Key Management Protocol with Implicit Certificates for IoT systems

Savio Sciancalepore
Dep. of Electrical and Information Engineering (DEI)
Politecnico di Bari, Italy
name.surname@poliba.it

Angelo Capossele
Dep. of Computer Science
Sapienza University of Rome, Italy
capossele@di.uniroma1.it

Giuseppe Piro
Dep. of Electrical and Information Engineering (DEI)
Politecnico di Bari, Italy
giuseppe.piro@poliba.it

Gennaro Boggia
Dep. of Electrical and Information Engineering (DEI)
Politecnico di Bari, Italy
gennaro.boggia@poliba.it

Giuseppe Bianchi
Dep. of Electronic Engineering
University of Rome 2 Tor Vergata, Italy
giuseppe.bianchi@uniroma2.it

ABSTRACT

This paper proposes a Key Management Protocol for mobile and industrial Internet of Things systems, targeting, at the same time, robust key negotiation, lightweight node authentication, fast re-keying, and efficient protection against replay attacks. The proposed approach pragmatically leverages widely accepted Elliptic Curve Cryptography constructions, specifically the (Elliptic Curve) "Fixed" Diffie Hellman key exchange and the (Elliptic Curve) Qu-Vanstone implicit certificates. Our value added is their suitable integration into a security protocol exchange, designed at layer 2, in the 802.15.4 protocol stack, which permits to i) avoid Elliptic Point multiplications upon rekeying of previously paired devices, and ii) support mutual authentication while securing the protocol exchange. To prove its viability, the proposed Key Management Protocol has been implemented and assessed on severely constrained devices. As expected, but made explicit and quantified by our experimental performance evaluation, the usage of implicit certificates in conjunction with an optimized message exchange yields impressive gains in terms of airtime consumption with respect to state of the art schemes.

1. INTRODUCTION

The revolutionary Internet of Things (IoT) paradigm is enabling the interaction among smart objects, pervasively diffused across the Internet [9]. In this evolving context, security risks and threats are ever more critical; as such the research community and the standardization bodies are currently working to define novel methodologies, protocols, and algorithms, in order to provide confidentiality, authentication, integrity, and availability services in mobile and industrial IoT systems [24][9].

Secure communication mainly grounds its roots in the implementation of robust Key Management Protocols (KMPs) [15]. About a decade ago, there was considerable skepticism on the feasibility of Public Key Cryptographic techniques over sensor devices, with the community largely in favor of symmetric techniques such as key pre-distribution [8]. Nowadays, many works, among which [7, 4, 19, 3] and many more, have duly assessed the viability of Elliptic Curve Cryptography (ECC) implementations even over severely constrained devices. What however remains surprising is the fact that, at the time of writing and to the best of our knowledge, most of the proposed approaches in both the standard bodies as well as from the scientific community (for more details see Sec. 2) still handle key negotiation and peer authentication via large X.509 certificates, indeed extremely expensive in terms of transmission requirements.

In this work, we describe a KMP, integrated at the layer-2 of the protocol stack, which aims at *maximal airtime savings* by natively exploiting the Elliptic Curve Qu-Vanstone (ECQV) technique for generating ultra-lightweight "implicit" certificates [6]. More specifically, our proposed protocol leverages a "fixed" Elliptic Curve Diffie Hellman (ECDH) exchange [10], with (statically assigned) public coefficients implicitly certified using ECQV. The protocol is complemented by the exchange of nonces along with the (lightweight) authentication of the exchanged message sequence, so as to guarantee mutual authentication and freshness in the key derivation (and very fast re-keying, when necessary). Our KMP has been implemented in the open source OpenWSN protocol stack [27], and its performance are preliminarily assessed in the remainder of the paper.

The rest of the paper is structured as it follows: background material is reported in Sec. 2 along with a brief discussion of related work; Sec. 3 describes the conceived KMP mechanism and provides some implementation details; Sec. 4 illustrates the theoretical and experimental evaluation of the described solution; finally, Sec. 5 closes the paper and outlines directions for future work.

IoT-Sys 2015, May 19-22, 2015, Florence, Italy.

http://dx.doi.org/10.1145/2753476.2753477 .

