# Public Key Authentication and Key agreement in IoT devices with minimal airtime consumption

Savio Sciancalepore, *Student Member, IEEE,* Giuseppe Piro, *Member, IEEE,*
Gennaro Boggia, *Senior Member, IEEE*, and Giuseppe Bianchi

***Abstract*—Computational complexity of Public Key Cryptography over sensor nodes is not anymore a blocking concern in modern devices which natively (and efficiently) support Elliptic Curve Cryptography. The problem has rather shifted towards the significant airtime consumption required to exchange multiple messages and certificates so as to perform authentication and key agreement. This letter addresses such problem by exploiting implicit certificates (Elliptic Curve Qu-Vanstone). We specifically propose a novel Key Management Protocol which suitably integrates implicit certificates with a standard Elliptic Curve Diffie-Hellman exchange, and performs authentication and key derivation. As confirmed by a proof-of-concept implementation and relevant experimental results, the proposed Key Management Protocol guarantees maximal airtime savings (up to 86,7%) with respect to conventional approaches, robust key negotiation, fast re-keying, and efficient protection against replay attacks.**

***Keywords*—Industrial IoT, security, key management, ECC, ECDH, ECQV, X.509**

## I. Introduction

Security is vital in Industrial Internet of Things (IIoT). Critical applications such as process monitoring and automation require device-level authentication and support for encrypted and authenticated data exchange over the underlying low-power wireless sensor networks [5].

Standardization in this field has significantly advanced in the last years. At the application layer, the Constrained Application Protocol (CoAP) has modified the widely employed Datagram Transport Layer Security (DTLS) so as to fit it into memory/energy constrained devices [14]. At the data link layer, cryptographic extensions have been included in the IEEE 802.15.4 standard, and a new standardization initiative, namely the IEEE 802.15.9 standard [3], is currently addressing the support for Key Management Protocol (KMP) messages through IEEE 802.15.4 Information Elements. In the mean time, Public Key Cryptography (PKC) - a fundamental building block in most KMPs - has become viable also on constrained devices. Indeed, we are now at a point in time where Elliptic Curve Cryptography (ECC) is not only affordable on today's ever more powerful and (memory) capable constrained devices [7], [16], but it is even cheap and natively introduced in the latest generation of IIoT devices [14].

S. Sciancalepore, G. Piro and G.Boggia are with the Department of Electrical and Information Engineering (DEI), Politecnico di Bari, Bari, Italy; e-mail: {name.surname}@poliba.it.

G. Bianchi is with the Department of Electronic Engineering, University of Rome Tor Vergata, Rome, Italy; e-mail: giuseppe.bianchi@uniroma2.it

Manuscript received xxx xx, 2016; revised xx, 2016.

Such technological evolution is bringing us to a point where the primary concern is not anymore the computationally efficient support of ECC, but rather stems in *how to use* such primitives for building *airtime-efficient* authentication and key management protocols. Indeed, most of the proposed PKC-based handshakes [15] suffer from a significant shortcoming in terms of *number and size of the messages exchanged* [18]. In particular, transmission of long messages containing conventional X.509 certificates [12] yields a sizeable *airtime consumption*, whose major consequences are i) a significant latency in the authentication protocol when run over a typical low-rate communication channel, and ii) a significant power consumption, being airtime a major power drain component.
**Contribution**. Based on these premises, the contribution of this letter is threefold. First, our proposed approach is among the first to concretely integrate and experimentally evaluate "implicit" Elliptic Curve Qu-Vanstone (ECQV) certificates [1] within an authentication and key agreement protocol devised for IIoT devices and scenarios. While in our former work [18] performance were affected by a software implementation of the ECC primitives, this work shows that the viability of such technique is greatly improved by the native (e.g., hardware) and efficient support of ECC over modern IIoT devices. Second, our novel proposed KMP relies on an ordinary and widely established "fixed" Elliptic Curve Diffie-Hellman (ECDH) exchange [6], which provides authentication without any explicit signature, as well as ephemeral key derivation (and very fast re-keying, when necessary). This is obtained by exchanging per-session nonces and by securing the exchange using a minimized number of messages (two per each direction, i.e., four in total). Finally, experimental performance results over both single-hop and multi-hop networks show significant improvements in terms of *maximal airtime savings* (up to 86,7%) with respect to a traditional approach relying on an ECDH exchange with public coefficients certified/signed using the ECC Digital Signature Algorithm (ECDSA).

