Low-Energy Symmetric Key Distribution in Wireless Sensor Networks

Kealan McCusker, Noel E O'Connor, *Member, IEEE*

Abstract—In this work a scheme for key distribution and network access in a Wireless Sensor Network (WSN) that utilises Identity-Based Cryptography (IBC) is presented. The scheme is analysed on the ARM920T processor and measurements were taken for the run time and energy of its components. It was found that the Tate pairing component of the scheme consumes significant amounts of energy, and so should be ported to hardware. An accelerator was implemented in 65nm Complementary Metal Oxide Silicon (CMOS) technology and area, timing and energy figures have been obtained for the design. Results indicate that a hardware implementation of IBC would meet the strict energy constraint required of a wireless sensor network node.

Index Terms—Wireless sensor networks, identity based cryptography, hardware architecture

1 Introduction

Recent advances in radio and digital electronics have enabled system on chip technologies to be developed that will incorporate sensing, computation and communication. These devices are known as wireless sensor nodes and are the subject of very active research at present. It is envisaged that eventually they will cost considerably less than one dollar, and hence could be leveraged to provide a distributed WSN containing many thousands of nodes [1]. The characteristics that are attributed to wireless micro-sensors are that they have limited memory, are very inexpensive, and have multi-year life spans from a single power source. The total energy in the power source for the wireless sensor node would be of the order of 1000 joules [2]. Thus, it is imperative that the system architecture of the nodes and the network as a whole should be designed with an aim to minimizing energy dissipation in all aspects of operation.

Even though the computational and sensing ability of an individual node may be quite limited, the aggregated effect of a large number of sensors working together would be to provide a more accurate global picture of the spatial region in which the sensors are placed than could be achieved through conventional sensing technology. This opens up a whole vista of scenarios where sensor networks could be deployed that up until now could not be considered. Furthermore, it is clear that in order for these networks to be deployed in real applications that the issue of security is solved [3], [4].

One such application, where a WSN could play an important role, is in environmental pollution monitoring. Chemical sensors attached to devices with integrated Radio Frequency (RF) transceivers could provide information on the toxic gas present and also the position of the contaminant. Given the sensitive nature of such sensing and potential repercussions associated with it, security is extremely important in this application. This is the target scenario for this paper. The assumptions that are made are that the network is static, the batteries of the devices cannot be replaced and that the nodes are not protected by tamper proof hardware. It is also assumed that the devices that make up the network are homogeneous and have the ability to determine their position by running a localisation algorithm [5].

We believe that a symmetric key cryptosystem is appropriate for communication between the nodes, though a pre-installed system wide symmetric key or pair-wise keys stored on the devices are not suitable for reasons of security and lack of memory, respectively. Therefore an asymmetric or public key system is required to establish the symmetric keys between individual nodes.

With a traditional approach, if node A wants to communicate with node B, it first has to receive B's digital certificate before it can send a message. When the two nodes are in direct radio communication this will mean one transmission from B to A. In the case of B being out of range of A, the digital certificate would have to be relayed to A via intermediate nodes. As the radio is likely to be the main consumer of energy in the node, it is important to minimize the number of transmissions. This can be achieved using IBC [6] in which there is no need for a certificate to bind a node's identity to its public key, as the node's identity can be used as the public key.

There is a lot of related work that use IBC in the context of WSNs [7], [8], [9], [10], [11], [12]. Doyle *et al.* proposed the use of IBC for security in WSN [7]. They profiled the energy required to run the Tate pairing on a 32-bit processor for a curve over $GF(2^{107})$ and arrived at a figure of 0.44J. This work was carried out using simulation on a curve that would not be considered secure due to the small field size.

Cheng *et al.* [8] present an IBC scheme based on the work of Boneh-Franklin [13]. They do not propose the use of a symmetric key crytosystem for WSN and hence are using their IBC scheme for encrypting data. The Tate pairing will only be required to be calculated once for a pair of nodes and can be cached for future use in encryption and decryption. They

• *The authors are with CLARITY: Centre for Sensor Web Technologies, Dublin City University, Ireland.*
E-mail: kealan.mccusker@eeng.dcu.ie

Manuscript received April 19, 2005; revised January 11, 2007.

do not investigate the energy usage of their scheme.