2. A BACKGROUND ON SECURE DEVICE PAIRING

The literature background presented in this section remarks pros and cons of the most important solutions proposed so far. Moreover, it sheds some lights on powerful approaches that can be considered for designing key negotiation schemes more suitable in mobile and industrial IoT systems.

2.1 Reference key negotiation algorithms

The pre-distribution of cryptographic keys represents the simplest approach that can be adopted for enabling security services in the IoT. It presents two main limitations: (i) in a scenario, where the same key is shared among all the nodes, the impairment of a single device compromise the security of the whole network; (ii) the idea to configure a dedicate key for each couple of devices does not scale with the network size [8]. Key agreement mechanisms may solve these issues. Most proposals are based on Diffie-Hellman (DH) and ECDH approaches [15]. As well known, built on the multiplicative group of integers modulo p, the DH algorithm founds its potentiality on the difficulty to solve the discrete logarithm problem. In ECDH, instead, the secret is negotiated through Elliptic Curve Cryptography (ECC) primitives and same security levels offered by DH can be achieved with shorter keys.

2.2 Authentication of communicating peers

Node authentication techniques are introduced for preventing Man-In-The-Middle (MITM) attacks during the key negotiation procedure. To this aim, devices involved in a KMP session need to exchange specific data structures (i.e., a certificate) to bind and authenticate their identities with public keys.

All the contributions discussed in both research community and IETF standardization bodies (like [4] [16] [17] [19] [20][25][22]) make use of X.509 certificates [11], that contain, among other parameters, the device identity, its public key, and an explicit signature, provided by a trusted Certification Authority (CA). Unfortunately, the size of such certificates is too large. As example, focusing on a 40 bytes long ECC public key, the *OpenSSL* tool [1] generates a X.509 certificate with a size equal to 864 byts; of course, this value tends to increase with the length of the key. As a consequence, their transmission requires significant bandwidth usage and brings to high latencies and notably energy wastefullness; thus it cannot be considered well suitable for networks made up of constrained devices.

An effective alternative can be the adoption of implicit certificates, that bind the identity of a node and its public key within a single data structure and to certify the authenticity of this relation without an explicit signature [10]. Thanks to the significant reduction of the certificate size and the transmission requirements (expressed in terms of bandwidth, latency, and energy consumption), implicit certificates can be considered a powerful technique for conceiving enhanced KMP schemes for IoT systems.

The ECQV algorithm is at the basis of the generation of implicit certificates [6]. Such a scheme uses a fixed elliptic curve with a n-th order generator G, and an arbitrary hashing function $H()$. Moreover, let C and c be the public and the private keys of the CA, respectively. Before asking for a certificate, the device I_A generates a random positive integer r_A; then, it computes a point R_A on the chosen elliptic curve, $R_A = r_A \cdot G$, and sends it to the CA. The CA extracts a random positive integer k, computes $k \cdot G$ (another point on the elliptic curve), generates the implicit certificate,P_A, and the implicit signature, γ_A, by using:

$$\begin{cases} P_A = R_A + k \cdot G \\ \gamma_A = c + k \cdot H(P_A, I_A) \end{cases} \quad (1)$$

Now, it sends them to the device. Then, it can generate its private key, $s_{v,A}$, and its public key, $P_{b,A}$, by:

$$\begin{cases} s_{v,A} = \gamma_A + r_A \cdot H(P_A, I_A) \\ P_{b,A} = s_{v,A} \cdot G \end{cases} \quad (2)$$

As well-known, the most important strength of ECQV is that the public key of a given device can be computed by any other third-party starting from the knowledge of the implicit certificate and the public key of the authority:

$$P_{b,A} = s_{v,A} \cdot G = C + P_A \cdot H(P_A, I_A). \quad (3)$$

At the time of this writing, implicit certificates are only used in the Certificate Based Key Establishment (CBKE) protocol, integrated in the ZigBee IP specifications, for managing the authentication of nodes during the join procedure [1].

2.3 Re-keying mechanisms

Re-keying mechanisms are generally adopted when it is necessary to reduce the lifetime of a specific key during a communication session. Despite lightweight mechanisms for the IoT have been proposed in [7] and [13], their jointly adoption with ECC and ECQV has been not yet discussed in literature.