## II. Background: ECQV implicit certificates

For the reader's convenience, we briefly review the notion of *implicit certificates* along with the details of the ECQV algorithm [1]. Let $\mathcal{G}$ be an Elliptic Curve Group, and let $G \in \mathcal{G}$ be a generator of (prime) order $n$ of the group $\mathcal{G}$. Let $U$ be the bit-string identifying a given user, and let CA be a Certification Authority with Private Key $p_{ca} \in \{0, n\}$, and Public Key $P_{CA} = p_{ca} \cdot G \in \mathcal{G}$. An *implicit* certificate for the user $U$ is a single point $C_U \in \mathcal{G}$ of the group, issued by the CA, which permits a receiver who *knows* the user identity $U$

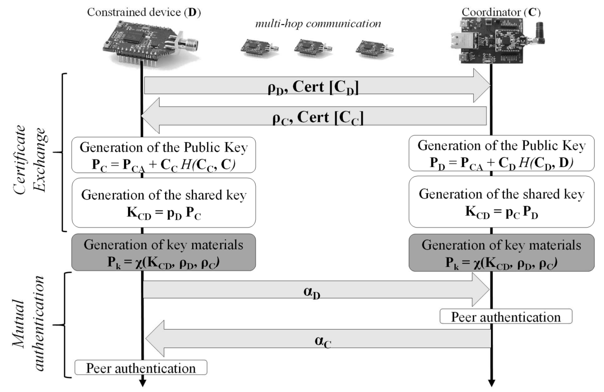
Constrained device (D)
multi-hop communication
Coordinator (C)
ρD, Cert [CD]
ρC, Cert [CC]
Certificate Exchange
Generation of the Public Key PC = PCA + CC H(CC, C)
Generation of the Public Key PD = PCA + CD H(CD, D)
Generation of the shared key KCD = pD PC
Generation of the shared key KCD = pC PD
Generation of key materials Pk = χ(KCD, ρD, ρC)
Generation of key materials Pk = χ(KCD, ρD, ρC)
Mutual authentication
αD
Peer authentication
αC
Peer authentication

Fig. 1. Key negotiation protocol.

*and* the public key $P_{CA}$ of the CA, to *extract* the user's Public Key $P_U \in \mathcal{G}$ such that $P_U = p_u \cdot G$, with $p_u \in \{0, n\}$ being the user's private key. Specifically, let $H(\cdot)$ be a cryptographic hash function; then an implicit certificate is issued as follows:
**1.** the user $U$ generates a random positive integer $r$, computes an elliptic curve point $R = r \cdot G$, and sends it to the CA;
**2.** the CA generates a random positive integer $k$, and returns the implicit certificate as the elliptic point $C_U = R + k \cdot G$, along with the *implicit signature* $\gamma_u = p_{ca} + k \cdot H(C_U, U)$;
**3.** the user generates the private key $p_u = \gamma_u + r \cdot H(C_U, U)$ and the relevant public key $P_U = p_u \cdot G$.

Any party can trivially compute $U$'s public key by knowing only its identity $U$, its implicit certificate $C_U$, and the CA's public key $P_{CA}$, as $P_U = P_{CA} + H(C_U, U) \cdot C_U$. Indeed,

$$P_{CA} + H(C_U, U)C_U = \{p_{ca} + (r+k)H(C_U, U)\}G = p_uG = P_U.$$

The security of such construction was formally proven in [10], and an X.509-compliant implicit certificate format using in total 78 bytes was presented in [11].

## III. Key Management Protocol

Our proposed Key Management Protocol (KMP) relies on the widely accepted ECDH handshake, but minimizes airtime consumption by exchanging the public ECDH coefficients in *implicit format*, i.e. by using ECQV implicit certificates. Our design challenge is therefore to comply with the "fixed" nature of the resulting handshake (ephemeral ECDH coefficients would mandate an explicit digital signature). This is addressed by the exchange and handling of fresh nonces per each session, via two supplementary *authentication* messages following the initial ECDH exchange, which guarantee peer authentication and (fresh) key derivation.

