order q and G_2 a cyclic multiplicative group of the same order. A pairing is a map: $e: G_1 \times G_1 \rightarrow G_2$ and has an important property that is bilinearity; if $P, Q, R \in G_1$ and $a \in Z_q^*$, $e(P+Q,R) = e(P,R).e(Q,R)$, $e(P,Q+R) = e(P,Q).e(P,R)$ and $e(aP,Q) = e(P,aQ) = e(P,Q)^a$. Typically, the map is derived from the Weil, or the Tate, or other pairings on an elliptic curve over a finite field. IBC based on bilinear pairing is used to compute a common secret between two interlocutors to establish a confidential channel between them without any messages exchange.

B. *Network Model and assumptions*

The network model that we have considered is based on 6lowPAN environment (Figure 4). We consider a network composed by a set of IP subnets. Each subnet contains multiple 6lowPAN nodes. We have considered heterogeneous IP subnets. Each subnet constitutes a Wireless Sensor and Actor Network (WSAN).

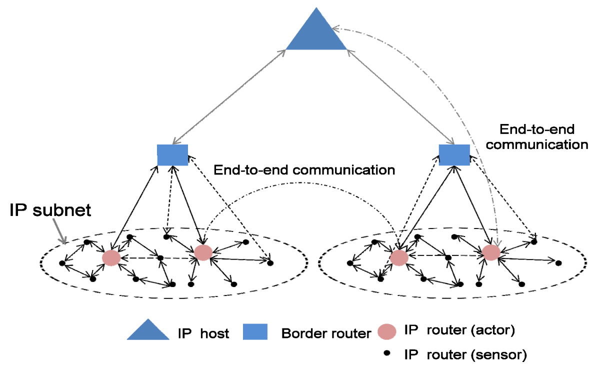

Figure 4. Network Model

WSANs are characterized by the collective effort of heterogeneous nodes called sensors and actors. Sensor nodes collect information about the physical world and forward packets to an actor node, while actor nodes take action decisions and ensure the communication with IP hosts and other actors belonging to other IP subnets. Comparing to the sensor, actor node has more memory and more ability to process. We target especially event-driven applications. When a sensor detects an event and wants to transmit data to an IP host (or actor belonging to other IP subnet) in a secure way, it sends data to the closest actor. The actor then negotiates security parameters with the remote target and establishes a secure tunnel with the latter to exchange data. We suppose the use of in-network processing techniques [16] . Each 6lowPAN node (sensor or actor) is known as an IP router. The communication between an IP router and an IP host is controlled by a border router whose role is to connect the WSAN to Internet. The mobility issue is not considered in our case. We are going to consider a static network. Each node has a static location. We suppose that we have a trusted third party that is never compromised. It will be responsible for distributing keys in the pre-deployment phase.

C. *TLS handshake protocol using IBC and ECDH*

As mentioned in the introduction, the idea of ensuring end-to-end security for WSNs has been considered [8][11]. Our contribution, compared to [11], is the use of IBC instead of certificates. We will consider two phases: pre-deployment phase and after deployment phase.

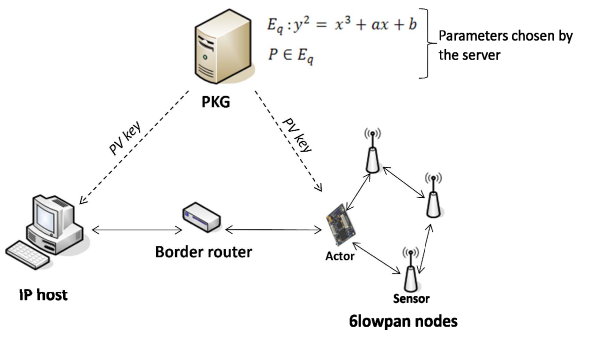

Figure 5. Pre-deployment phase

1) Pre-deployment phase: Public key is derived from the node identity (IPv6 address since we consider IP-based WSNs). PKG generates the private key of each node from its public key. This private key will be known only by the PKG and the concerned node. In addition, PKG initiates some useful parameters (Figure 5) (e.g. elliptic curve E_q and $P \in E_q$ as a generator). At the end of this phase, each actor and IP machine has a private key stored in its memory.