Oliveira *et al.* [9] present an implementation of the Tate pairing on a 8-bit ATmega128L microcontroller used in the Mica devices [14]. The time taken for the Tate pairing calculation is $5.5s$. They detail a scheme for key establishment based on Identity-Based Non-Interactive Key Distribution Scheme (ID-NIKDS).

Szczechowiak *et al.* [10] present a ID-NIKDS scheme and also profile the Tate pairing on a range of different wireless sensor nodes including the Imote2 [15]. Their fastest implementation of the Tate pairing is $0.06s$ and it consumes $3.76mJ$ of energy. They present a scheme that defends against node capture if most of the nodes only have the capability to act as a data source.

The scheme proposed by Kim *et al.* [11] is based on devices being present in the network that can act as security managers. These security managers perform the expensive Tate pairing calculation and the stated advantage of this system is that the low-power nodes do not have to perform this task. As the WSN envisaged in this work is made up of homogeneous nodes then every device would have to perform the role of security manager, if key establishment within the network is to be achieved. Therefore this scheme would not be an improvement over one based on ID-NIKDS.

Zhang *et al.*'s approach is similar to the one pursued in this work, in that it also uses location in its security mechanisms [12] . The security of their system is based on the fact that a network master secret, that is used to generate location based keys, is kept secret for a minimum time. This is the time that it is believed that an adversary would require to access this key if in control of the node. When the nodes have all calculated their location based keys then this network master secret is securely erased. Additional devices added to the network will require access to this network master secret. This scheme's security depends upon keeping this network master secret secure.

It has been identified by previous work that IBC can provide a mechanism for authenticated key agreement in a WSN. This work follows this approach and also proposes a technique that could be used to improve the resistance of the WSN to node capture by maintaining a list of authenticated devices in radio range. A method for adding nodes to the network and removing them from the WSN is outlined. A low energy Tate pairing accelerator is implemented and this, to the best of our knowledge, is the first attempt at designing an accelerator for minimising the energy of the pairing.

The remainder of the paper is organised as follows: the Tate pairing, a key component of IBC, and methods for calculating it, are discussed in Section 2. In Section 3 the suitability of different methods of key distribution for application in WSNs are discussed. IBC is proposed as the most suitable candidate for secure distribution of keys in the network. A scheme for implementing key distribution and network access in a WSN is described in Section 4. How the scheme performs against well know attack is presented in Section 5. A software implementation of the scheme is profiled in Section 6. In Section 7 the Galois field arithmetic units that make up the Tate pairing are discussed and implemented in hardware. Timing and energy figures for the various units are also presented. The overall architecture for the Tate pairing accelerator is presented in Section 8. The timing and energy figures are compared against previous implementations and software. Finally, a discussion of the results and conclusions drawn are presented in Sections 9 and 10.

2 Mathematical Background

This work is concerned with applying IBC to solving the key distribution problem in WSNs. In order to aid understanding of later sections a brief mathematical summary of the area is presented in this section. Further introductory material on Elliptic Curve Cryptography can be found in [16].

2.1 Tate Pairing

The identity based cryptosystems discussed later in this thesis are based on the hardness of the Bilinear Diffie Hellman Problem (BDHP).

Definition 2.1 (Bilinear Diffie Hellman Problem): Given $P, aP, bP, cP \in E(GF(2^m))$ it is computationally infeasible to calculate $e_l(P,P)^{abc} \in GF(2^{mk})$, where $E(GF(2^m))$ is a supersingular elliptic curve defined on the Galois field $GF(2^m)$ and $e_l(P,P)$ is an application of the Tate pairing.