3. THE PROPOSED KMP ALGORITHM

Without loss of generality, we suppose that implicit certificates are preloaded in each device by the network administrator before the deployment of the network.

In its main rational, the developed *KMP* scheme is based on the exchange of four different logical messages. The first two messages carry the key materials (i.e., the ECQV implicit certificate and a nonce). ECQV implicit certificates jointly offer authentication and key agreement services in the sense that each node is able to compute, through a fixed ECDH mechanism, a shared secret starting from an authenticated public key. The latest two messages, instead, are exchanged for finalizing the mutual authentication. To prevent *replay* attacks on the second part of the protocol, the authentication field stored in these messages is computed by also considering the nonces initially exchanged. All the KMP operations are handled at the layer-2 of the protocol stack based on the IEEE 802.15.4 technology [12]. Moreover, the secret negotiated during the procedure will be adopted for generating key materials to use with the CCM* algorithm, which represents the cryptography primitive of the IEEE 802.15.4 standard. As reported in Fig. 1, in details the protocol considers the following steps:

1. node A sends a first message with its implicit certificate, P_A, and a nonce, ρ_A.

[1] https://www.openssl.org/

2. Nodes B evaluates the public key of the remote device, $P_{b,A}$, and computes the shared secret, P, as described in the ECDH protocol:

$$P = s_{v,B} \cdot P_{b,A}. \quad (4)$$

3. B sends a first message with its implicit certificate, P_B, and a nonce, ρ_B.

4. A evaluates the public key of the remote device, $P_{b,B}$, and computes the shared secret P, as described in the ECDH protocol:

$$P = s_{v,A} \cdot P_{b,B}. \quad (5)$$

5. A and B use a Key Derivation Function (KDF) for generating the *Pre Link Key*, P_K, adopted for authentication purposes.

6. To prove the possession of the *Pre Link Key*, node A computes, by using Eq. (6), the authentication field, α_A, and sends it to the remote node.

$$\alpha_A = Auth(P_K, (P_A, P_B, \rho_A, \rho_B)). \quad (6)$$

Note that the $Auth()$ operator refers to a generic authentication algorithm, which could be adopted without any limitation.

7. At the same time, the same operation is handled by node B, which computes, by using Eq. (7), the authentication field, α_B, and sends it to the remote node.

$$\alpha_B = Auth(P_K, (P_B, P_A, \rho_B, \rho_A)). \quad (7)$$

8. Nodes A and B finalize the KMP procedure by verifying the correctness of received authentication fields.

9. For each i-th group of block ciphers handled by the CCM* algorithm, the KDF is used to generate the *Link Key*, L_k, really used to protect MAC frames, that is: $L_k = KDF(i, P_K, \rho_A, \rho_B)$.

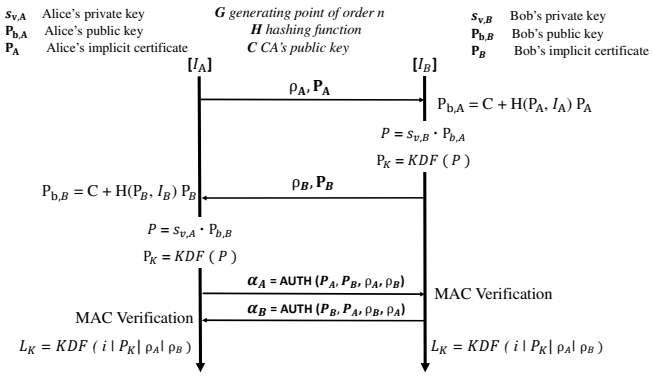

Figure 1: Key negotiation protocol.

It is very important to note that the conceived KMP approach intrinsically supports fast opportunistic re-keying. In fact, based on the fixed ECDH mechanism, the key negotiation algorithm always produces the same key for a given couple of devices. Hence, the computation of the neighbor public key and the derivation of the *Pre Link Key* can be avoided when nodes involved in the KMP mechanism have already negotiated the key in the past and have stored each value in the cache. In this case, even if the protocol still provides the exchanging of four messages, some greedy operations are not executed anymore. Obviously, the robustness against *replay* attacks can be still guaranteed by using a new set of nonces.