*2) After-deployment phas*e: We present below (Figure 6) the message flow for a Full Handshake. The improved TLS uses ECDH [9] key exchange and IBC [12] instead of certificates. The client is the 6lowPAN node and specifically an actor belonging to an IP subnet. The server can be the IP machine or an actor belonging to a different IP subnet depends on the end-to-end communication. As mentioned in the pre-deployment phase, both client and server have their private keys stored in their memories. Public key is known since it derives from the node identity (IPv6 address).

The first two messages, ClientHello and ServerHello are similar to the TLS specification. Each node generates random number denoted *ds* for the server and *dc* for the client. P is a point on the elliptic curve determined during the pre-deployment phase. Serverkeyexchange message contains *ds*P* and *ds*P* encrypted with the server private key which can provide authentication of the latter. If the client must be authenticated, the server sends the Authentication request message. ServerHellodone message indicates the end of hello phase. The client computes *dc*P* and sends it in the Clientkeyexchange message. If authentication is needed, *dc*P* will also be sent encrypted with the client private key. For the remaining messages nothing changes. We note that *ds*P* and *dc*P* are respectively sent in Serverkeyexchange message and Clientkeyexchange message in clear. I.e. No encryption is required because determining *ds* knowing *ds*P* and *P* returns to solve ECDLP problem[13]. This protocol reduces the number of messages during a full handshake from 13 to 10 when the authentication of the client is required.

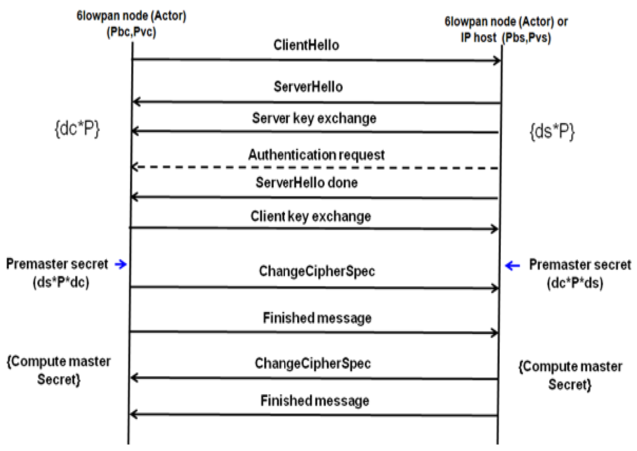

Figure 6. Message flow for full handshake protocol using IBC and ECDH

D. TLS handshake protocol using IBC based on ECC and bilinear pairing

This section describes the second proposed improvement to adapt TLS handshake protocol for IP-based WSNs. It uses IBC based on ECC and bilinear pairing. In [17], authors propose a pairing identity based key management protocol. This protocol uses bilinear pairing to establish a shared key between two nodes. Bilinear pairing allows two communicating entities to generate a shared session key without any interaction between them. Hence its use minimizes the number of messages between client and server communicating during a full handshake. This protocol consists also of two phases: pre-deployment phase and after deployment phase.

1) Pre-deployment phase: As for the previously described protocol we have a trusted party, a PKG that will never be compromised. The role of this PKG is to introduce some parameters. During this phase PKG chooses:

- Two groups G_1 and G_2, of the same prime order q. G_1 is an additive group and G_2 is a multiplicative group. G_1 is a set of points of an elliptic curve E_q with $q = 2^m$
- P an arbitrary generator of G_1
- A bilinear map e: $G_1 \times G_1 \rightarrow G_2$
- A hash function H which returns a point of an elliptic curve from an identity
- A random number $S \in Z_q^*$

The PKG uses the value of S to generate private keys of all nodes. In fact, PKG computes $H(ID_i) \in G1$ for each node i and then $Q_i = S \times H(ID_i)$. In our case ID_i is the IPv6 address. S is known only by the PKG and all other parameters are publicly known. At the end of this phase, PKG sends for each node (IP host and 6lowPAN node (actor)) a set of parameters: $< G_1, G_2, e, q, P, Q_i, H >$

2) *After-deployment phase*: The message flow for a Full Handshake protocol using bilinear pairing is presented in Figure 7.