If $l|\#E(GF(2^m))$ where l is a large prime l, and k is the smallest integer such that $l \mid (2^{mk}-1)$ then the Tate pairing is defined as [17], [18];

Definition 2.2 (The Tate Pairing): The Tate pairing, e_l, is the mapping

$$\begin{aligned} e_l &= E(GF(2^{mk}))[l] \times E(GF(2^{mk}))/lE(GF(2^{mk})) \\ &\rightarrow (GF(2^{mk}))^*/(GF(2^{mk}))^{*l} \\ &= \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T \end{aligned}$$

where the l torsion points, $\mathbb{G}_1$, are

$$E(GF(2^{mk}))[l] = \{P \in E(GF(2^{mk})) \mid lP = \mathcal{O}\}. \quad (1)$$

And two points $P, Q \in E(GF(2^{mk}))$ are members of the same equivalence class, $\mathbb{G}_2$, if

$$P \equiv Q \pmod{E(GF(2^{mk}))/lE(GF(2^{mk}))} \quad (2)$$

i.e. $P = Q + lR$ where $R \in E(GF(2^{mk}))$. Similarly $a, b \in (GF(2^{mk}))^*$ are members of the same equivalence class, $\mathbb{G}_T$, such that

$$a \equiv b \pmod{(GF(2^{mk}))^*/(GF(2^{mk}))^{*l}} \quad (3)$$

which can be also be stated as $a = bc^l$ for $c \in (GF(2^{mk}))^*$.

Its most desirable property in the context of cryptography is bilinearity

$$e_l(aP, bQ) \equiv e_l(aP, Q)^b \equiv e_l(P, bQ)^a \equiv e_l(P,Q)^{ab} \quad (4)$$

where a, b are integers. The exponent $\frac{2^{mk}-1}{l}$ of the output of the pairing provides a unique value rather than a member of an equivalence class. The integer k is known as the security multiplier and is four for the particular curve considered in this paper.

The Tate pairing essentially takes two points on an elliptic curve and maps them to a element of a multiplicative group of a large finite extension field. The choice of the elliptic curve group over which the Elliptic Curve Discrete Logarithm Problem (ECDLP) is posed must be such that it requires at least 2^{80} operations to solve. Therefore l has to be at least of the order of $\approx 2^{160}$. Also, the finite field to which the Tate pairing maps must be sufficiently large to make the Discrete Logarithm Problem (DLP) intractable i.e. that it has a running time of 2^{80}. For a binary field, as used in this paper, it has to be of the order of 2^{1024} . As $k = 4$ for the curve used in this paper, this means that m must be at least 250.

2.2 η Algorithm

Based upon the work of Duursma and Lee [19], a closed form of the Tate pairing calculation, which is known as the η algorithm, has been obtained for characteristic two [20], [21]. The Tate pairing is given by

$$f_p(\phi(Q))^{\frac{2^{mk}-1}{l}} = g_p(\phi(Q))^{2^{2m}-1} \tag{5}$$

where

$$g_p = \prod_{i=1}^{2m} l_{2^i P}^{2^{2m-i}} \tag{6}$$

and $l_{2^i P}$ is the equation of the tangent to the curve at the point $2^i P$. Through application of the distortion map, ϕ, and a lot of algebraic manipulation Equation (7) is arrived at and this is rewritten in the form of Algorithm 1.

$$\begin{aligned} g_P(\phi(Q)) = \prod_{i=1}^{m} x_p^{2^i} x_q^{2^{(-i+1)}} + y_p^{2^i} + y_q^{2^{(-i+1)}} \\ + s^2(x_p^{2^i} + x_q^{2^{(-i+1)}}) + t^2 + 1 \end{aligned} \tag{7}$$

where $s, t \in GF(2^{283\times 4})$. This algorithm requires seven

Algorithm 1 η Algorithm [20], [21]

```
$P = (x_p, y_p), Q = (x_q, y_q)$     $: P, Q \in GF(2^{283})$
$C(x) = c_3x^3+c_2x^2+c_1x+c_0 = 1$     $: C(x) \in GF(2^{283\times 4})$
for $i = 1$ to $283$ do
    $x_p = x_p^2$
    $y_p = y_p^2$
    $z = x_p + x_q$
    $w = z + x_p x_q + y_p + y_q + 1$
    $C(x) = C(x)(w + zx + (z + 1)x^2)$
    $x_q = x_q^{2^{283-1}}$
    $y_q = y_q^{2^{283-1}}$
end for
return $C(x) = C(x)^{2^{2\times 283}-1}$
```

multiplications in the field $GF(2^m)$. It has a regular structure that maps well to hardware, and it is the Tate pairing algorithm that is implemented in this work.