Since the designed KMP efficiently integrates a set of well-known approaches, its security can be demonstrated by using other existing analysis. First, the usage of the fixed ECDH algorithm ensures the secrecy of the negotiated key [14]. Moreover, ECQV implicit certificates, binding a public key to its owner in a trusted way, make the proposed strategy robust against MITM attacks [5]. Furthermore, it is possible to demonstrate that the mutual authentication scheme implemented in the second part of the protocol protects the entire approach against replay attacks. The latest two messages offer the same functionalities of *Finished* message in the Transport Layer Security (TLS) protocol. Similarly to TLS, the latest two messages of the proposed KMP carry an authentication field that is computed by considering all values exchanged before. As a consequence, the validity of the adopted mutual authentication scheme can be proved by using the same security proof as for the TLS protocol [18].

3.1 Implementation details

The KMP scheme described in the previous subsection has been implemented within the OpenWSN protocol stack [27], on top of the security extension recently presented in [21]. The experimental platform considered in this work is based on TelosB motes. To work with this really constrained devices, a number of issues have been properly solved, as-discussed below.

3.1.1 KMP messages

OpenWSN implements a protocol stack recently developed for TSCH-based networks [2]. Taken into account its structure, attributes and methods related to the devised KMP have been implemented at the MAC layer. To handle the key negotiation phase, high-level commands have been mapped into specific *Header Information Elements*. The *Crypto Information Element* is used to deliver the key materials, e.g., the implicit certificate and a random number. The *Authentication Information Element* is used to store the value of the authentication field.

3.1.2 Elliptic Curve Operations

Sensor nodes are usually equipped with simple and cheap microcontrollers. Hence, to efficiently optimize the implementation of cryptographic operations, reduce their computational complexity, and provide an acceptable security level, specific optimizations have been tuned with a balanced tradeoff between energy efficiency and RAM/ROM consumption.

The first type of optimization refers to modular arithmetic on large integer. Working on devices with 16bit registers, large integers are implemented using arrays. When operations (like multiplication and squaring) are used, an efficient use of registers should be maximized. Assembly code routines specifically optimized for the TelosB platform have been deployed [14], with an improved use of registers. Furthermore, Barrett Reduction method has been used, that is an algorithm to perform faster modular reductions of a large integer replacing divisions with multiplications. By precom-

puting some values based on a fixed module, the computation time of performing modular reductions can be reduced.

Also the ECC scheme has been properly optimized. A dedicated ECC library based on a combination of TinyECC and ContikiECC libraries has been developed, implementing the elliptic curve as a curve, described in the simplified Weierstrass [10] form $E(\mathbb{F}_p) : y^2 = x^3 + ax + b$. The elliptic curve E is defined over a prime field $\mathbb{F}_p$ where $p = 2^{160} - 2^{31} - 1$ according to SECG recommendations [23]. It provides a security level of 80 bits. In addition, the group size p is a pseudo Mersenne prime, therefore modular multiplication and squaring can be speeded up by adopting curve-specific optimizations. Moreover, the elliptic curve E can be converted from affine coordinates to Jacobian ones as $E(\mathbb{F}_p) : Y^2 = X^3 + aXZ^4 + bZ^6$, where $X = xZ^2$, $Y = yZ^3$. The adding of a third element to represent a point (X, Y, Z), allows to separately calculate the numerator and the denominator during certain computational costly operations such as the modular inversion. As a result, the execution time to perform such operations can be further reduced.

Scalar multiplication is often the most expensive operation in elliptic curve-based cryptography, therefore optimizing it can drastically improve overall performance of the specific cryptographic protocol. Let $G \in \mathbb{G}_q$ be a generator of a cyclic subgroup of order q. The elliptic curve scalar multiplication $P = k \cdot G$ is defined as the addition of the point G along the curve E repeated k times. However, to reduce the number of point additions performed, the considered implementation adopts the *double-and-add* method [10], whose benefit is to trade slower point additions for point doublings. To further reduce the number of point additions performed during a scalar multiplication, the *sliding window* method has been adopted [10]; it defines w as the window size, precomputing points $2G$, $3G$, ..., $(2^w - 1)G$ and processing w digits of k at a time. More specifically, for each w-bit window processed, the sliding window method performs w point doubling and just one point addition, thus reducing computation complexity at the cost of additional RAM, depending on the window size, for storing precomputed points.