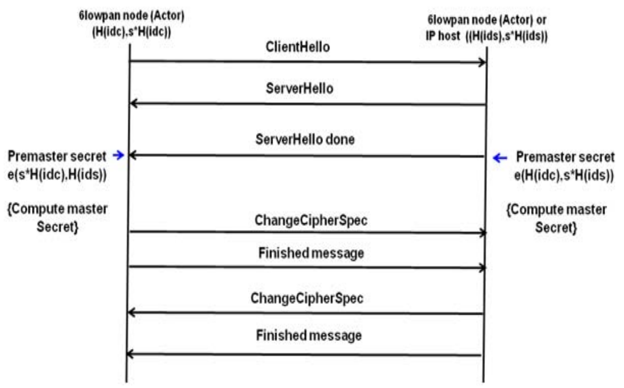
6lowpan node (Actor) (H(idc),s*H(idc))
6lowpan node (Actor) or IP host ((H(ids),s*H(ids))
ClientHello
ServerHello
ServerHello done
Premaster secret e(s*H(idc),H(ids))
Premaster secret e(H(idc),s*H(ids))
{Compute master Secret}
{Compute master Secret}
ChangeCipherSpec
Finished message
ChangeCipherSpec
Finished message

Figure 7. Message flow for a full handshake using bilinear pairing

When a client (6lowPAN node (actor)) wants to send data to a server (IP host or actor) in a secure way, it starts by sending the ClientHello message. ClientHello message must indicate the use of bilinear pairing. After receiving ServerHello message by the client, both client and server compute Premaster secret using pairing function. The properties of pairing allow computing a session key without any message exchange. The ServerHellodone indicates the end of the hello phase. For the rest of messages, nothing changes. This second improvement of the TLS handshake protocol reduces the number of messages from 13 to only 7.

V. SECURITY AND PERFORMANCE ANALYSIS

A. *Security analysis*

On the basis of security mechanisms of TLS, the proposed modifications to TLS attain several basic security properties including authentication, confidentiality and integrity.

1) *Authentication:* For TLS handshake Protocol using IBC and ECDH, authentication can be ensured by encryption with private key. This key is distributed during the pre-deployment phase by the PKG. Only trusted nodes have valid private keys. For the improved TLS handshake protocol using bilinear pairing, properties of pairing function ensure authentication. In fact, to compute a session key using pairing, two communicating nodes must have valid identities and thus authentication is ensured. By ensuring authentication, several attacks can be prevented such as DoS attacks and impersonation attacks.

2) *Confidentiality:* It is ensured by symmetric cryptography using a shared key computed during TLS handshake protocol. For the TLS handshake protocol using IBC and ECDH, the secrecy of the session key results from the difficulty to solve ECDLP problem [13]. For the TLS handshake protocol using bilinear pairing the computation of the shared key is done without any message exchange. So, the session key in this case is shared only by the end points and can never be known by an intruder. By ensuring confidentiality, we can prevent from sniffing as passive attack. The attacker is unable to intercept the data exchanged between the client and the server.

3) *Integrity:* The integrity is ensured by applying a MAC using the shared secret negotiated during the handshake protocol. By ensuring integrity, we can prevent against multiple attacks such as the man in the middle attack.