In details, the proposed protocol, non restrictively illustrated in Fig. 1 for the case of a constrained device $D$ which mutually authenticates and negotiates a session key with a coordinator device $C$, comprises two phases. Without loss of generality, we assume that all the constrained devices, including the network coordinator, trust the same Certification Authority (CA), and possess the following cryptographic material: own public key, own private key, own implicit certificate, and CA's public key.

In the first phase, each peer $X$ (i.e., $C$ or $D$) sends to the other party a message comprising i) a nonce $\rho_X$, and ii) an implicit ECQV certificate $C_X$. The receiver is thus able to:
**1.** extract the Public Key $P_X = P_{CA} + C_X \cdot H(C_X, X)$ from the implicit certificate $C_X$, as described in section II, and
**2.** compute the (same and fixed) pre-master session key $K_{CD}$ using an ordinary ECDH. For instance, once received $C_D$ and after having extracted $P_D$, the network coordinator $C$ can use its private key $p_c$ to compute $K_{CD} = p_cP_D = p_cp_dG$; the same result is obtained on the other side by the device $D$ which computes $K_{CD} = p_dP_C = p_cp_dG$.

Note that (as indeed expected by a "fixed" ECDH exchange) subsequent handshakes among the same pair of devices would yield the same key $K_{CD}$. Moreover, note that any attacker could *replay* the message containing the implicit certificate so as to impersonate one of the two devices.

To address both such issues, our proposed Key Management Protocol relies on fresh nonces in each "new" exchange. Nonces serve for two complementary purposes:
**1. Authentication.** To authenticate the exchange, peers $D$ and $C$ compute an *authentication tag*

$$\alpha_D = \Gamma[K_{CD}, (C_D, D, C_C, C, \rho_D, \rho_C)]$$
$$\alpha_C = \Gamma[K_{CD}, (C_C, C, C_D, D, \rho_C, \rho_D)]$$

where the $\Gamma[k, s]$ operator refers to a generic symmetric authentication algorithm (e.g., an HMAC) working on the bit stream $s$ by using key $k$. Note that the two tags differ, as they use the same information but in a different order [6].
**2. Key Derivation.** Finally, each peer computes the actual session key $P_K$ (or multiple keys, to distinguish encryption key from integrity key as customarily done in mainstream security protocols) via an ordinary/standard Key Derivation Function (KDF) $\chi$, as: $P_K = \chi(K_{CD}, \rho_D, \rho_C)$ - details in section IV.

We conclude this section by noting that the proposed KMP intrinsically supports fast opportunistic re-keying. In fact, based on the fixed ECDH mechanism, the key negotiation algorithm always produces the same shared secret for a specific device pairs. Therefore, the heavy cryptographic operations required by both ECQV and ECDH algorithms can be avoided in a subsequent handshake: a device can retrieve from a local cache the previously computed pre-master key and simply compute a new session key using the new exchanged nonces.

### A. Security considerations

The proposed approach relies on building blocks whose security is widely established, namely ECDH [6] and ECQV [10]. Note that these building blocks remain *independent* in our construction, opposed to being composed, and secret parameters for both schemes are neither exposed to external entities nor used in a way different from their standard usage. In this way, we avoid possible issues that may emerge when mixing blocks whose conjuncted adoption is not universally guaranteed ([9], for instance, shows problems related to the composition of implicit certificates and ECDSA technique).

Moreover, ECQV implicit certificates, binding a public key to its owner in a trusted way, make the proposed strategy robust against Man-In-The-Middle (MITM) attacks. Furthermore, the mutual authentication scheme implemented in the second part of the protocol protects the entire approach against replay attacks, and explicitly binds the exchanged cryptographic quantities to the involved peer identities using per-session nonces. It is worth noting that the two authentication messages closely mimic the operation of the *Finished* message in the Transport Layer Security (TLS) protocol, and therefore inherit the relevant security properties assessed for the TLS protocol [17]. Indeed, each authentication tag is computed by including *all* the information exchanged in the first two messages (plus the peer identities) and hence protects the entire exchange from MITM modifications.