3 Security Considerations in a WSN

There is a clear need for security in a WSN. The main requirements, known as confidentiality and network access respectively, are that the data exchanged in the network should not be read by an unauthorised third party and also that this third party cannot join the network. The unique challenges of WSNs is that the nodes have limited energy and radio communication range, there is no device that can act as a trusted server and their topology is not known before deployment. The lack of a trusted server being present in the network means that there are only three approaches to distribute symmetric keys; standard public key schemes, IBC or key predistribution. Standard public keys schemes are not an apporpiate choice due to the extra communication overhead of sending digital certificates as compared to the solution offered by IBC. There are a number of different key predistribution schemes and these are discussed below.

The simplest approach to deploy a symmetric system would be that all the nodes share the same key. As the nodes could be placed in a region where an adversary can capture them, it is likely that it could extract the secret key, and therefore would be able to monitor all communication in the network. For this reason, this method of ensuring privacy is not appropriate in a hostile environment.

Another method would be for all the nodes to set up pair-wise keys between them before deployment. If there are n nodes in the network then each node would have to store $n-1$ keys in its persistent memory. In a resource constrained device this would be a problem as the size of the network would be determined by the memory available. The other main drawback to using this scheme is that it does not scale. If, after deploying the bulk of the nodes, it is required to add extra nodes then this is not possible unless the extra nodes' keys are already programmed in the deployed network. Upon capture of a node, however, only its $n - 1$ links will be compromised, which is a improvement on the system that uses only one symmetric key.

Eschenauer *et al.* developed a key distribution technique based on probabilistic key sharing [22]. In this approach a large pool of keys is generated from which a smaller ring of keys is randomly selected and preloaded before deployment onto each node. Each node thus has a separate ring of keys in which there may be a shared key. During the shared key discovery phase of the algorithm neighbouring nodes ascertain whether they share a key. If there is no path between nodes in radio range there is a further path-key establishment phase which make use of the already secure links to distribute pair-wise key. It has been shown that in order to create a network of $10,000$ nodes the pool of keys has to be $100,000$ and the key ring only has to be 250 [22]. This system is scalable as when a new node is added to network it only has to be preloaded with a random selection of 250 keys from the key pool. However, this scheme is not secure against capture by an adversary. The security of the probabilistic key sharing approach has been improved by Chan *et al.* [23] who proposed that nodes need to have q common keys. Probabilistic key sharing could impose a large transmission overhead upon nodes during the initial set up phase when path-keys are being established, and, due to its probabilistic nature, it might not generate a complete network when the nodes are sparsely dispersed.

Chan *et al.* also propose the random pairwise scheme where they observed that a node does not need to store $n - 1$ keys in order to establish a network [23]. Instead, it must store np keys where n is the size of the network and p is the probability of any two nodes being connected such that a complete network is established. In the initialization phase of this scheme m distinct pairwise keys are placed on the nodes. Upon deployment the nodes broadcast their IDs so that nodes in communication range can ascertain whether they share a common key. This scheme suffers from one of the drawbacks of the naive pairwise scheme as it is not scalable.

Blundo *et al.* present a scheme for distributing conference keys that could be used in WSN [24]. In this scheme a secret symmetric bivarate polynomial, $f(x_1, x_2)$, of degree k with coefficients in $GF(q)$ is selected by the programming entity. Each node will be programmed with a unique identity and this identity, $i \in GF(q)$, is input to the polynomial giving $f(i, x_2)$, which is then stored on the node. If two nodes wish to establish a pair-wise key they insert the identity of the device that they are communicating with into this polynomial share. Each device will need to store a polynomial which occupies $(k+1)log_2q$ bits of memory, thus potentially making the memory a limiting factor on the size of the network. This scheme is only secure as long as less the k nodes are compromised.