3.1.3 ECC operations and de-synchronization

Elliptic curve operations are really time-consuming and their execution significantly influences the management of other competitive tasks active into a single mote. For this reason, some tricks should be introduced to avoid that the node may loose the control of enqueued tasks (software overload), works with inconsistent and incoherent variables stored within registers, and involves in frequent de-synchronizations.

To improve the control of competitive tasks, the *task-list* depth, which is set to 10 items as default, has been increased to 25. This value ensures a good compromise between the number of tasks that can be enqueued and the required RAM footprint. ECC operations involve frequent communications with registers and other hardware components of the mote. To guarantee data consistency and data coherency, hardware interrupts have been disabled during the execution of ECC operations. However, without physical interrupts, the mote may loose the synchronization with its parent (because it does not receive keep-alive packets anymore). To solve this issue, the de-synchronization timeout has been increased from 5s to 10s. Finally, the periodicity through which KMP tasks are executed have been properly sized to take into account the impact of time-expensive ECC operations. In particular, the KMP module is implemented as a Finite State Machine, which triggers the execution of a novel operation every specific time interval. This time interval is set to 4s and 6s when the KMP is involved in the exchange of messages and execution of ECC operations, respectively.

3.1.4 Authentication field and Key Derivation

In the developed implementation, the authentication field exchanged in the second part of the KMP is computed by using the Advanced Encryption Standard (AES) algorithm operating in the CBC-MAC mode. Of course, traditional HMAC techniques may ensure lower computational efforts. However, since AES primitives are already available within constrained devices, with such it is possible to obtain the best compromise between computational time and storage footprint.

To ensure a good compromise among simplicity, security, and code footprint the *MGF1* function, specified by IEEE P1363a has been implemented.

3.1.5 Workload at the coordination side

Due to its limited computational and storage capabilities, a single node cannot execute parallel KMP instances. For this reason, the coordinator has been designed to manage a single key negotiation procedure at a time.In fact, it is not possible to store the set of variables belonging to each KMP instance, due to the lack of ROM space. Therefore, messages belonging to KMP procedures initiated by other nodes are silently discarded (i.e., deleted without generating any error message).

4. PERFORMANCE ASSESSMENT

To demonstrate the effectiveness of the proposed approach, bandwidth requirements and ROM footprint have been firstly evaluated and compared with benchmark solutions proposed in [4] [17][19], that use explicit 864 bytes long X-509 certificates[2]. Results reported in Tab. 1 demonstrate how the proposed approach asks for the lowest bandwidth requirements, while requiring a not excessive ROM footprint. It emerges that the proposed solution is more suitable for challenging IoT systems, thanks to its ability to guarantee the maximal airtime savings.

To fully capture the computational requirements characterizing the conceived KMP protocol, a simple testbed composed by only two TelosB motes has been arranged. The superframe structure of the IEEE 802.15.4 network has been set according to the guidelines proposed in [26] From results shown in Fig. 2, it is possible to observe that the highest computational effort is required for calculating the public key from the implicit certificate and the *Pre Link Key*. The mutual authentication phase, instead, does not need for a significant computational load. As a consequence, when a couple of nodes have to negotiate the key for the first time, more than 18 s are required to complete the whole protocol. When the re-keying is just required, the key can be negotiated in less than 5 s (very fast re-keying).

Finally, in order to evaluate the impact that contemporary KMP sessions handled by a single device has on the overall time needed to establish all the secure links, a more complex testbed where a number of nodes are arranged in a

[2]Note that the results reported for benchmark solutions have been extracted from reference papers.

Table 1: Bandwidth requirements and ROM footprint.