Finally, the designed protocol does not specifically influence (i.e. neither positively or negatively) resilience against physical attacks such as tampering, fault, and side-channel attacks, resilience which is mandated to a careful technical implementation and choice of the involved Elliptic curves. For instance, standard software/hardware-based techniques can be used to mitigate tampering attacks and prevent the physical access to security parameters stored within the device. To prevent fault attacks which force the victim device to perform calculations on weak elliptic curves in order to leak the secret key, it is necessary to carefully select the considered ECC curve: the one adopted in our implementation (see Section IV for details) satisfies this requirement. And in terms of side-channel attacks (at least for what concerns side timing-channels), the ECC hardware implementation adopted in our prototype employs the *Montgomery ladder* [6] algorithm (see Section IV) and thus guarantees that the time needed to perform ECC point multiplication is independent from side-channel information and does not leak secret key information.

## IV. Implementation

The proposed KMP scheme has been implemented in the OpenWSN protocol stack (openwsn.atlassian.net), installed on the OpenMote-CC2538 constrained platform (www.openmote.com), and released as open source at the site http://telematics.poliba.it/iot-ecc, under the Berkeley Software Distribution (BSD) license. Some implementation aspects and design choices are listed below.

**1.** The KMP has been developed at the application layer and integrated in CoAP. The KMP instance running on a given node is handled as a CoAP resource, identified with the Uniform Resource Identifier (URI) "*coap://[ip-addr]:[port]/kmp*", where *[ip-addr]*, *[port]*, and *kmp* are the IPv6 address of the node, the port number (default value 5683), and the KMP resource, respectively.

**2.** At the link-level, the IIoT technology supports very small packets (e.g., Maximum Transmission Unit equal to 127 bytes). KMP messages carrying peer certificates thus need to be fragmented. Then, we used the *block-wise* transfer mode [8].

**3.** CoAP provides a reliable transfer by acknowledging packets after reception. To reduce bandwidth and energy consumption, and reduce delay (to a greater extent in multi-hop scenarios) we employed a *piggybacking* mechanism, where fragments of each peer's certificate are transmitted together with the ACK message referring to the previously received packet.

**4.** The OpenMote-CC2538 cryptoprocessor provides hardware accelerated atomic cryptographic ECC operations, which we have properly assembled and further optimized via software configuration. The well-known curve *secp160r1* provided by NIST [4] is used for handling ECC operations. To the best of our knowledge, this is the first contribution that integrates elliptic cryptography on OpenMote-CC2538 boards in the OpenWSN protocol stack. From the security perspective, *secp160r1* is able to guarantee the minimum acceptable security level (e.g., equal to 80) envisaged for an elliptic curve [13], while ensuring the lowest bandwidth requirements. Moreover, such curve is considered resilient against fault attacks [19]. Furthermore, the OpenMote-CC2538 cryptoprocessor also implements the *Montgomery ladder* algorithm, used to make ECC operations resilient against side-channel attacks. The authentication tag is also generated by a hardware accelerated execution of the HMAC-SHA256 algorithm.

**5.** The session key $P_K$ is currently computed by using the *MGF1* function, specified by IEEE P1363a [2], which provides a good compromise among simplicity, security, and code footprint. The code can be trivially extended to support better key derivation functions, such as the provably-secure KDFs recommended in the IETF RFC 5869.

**6.** For simplicity, we do not use ECC point compression techniques to further reduce the size of the implicit certificate. Due to fragmentation of the application payload, the first two logical messages are always sent through 2 link-layer packets.

**7.** The implemented KMP based on implicit certificates requires 12.240 kBytes of ROM and and 644 Bytes of RAM, whereas the KMP implementation based on X.509 explicit certificates used in the next section for benchmarking purposes uses 15.192 kBytes of ROM and 877 Bytes of RAM. It therefore follows that implicit certificates also bring about advantages in terms of memory footprint.

## V. Experimental performance assessment

The airtime consumption requested by the proposed KMP has been experimentally evaluated in a network with a variable number of devices organized in a chain topology. A benchmark approach that uses the same KMP procedure and adopts explicit X.509 certificates encoded through the standard Privacy Enhanced Mail (PEM) format and the binary Distinguished Encoding Rules (DER) format, as well as digitally signed via the ECDSA algorithm, has been taken into account for the comparison. The link layer, based on the IEEE 802.15.4e technology, has been configured with a slotframe of 101 slots, each lasting 10 ms. During a slotframe, each node has only two active slots: one for the management of the control traffic (i.e., routing messages, beacon frames, and keep-alive packets), the other one for exchanging KMP-related packets. As a result, a 2% duty-cycle for each node is ensured.