The symmetric key distribution scheme of Blom *et al.* could be applied to a WSN [25]. A $k \times n$ generator matrix, G, with elements from $GF(q)$ is selected, where n is the number of nodes in the network. The secret $k \times k$ matrix, D, over $GF(q)$ is generated and is multiplied with G to give $S = (DG)^T$. Each node is assigned the i^{th} row of S and i^{th} column of G. If two nodes now want to establish a shared key they exchange their columns (i, j) in G and perform matrix multiplication with the stored row of S resulting in an element of the matrix $K = (DG)^T G$. A shared key is generated as K is a symmetric matrix and therefore $K_{ij} = K_{ji}$. This scheme is only secure as long as k rows of S remain secret. As with the previous scheme there is a requirement for the node to store a large amount of keying material which in this case is $(k+1)log_2q$ bits.

Lui et al propose a technique that combines the work of Blundo *et al.* with that of Eschenauer et al [26]. Instead of a ring of keys on each node, a number of polynomial shares of different bivariate symmetric polynomials are placed on the devices. The nodes need to know what polynomial shares are on adjacent devices in the network and techniques for achieving this are outlined in the paper. Unlike the basic probabilistic key sharing scheme, each pair of nodes will have a unique key. But it is still a probabilistic technique with the same problems as outlined above. A similar scheme based on the work of Blom *et al.* is presented by Du *et al.* [27].

In comparison with other schemes IBC provides an simple, scalable and secure, against individual node capture, method of distributing symmetric keys. SOK ID-NIKDS was proposed by Sakai, Ohgishi and Kasahara [28] and can be implemented using the Tate pairing. If given $h_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$ and two devices with identity A and B respectively. Then $Q_A = h_1(A)$ and $Q_B = h_1(B)$ where $Q_A, Q_B \in \mathbb{G}_1$. The nodes have their private key sQ_A and sQ_B placed on them by the Key Generation Centre (KGC). The symmetric key, K_{AB}, can be calculated by both parties as;

$$K_{AB} = e_l(sQ_A, Q_B) = e_l(Q_A, Q_B)^s = e_l(Q_A, sQ_B) \quad (8)$$

Thus, the memory requirement of this scheme is better than the other key predistribution schemes as only the identity of the node with which it will communicate is required. Key authentication is also assured as only the KGC and a single node will have a copy of the private key. A major drawback of the using this approach is that an adversary could be able to extract the keying material from a node and generate a pair-wise key with any node in the network. Therefore key distribution has to be combined with network access control to prevent this happening, as outlined in the next section.

4 Scheme

The scheme outlined here, for implementing key distribution and network access control in a WSN, is designed for a static network, and uses SOK ID-NIKDS and BMLQ Identity-Based Signature (IBS) [29]. Environmental pollution monitoring is the target application. In this case the nodes that are detecting the pollution, such as chemical reagents have a fixed position that they determine by running a localisation algorithm. End users have to be able to easily extract data from the network and this can be achieved using a Personal Digital Assistant (PDA) type device. The scheme uses ID-NIKDS and IBS as a method for distributing symmetric keys, and also to allow devices access to the WSN.

We assume that the nodes themselves are not protected by tamper resistant hardware as this would increase their cost. Therefore it is possible that data and keying material on the devices can be extracted. Also, the KGC, which programs the devices, and the PDAs, which extract information form the WSN, are secure. The end users of the WSN would be able to find out if a PDA is lost and hence exclude that particular device from the WSN. Communication between the KGC and the network could be achieved remotely by using the extracting device, such as a PDA, as a proxy. Before the extracting device communicates with the WSN it could be programmed by the KGC with messages that it wishes to broadcast to the network.

There are five distinct stages to this scheme; prior to deployment, deployment, node addition, node removal, and data extraction. Each one of these stages are outlined below.