Strategy	Logical messages	MAC packets	ROM Footprint
Lightweight version of DTLS [19]	7	60	60 kByte
Proposal in [4]	6	59	1.6 kByte
Proposal in [17]	4	22	8.1 kByte
Proposed approach	4	4	5.8 kByte

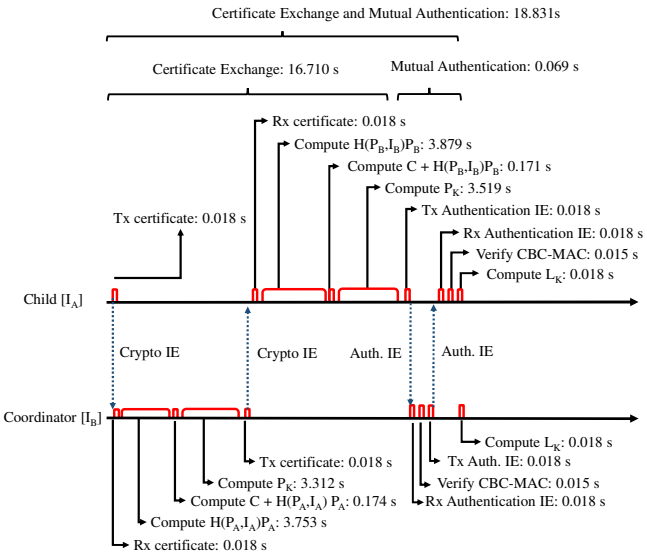

Figure 2: Temporal diagram of the proposed KMP.

star topology has been considered. Results reported in Fig. 3 demonstrate that the higher is the number of nodes that concurrently perform a KMP session with a single device, the higher is the time needed to create a secure domain. When the number of devices increases, in fact, the amount number of messages exchanged into the network and operations handled by the coordinator increases as well.

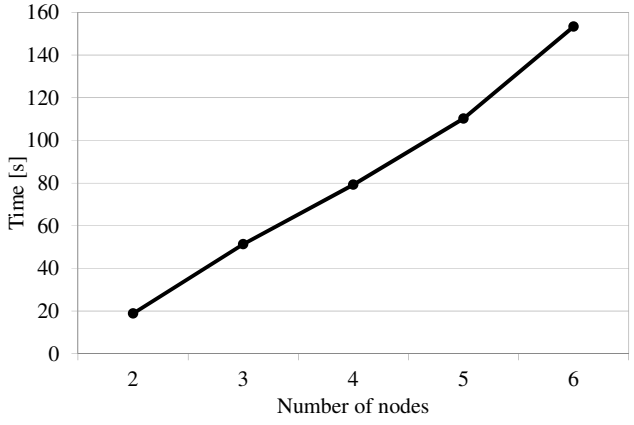

Figure 3: Duration of concurrently KMP schemes in a star topology.

5. CONCLUSIONS AND FUTURE WORKS

In this paper, a novel Key Management Protocol has been proposed for enabling security services in challenging IoT scenarios. It offers, at the same time, *robust* key negotiation, *lightweight* node authentication, *fast* re-keying, and *efficient* protection against relay attacks. Details about its open source implementation within the widely used OpenWSN protocol stack have been also presented. Finally, a preliminary performance assessment has been carried out for demonstrating its effectiveness in simple, but significant, scenarios. As a future work, an experimentally test of the proposed solution in more complex IoT systems, as well as the comparison with most of valuable benchmark approaches are planned.

6. ACKNOWLEDGMENT

This work was supported by the PON projects (RES NOVAE, DSS-01-02499, EURO6-01-02238), the PRIN project TENACE, and the National Technological Cluster CTN001_0034_ 23154 Social Museum and Smart Tourism, funded by the Italian MIUR and by the European Union (European Social Fund). It was also supported by the European Commission in the context of the H2020/ReCRED-653417 project.

7. REFERENCES

[1] ZigBee IP Specification Overview.

[2] Accettura, N., and Piro, G. Optimal and secure protocols in the IETF 6TiSCH communication stack. In *Proc. of Int. Symp. on Industrial Electronics (ISIE)* (Jun. 2014), pp. 1469–1474.

[3] Basu, S., and Pushpalatha, M. Analysis of energy efficient ECC and TinySec based security schemes in Wireless Sensor Networks. In *Proc. of IEEE Int. Conf. on Advanced Networks and Telecommun. Systems (ANTS)* (Dec. 2013).

[4] Bianchi, G., Capossele, A. T., Mei, A., and Petrioli, C. Flexible Key Exchange Negotiation for Wireless Sensor Networks. In *Proc. of the ACM Int. Workshop on Wirel. Netw. Testbeds, Experim. Evaluation and Characterization* (2010).