B. *Verification and performance analysis*

To validate security protocols and verify their robustness against typical security flaws and attacks, designers resort to formal validation tools. Such tools provide a formal specification language to construct a validation model of the protocol and the desired correctness properties to satisfy, together with a running environment that will automatically execute the model. This automatic formal analysis helps detect faults and correct them before proceeding to implementation. Examples of such validation tools are AVISPA [18], Hermes [19], and ProVerif [20].
In this work, we used AVISPA to verify the proposed modification to TLS handshake protocol. We chose AVISPA for many reasons: (i) It is expressive enough and allows the expression of several security properties, such as secrecy of keys, authentication, etc. (ii) It provides a user friendly specification language called HLPSL the "High Level Protocol Specification Language". (iii) It is widely used by industrial and academic circles to analyze security protocols.

AVISPA provides the possibility to verify two important security properties: (i) Secrecy of the master-secret: this property is verified by the primitive *secret (Master_secret,sec_key,{C,S})* (ii) Authentication: this property is verified by two primitives: *witness (C,S,na_nb2,Vc')* at the sender side and *request (S,C,na_nb2,Vc')* at the receiver side. The proposed IBC based TLS protocols have been successfully analyzed with respect to these two properties.

Next, we compare the TLS handshake protocol using IBC for authentication and ECDH for key exchange ($\mathrm{TLS}_{\mathrm{IBC-ECDH}}$) and the TLS handshake protocol using a variant of IBC based on ECC and bilinear pairing ($\mathrm{TLS}_{\mathrm{pairing}}$) with TLS handshake protocol using ECDSA for authentication and ECDH for key exchange ($\mathrm{TLS}_{\mathrm{ECDSA-ECDH}}$) [11]. The comparison is made in terms of storage cost, latency (handshake duration) and energy consumption. The comparison results are obtained throw modeling. We express latency and energy as a function of the time and energy, respectively, required for each modular operation in the different protocols. In [21], authors estimate the time and power consumption for most common RSA and ECC operations, i.e., signature generation and verification .They investigate four types of nodes; MICA2DOT , MICA2, Micaz and TelosB. We choose

TelosB as platform to draw different curves. Also, we focus on ECC-224 as cryptographic standard. For the time required to compute the pairing, we use the value given in[22]. In this paper the authors implement pairing on a Tmote sky. This mote is very similar to TelosB in terms of capacity.

1) Storage cost: $TLS_{ECDSA-ECDH}$ uses a public key scheme. So each node has to store its public key and its private key. However, the use of IBC makes the storage of public key useless since the public key is derived from the identity. Hence, in the two proposed schemes each node needs to store only its secret key assigned by a PKG.

2) Latency: Latency can be defined as the time required performing the negotiation between the client and the server and generating security parameters. This metric is very important especially when the application requires timely transmission of data (real time applications). We consider the case when only server authentication is required and the case when both client and server should be authenticated. Latency is expressed as the sum of time required for each operation and the transmission time. Table I and III illustrate the set of operations required to perform a full handshake respectively with and without client authentication. The transmission time is expressed as follow:

$Nb * Tr_{message} * 1/(1-PLR)$; Nb is the number of messages transmitted and it depends on the protocol. $Tr_{message}$ is the time required to transmit one message. $1/(1-PLR)$ is the number of possible retransmissions.

a) Without Client authentication:

TABLE I. OPERATIONS REQUIRED FOR A FULL HANDSHAKE WITHOUT CLIENT AUTHENTICATION

	$\mathbf{TLS_{ECDSA-ECDH}}$	$\mathbf{TLS_{IBC-ECDH}}$	$\mathbf{TLS_{pairing}}$
Client	$ECDSA_{verify}, ECDH_{op}$	$ECDH_{op}$	$pairing_{op}$
Server	$ECDH_{op}$	$ECDH_{op}$	$pairing_{op}$

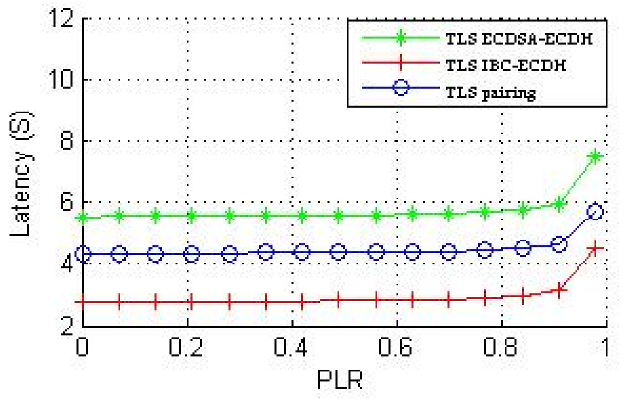

Figure 8. Handshake Latency of different TLS handshake protocols without client authentication