Fig. 2 shows the impact that cryptographic and communication functionalities have on each single atomic operation. These results have been obtained by testing the execution of the KMP in a single hop network. As expected, implicit certificates can significantly reduce the handshake time. This performance gain is mainly obtained thanks to two different aspects. First, the 78 bytes implicit certificates can be sent within just 2 link-layer packets, opposed to the 13 packets needed for the 725

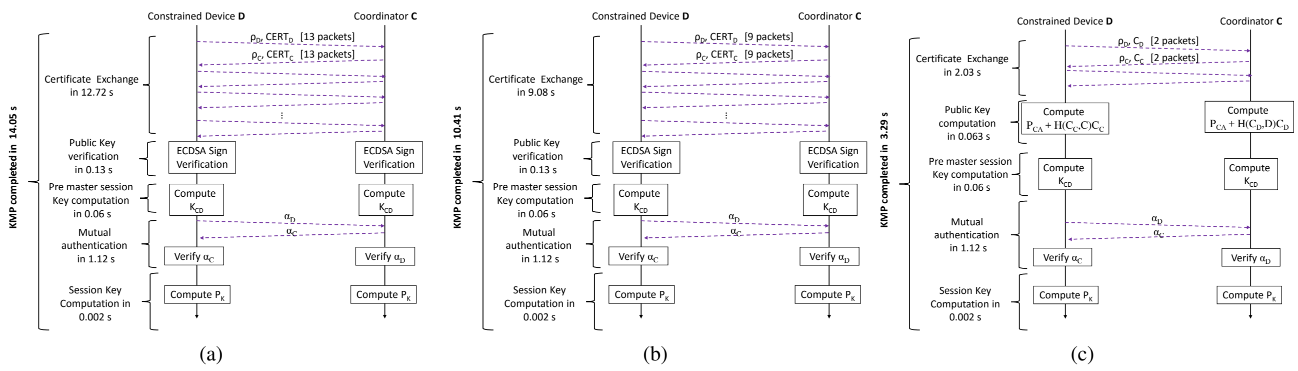
Constrained Device D
Coordinator C
KMP completed in 14.05 s
Certificate Exchange in 12.72 s
$p_D$, CERT$_D$ [13 packets]
$p_C$, CERT$_C$ [13 packets]
Public Key verification in 0.13 s
ECDSA Sign Verification
Pre master session Key computation in 0.06 s
Compute $K_{CD}$
Mutual authentication in 1.12 s
$\alpha_D$
$\alpha_C$
Verify $\alpha_C$
Verify $\alpha_D$
Session Key Computation in 0.002 s
Compute $P_K$
(a)
KMP completed in 10.41 s
Certificate Exchange in 9.08 s
$p_D$, CERT$_D$ [9 packets]
$p_C$, CERT$_C$ [9 packets]
(b)
KMP completed in 3.29 s
Certificate Exchange in 2.03 s
$p_D$, $C_D$ [2 packets]
$p_C$, $C_C$ [2 packets]
Public Key computation in 0.063 s
Compute $P_{CA} + H(C_C,C)C_C$
Compute $P_{CA} + H(C_D,D)C_D$
(c)

Fig. 2. Comput. load of atomic operations: (a) explicit X.509 certificates, PEM format; (b) explicit X.509 certificates, DER format; (c) implicit X.509 certificates.

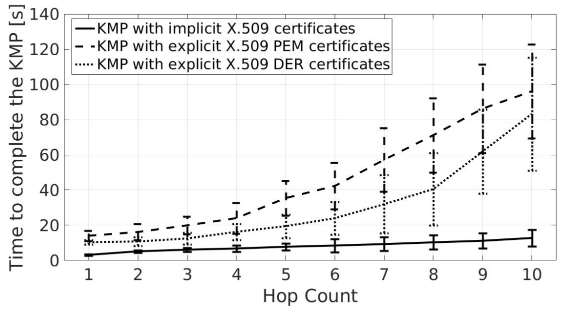

Fig. 3. Airtime consumption.

bytes of the explicit X.509 certificate in the PEM format, or the 9 packets required to send the 495 bytes of the explicit X.509 certificate in the DER format. Second, cryptographic operations of ECQV require a lower computational load with respect to that needed to compute an ECDSA signature.