[5] Brown, D. R. L., Gallant, R. P., and Vanstone, S. A. Provably Secure Implicit Certificate Schemes. In *Proc. of the Int. Conf. on Financial Cryptography* (2002), Springer-Verlag.

[6] Certicom. Explaining Implicit Certificates. Tech. rep., Certicom, 2004.

[7] Eldefrawy, M., Khan, M., and Alghathbar, K. A key agreement algorithm with rekeying for wireless sensor networks using public key cryptography. In *Proc. of Int. Conf. on Anti-Counterfeiting Security and Identification in Commun. (ASID)* (Jul. 2010).

[8] Eschenauer, L., and Gligor, V. D. A Key-management Scheme for Distributed Sensor Networks. In *Proc. of the ACM Conf. on Computer and Commun. Security* (2002), ACM.

[9] Granjal, J., Monteiro, E., and Silva, J. Security for the Internet of Things: A Survey of Existing

Protocols and Open Research issues. *IEEE Commun. Surveys Tuts. PP*, 99 (2015), 1–1.

[10] Hankerson, D., Vanstone, S., and Menezes, A. *Guide to Elliptic Curve Cryptography*. Springer, 2004.

[11] Housley, R., Polk, W., Ford, W., and Solo, D. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 3280, Apr. 2002.

[12] IEEE. Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs), Jun. 2011.

[13] Jiang, R., Luo, J., and Wang, X. A logic-route key tree based group key management scheme for wireless sensor networks. In *Proc. of IEEE Int. Conf. on Commun. in China (ICCC)* (Aug. 2013).

[14] Menezes, A. J., Vanstone, S. A., and Oorschot, P. C. V. *Handbook of Applied Cryptography*. CRC Press, Inc., 1996.

[15] Mirzadeh, S., Cruickshank, H., and Tafazolli, R. Secure Device Pairing: A Survey. *IEEE Commun. Surveys Tuts. 16*, 1 (First 2014).

[16] Piro, G., Boggia, G., and Grieco, L. A. A Standard Compliant Security Framework for IEEE 802.15.4 Networks. In *Proc. of IEEE World Forum on Internet of Things (WF-IoT)* (Seoul, South Korea, Mar. 2014).

[17] Piro, G. and Boggia, G. and Grieco, L.A. *Layer-2 security aspects for the IEEE 802.15.4e MAC*. IETF 6TiSCH WG, Dec. 2014.

[18] Ranjan, A. K., Kumar, V., and Hussain, M. Security analysis of TLS authentication. In *Proc. of Int. Conf. on Contemporary Computing and Informatics (IC3I)* (Nov. 2014).

[19] Raza, S., Shafagh, H., Hewage, K., Hummen, R., and Voigt, T. Lithe: Lightweight Secure CoAP for the Internet of Things. *Sensors Journal, IEEE 13*, 10 (2013).

[20] Richardson, M. 6tisch secure join using 6top . Internet draft, IETF, Nov. 2014.

[21] Sciancalepore, S., Piro, G., Boggia, G., and Grieco, L. A. Application of IEEE 802.15.4 security procedures in OpenWSN protocol stack. *IEEE Standards Education e-Magazine 4* (4th Quarter 2014).

[22] Sciancalepore, S., Piro, G., Vogli, E., Boggia, G., and Grieco, L. On securing IEEE 802.15.4 networks through a standard compliant framework. In *Proc. of IEEE EuroMed Telco Conf. (EMTC)* (Naples, IT, November 2014).

[23] SECG. Sec 2: Recommended elliptic curve domain parameters version 2.0.

[24] Sicari, S., Rizzardi, A., Grieco, L. A., and Coen-Porisini, A. Security, privacy and trust in internet of things: The road ahead. *Computer Networks 76* (2015).

[25] Struik, R. 6TiSCH Security Architectural Elements, Desired Protocol Properties, and Framework . Internet draft, IETF, Oct. 2014.

[26] Villajosana, X., and Pister, K. Minimal 6TiSCH Configuration,. Internet draft, IETF, Jan. 2015.

[27] Watteyne, T., Vilajosana, X., Kerkez, B., Chraim, F., Weekly, K., Wank, Q., Glaser, S., and Pister, K. OpenWSN: a standards-based low-power wireless development environment. *IEEE Trans. on Emerg. Telecommun. Technol. 23*, 5 (2012).