TABLE II. ESTIMATION OF LATENCY IN SECONDS FOR FULL HANDSHAKE WITHOUT CLIENT AUTHENTICATION (PLR=0)

	Throughput	$TLS_{ECDSA-ECDH}$	$TLS_{IBC-ECDH}$	$TLS_{pairing}$
Mica2Dot	38,4 kbps	17,85	8,99	-
TelosB	250 kbps	5,53	2,77	4,35

Table II presents an estimation of latency when the authentication of the client is not required for two different platforms (mica2Dot and TelosB) with PLR (Packet Loss Ratio) equal to 0 (i.e. we consider the ideal case; any retransmission is required).

b) With Client authentication:

TABLE III. OPERATIONS REQUIRED FOR A FULL HANDSHAKE WITH CLIENT AUTHENTICATION

	ECDSA-ECDH	IBC-ECDH	Pairing
Client	$ECDSA_{verify}, ECDH_{op}, ECDSA_{gen}$	$ECDH_{op}$	$pairing_{op}$
Server	$2 * ECDSA_{verify}, ECDH_{op}$	$ECDH_{op}$	$pairing_{op}$

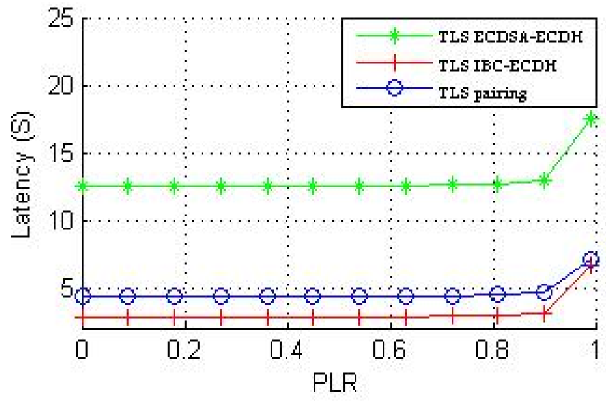

Figure 9. Handshake Latency of different TLS handshake protocols with client authentication

We can note from Figure 8 and Figure 9 that the introduction of IBC significantly improves the performance of the TLS handshake protocol in terms of latency. In fact, IBC eliminates the generation and transmission of certificates, improves performances and reduces the handshake duration. In both cases (with and without client authentication) TLS handshake using IBC and ECDH offers better performance. Indeed, when the authentication of the client is requested, the improvement is significantly better since, in this case, the introduction of IBC eliminates the certificate use in both client and server sides.

TABLE IV. HANDSHAKE LATENCY IN SECONDS FOR FULL HANDSHAKE WITH CLIENT AUTHENTICATION (PLR=0)

	Throughput	$TLS_{ECDSA-ECDH}$	$TLS_{IBC-ECDH}$	$TLS_{pairing}$
Mica2Dot	38,4 kbps	40	9,01	-
TelosB	250 kbps	12,46	2,78	4,35

3) Energy Consumption: The two proposed handshake protocols will be evaluated in terms of energy consumption. We will consider the case when the client should be authenticated and the case when the authentication of the client is not required. Like latency, the total energy consumed in expressed as a function of the energy required for each operation and the energy required for transmission. Energy required for transmission is expressed as follow:

$$E_{tr} = Nb * \left(\frac{1}{(1 - PLR)}\right) * Tr_{message} * P_{tr}$$

P_{tr}: Transmission Power; constant for the same Platform

a) Without Client authentication: Figure 10 represents the energy consumed during a full handshake as a function of PLR.