Fig. 3 reports (with 95% confidence intervals) the time required to complete the KMP protocol as a function of the number of hops that separates the involved nodes. These experimental results clearly demonstrate that the usage of implicit certificates always ensures the maximal airtime saving, with performance gains over explicit X.509 PEM certificates ranging from 77,1% to 86,7%, and from 50,9% to 84,7% with respect to explicit X.509 certificate in the DER format.

## VI. Conclusions

In this letter we have proposed and experimentally evaluated a Key Management Protocol specifically designed for IIoT scenarios, where bandwidth and energy consumption are critical aspects. While relying on a standard and widely accepted ECDH scheme, it significantly improves airtime savings by employing implicit ECQV certificates. The proposed scheme also provides peers' authentication, ephemeral key derivation, fast re-keying, and efficient protection against replay attacks. Experimental assessment on a concrete CoAP-based implementation, supporting a join protocol for devices potentially many hops far from each other, shows airtime savings up to 86,7% with respect to conventional schemes relying on explicit X.509 certificates.

## Acknowledgements

This work was framed in the context of the project SymbIoTe, which receives funding from the European Union's Horizon 2020 research and innovation programme under grant agreement 688156.

## References

[1] Explaining Implicit Certificates. Technical report, Certicom, 2004.

[2] IEEE Standard Specifications for Public-Key Cryptography - Amendment 1: Additional Techniques. IEEE P1363a, 2004.

[3] IEEE Recommended Practice for Transport of Key Management Protocol (KMP) Datagrams. *IEEE Std 802.15.9-2016*, Aug 2016.

[4] E. Barker, W. Barker, W. Burr, W. Polk, and M. Smid. Recommendation for Key Management - Part 1: General. NIST Spec. Pub. 800-57, 2012.

[5] A. Barki, A. Bouabdallah, S. Gharout, and J. Traore. M2M Security: Challenges and Solutions. *IEEE Commun. Surveys Tuts.*, (99), 2016.

[6] I. Blake, G. Seroussi, and N. Smart. *Elliptic Curves in Cryptography*. Cambridge University Press, 1999.

[7] C. Bormann, M. Ersue, and A. Keranen. Terminology for Constrained-Node Networks. RFC 7228, May 2014.

[8] C. Bormann and Z. Shelby. Block-wise transfers in CoAP - draft-ietf-core-block-18. Internet draft, IETF, Sep. 2015.

[9] D. R. Brown, M. J. Campagna, and S. A. Vanstone. Security of ECQV-Certified ECDSA Against Passive Adversaries. *IACR Cryptology ePrint Archive*, 2009.

[10] D. R. Brown, R. Gallant, and S. A. Vanstone. Provably secure implicit certificate schemes. In *Financial Cryptography*, pages 156–165. Springer, 2001.

[11] M. Campagna. SEC 4: Elliptic Curve Qu-Vanstone Implicit Certificate Scheme (ECQV). Technical report, Certicom Research, Jan. 2013.

[12] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280, May 2008.

[13] ECRYPT. *Yearly Report on Algorithms and Keysizes*, 2015.

[14] S. L. Keoh, S. Kumar, and H. Tschofenig. Securing the Internet of Things: A Standardization Perspective. *IEEE IoT J.*, 1(3), 2014.

[15] S. Mirzadeh, H. Cruickshank, and R. Tafazolli. Secure device pairing: A survey. *IEEE Commun. Surveys Tuts.*, 16(1):17–40, 2014.

[16] O. P. Pinol, S. Raza, J. Eriksson, and T. Voigt. BSD-based elliptic curve cryptography for the open Internet of Things. In *Int. Conf. on New Technol., Mobil. and Secur. (NTMS)*, pages 1–5, Jul. 2015.

[17] A. K. Ranjan, V. Kumar, and M. Hussain. Security analysis of TLS authentication. In *Proc. of Int. Conf. on Contemporary Computing and Informatics (IC3I)*, pages 1356–1360, Nov. 2014.

[18] S. Sciancalepore, A. Capossele, G. Piro, G. Boggia, and G. Bianchi. Key Management Protocol with Implicit Certificates for IoT systems. In *ACM IoT-Sys Workshop*, May 2015.

[19] SECG. *SEC 2: Recommended Elliptic Curve Domain Parameters version 2.0*, 2004.