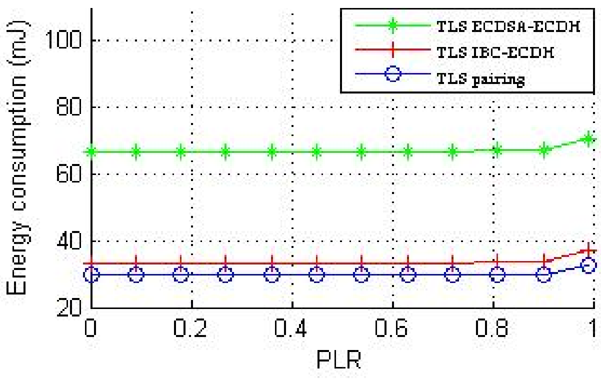

Figure 10. Energy consumption of different TLS handshake protocols without client authentication

The following table represents an estimation of the energy consumed during the full handshake protocol when the authentication of the client is not required. We will consider two different platforms (mica2Dot and TelosB) and the particular case when PLR=0:

TABLE V. ESTIMATION OF ENERGY CONSUMPTION IN (MJ) FOR FULL HANDSHAKE WITHOUT CLIENT AUTHENTICATION (PLR=0)

	Throughput	$TLS_{ECDSA-ECDH}$	$TLS_{IBC-ECDH}$	$TLS_{pairing}$
Mica2Dot	38,4 kbps	243,04	121,03	-
TelosB	250 kbps	66,82	33,27	29,56

b) With Client authentication:

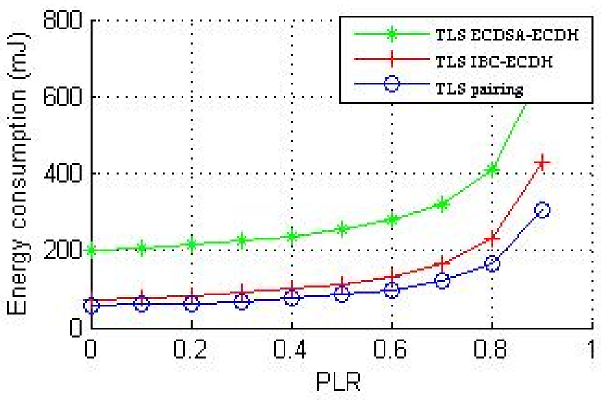

Figure 11. Energy consumption of different TLS handshake protocols with client authentication

TLS handshake protocol using pairing gives better performance in terms of energy. This comes as no surprise since we have eliminated the transport and process of certificates and significantly reduced the number of messages transmitted. It may be noted that energy consumption does not vary significantly with the PLR. This can be explained by the fact that operations based on ECC are more expensive than transmission in terms of energy and even in terms of latency.

TABLE VI. ESTIMATION OF ENERGY CONSUMPTION IN (MJ) FOR FULL HANDSHAKE WITH CLIENT AUTHENTICATION (PLR=0)

	Throughput	$TLS_{ECDSA-ECDH}$	$TLS_{IBC-ECDH}$	$TLS_{pairing}$
Mica2Dot	38,4 kbps	548,61	121,05	-
TelosB	250 kbps	66,82	33,27	29,56

VI. Conclusion

In this paper, we propose two improved TLS handshake protocols for IP-based WSNs using Identity Based Cryptography. We show that using IBC in the TLS handshake protocol induces significant performance gain. The two proposed schemes have been evaluated in terms of storage cost, latency (handshake duration) and energy consumption. The first proposed scheme gives better performance in terms of latency, whereas the second proposed scheme requires lower energy consumption. In the absence of hard timing constraints, the second proposal should be preferred.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," IEEE Communications Magazine, vol. 40, no. 8, pp. 102–114, August 2002.

[2] N. Kushalnagar, G. Montenegro, and C. Peter Pii Schumacher., "6LoWPAN: Overview, Assumptions, Problem Statement and Goals". Internet-Draft Version 08, IETF, February 2007

[3] G. Montenegro, N. Kushalnagar, D. E. Culler, and J. W. Hui, “Transmission of IPv6 Packets over IEEE 802.15.4 Networks”. Internet-Draft Version 13, IETF, April 2007.

[4] S. Park, K. Kim, W. Haddad (Ed), S. Chakrabarti, J. Laganier, ”IPv6 over Low Power WPAN Security Analysis”. Internet-Draft Version 03, IETF, July 2009.

[5] N. Koblitz, “Elliptic curve cryptosystems,” Mathematics of Computation 48, 1987, pp. 203-209.

[6] N. Gura, A .Patel, A. Wander, H. Eberle, S.C. Shantz. “Comparing elliptic curve cryptography and RSA on 8-bits CPUs”, In Proceedings of the 6th International workshop on cryptographic hardware and embedded systems, Bosten, Massachusetts, August 2004

[7] V. Gupta, S. Blake-Wilson, B. Moeller, C. Hawk, N. Bolyard., “ECC Cipher Suites for TLS”. Internet-Draft Version 10,IETF,May,2005

[8] V. Gupta et al., "Sizzle: A standards-based end-to-end security architecture for the embedded internet," PerCom 2005, Mar. 2005.

[9] ANSI X9. 62, “Elliptic Curve Key Agreement and Key Transport Prtocols,” American Bankers Association, 1999

[10] ANSI X9. 63, "The Elliptic Curve Digital Signature Algorithm," American Bankers Association, 1999.

[11] W. Jung, S. Hong, M. Ha, Y. Kim, D. Kim. “SSL-based Lightweight Security of IP-based Wireless Sensor Networks”. In Proceeding of the IEEE 23rd International Conference on Advanced Information Networking and Applications 5AINA), United Kingdom, May 2009

[12] M. Joye. “Identity Based Cryptograph”, 2009

[13] S.D. Galbaraith and N.P. Smart, “Evaluation report for Cryptrec: Security level of cryptography- ECDLP mathematical problem. Available at: www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1029_report.pdf

[14] A. Shamir, “Identity-based cryptosystems and signature schemes,” in Proc. Crypto’84, Santa Barbara, California, USA, Aug. 1984, pp. 47–54.

[15] D. Boneh, M. Franklin, “Identity-based encryption from the weil pairing”, in: Proc. Advances in Cryptology, Crypto 2001, Springer-Verlag, LNCS 2139, 2001, pp. 213–229.

[16] S. Peter, K. Piotrowski, and P. Langendoerfer, “On concealed data aggregation for wireless sensor networks”. In Proceedings of the IEEE Consumer Communications and Networking Conference, Jan. 2007.

[17] M. Boujelben, O. Cheikhrouhou, H. Youssef, M. Abid , “A Pairing Identity based Key Management Protocol for Heterogeneous Wireless Sensor Networks”, in proceeding of the First IFIP/IEEE International conference on network and service security N2S, Paris, France, 2009

[18] AVISPA Project. AVISPA protocol library. Available at http://www.avispa-project.org/

[19] L. Bozga, Y. Lakhnech and M. Périn. Hermes, a tool verifying secrecy properties of unbounded security protocols. In 15th international conference on Computer-Aided Verification (CAV’03), Lecture Notes in Computer Science, Springer Verlag, July 2003.

[20] C.J.F. Cremers. Scyther - Semantics and Verication of Security Protocols. Ph.D. dissertation, Eindhoven University of Technology, 2006.

[21] K. Piotrowski, P.Langendoerfer, S. Peter. “How Public Key Cryptography Influences Wireless Sensor Node Lifetime”. In proceeding The Fourth ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN 2006), Alexandria, Virginia, USA, October 2006.

[22] Szczechowia. P, Kargl. A, Scott. M, Collier. M “On the Application of Pairing Based Cryptography to Wireless Sensor Networks”, in Proceedings of the second ACM conference on Wireless network security SIGSAC/ACM, Zurich, Switzerland , March 16 - 19, 2009