4.1 Prior to deployment

This part of the scheme is concerned with distributing the domain parameters and private keys to the nodes. The elliptic curve and Galois fields being used are hard coded on the device. For SOK ID-NIKDS the devices have to be able to calculate $K_{AB} = e_l(sQ_A, h_1(B))$. For BLMQ IBS they have to be able to generate a signature, S, and verify a signature, V. Therefore, among the parameters that are placed on the devices are

- $(h_2 : \{0,1\}^* \rightarrow \mathbb{Z}_l) \rightarrow N_X$
- $(h_3 : \{0,1\}^* \times \mu_l \rightarrow \mathbb{Z}_l) \rightarrow N_X$
- $(Q \in \mathbb{G}_2, P = \phi(Q) \in \mathbb{G}_1) \rightarrow N_X$
- $(Q_{PUB} = sQ, g = e_l(P, Q)) \rightarrow N_X$

- $(Q_{KGC}, Q_X, sQ_X) \to N_X$

where a generic node is given the identity N_X and has public key $Q_X = h_1(N_X)$ and private key sQ_X. Q_{KGC} is the public key of the KGC. Instead of placing h_1 on the device, the hash function is carried out by the programming device and the point on the curve to which an identity equates is placed on the node. For the rest of this section Q_* represents the identity of the node and also its public key.

4.2 Deployment

During this phase symmetric keys are set up between neighbouring nodes in a pair-wise fashion. The nodes would transmit a small signed message to every device in radio range at time T_1. This would means that devices that can generate a valid signature are permitted to join the network. They would then generate a pair-wise symmetric key, K_{AB}. The nodes maintain a list of authenticated devices in radio range.

- $Q_A \to Q_B : m||S(m)$
- $Q_B : V(m||S(m))$
- $Q_B : K_{AB} = e_l(Q_A, sQ_B)$.
- Maintain list of nodes in radio range i.e. $\mathbb{C}_B = \{Q_A, Q_C, Q_D\}$

Where K_{AB} is a shared symmetric key between Q_A and Q_B, S represents signing and V represents verification.

4.3 Wireless sensor node addition

At time T_3 extra devices may be added to the WSN. At a previous time, T_2, the KGC will broadcast though the network the identity of the nodes to be added e.g. $\mathbb{E}_{KGC} = \{Q_O, Q_P\}$. The identities of these devices, along with a time-stamp, are signed by the KGC. It does this in order to authenticate these identities and prevent the message, requesting the addition of these identities, being replayed by an adversary in the future.

- $Q_{KGC} \to Q_X : Q_O||Q_P||T_2||(S(Q_O||Q_P||T_2))$
- $Q_X : V(Q_O||Q_P||T_2||(S_{sQ_{KGC}}(Q_O||Q_P||T_2))$
- $Q_P \to Q_X : m||S(m)$
- $Q_X : V(m||S(m))$
- $Q_X : K_{XP} = e_l(sQ_X, Q_P)$.
- Q_X : If $Q_P \notin \mathbb{E}_{KGC}$ then reject Q_P
  else $Q_P \in \mathbb{C}_X$

4.4 Wireless sensor node removal

A node's membership of the WSN can be revoked by the following process. At time T_2 the KGC will broadcast the identity of the nodes to be removed i.e. $\mathbb{E}_{KGC} = \{Q_O, Q_P\}$. The identities of these devices, along with a time-stamp, are signed by the KGC in order to prevent a replay of the message.

- $Q_{KGC} \to Q_X : Q_O||Q_P||T_2||(S(Q_O||Q_P||T_2))$
- $Q_X : V(Q_O||Q_P||T_2||(S_{sQ_{KGC}}(Q_O||Q_P||T_2))$
- Q_X : If $Q_O|Q_P \in \mathbb{C}_X$ remove $Q_O|Q_P$

4.5 Data extraction

In the environmental monitoring scenario it is envisaged that the WSN itself would be static, but that the entities extracting data from the network are mobile. For example, they could be a member of the Environmental Protection Agency who uses a PDA to extract information from the network. The PDA in this case will be programmed with the same domain parameters as the nodes. Only nodes authorised by the KGC can join the network; hence the KGC needs to send a packet that contains the identity of Q_{PDA} and is signed by its private key.

- $Q_{KGC} \to Q_X : Q_{PDA}||S(Q_{PDA})$
- $Q_X : V(Q_{PDA}||S(Q_{PDA}))$
- $Q_X : K_{XPDA} = e_l(sQ_X, Q_{PDA})$.

When a PDA requests a reading, from a certain geographical area, it will diffuse this request through the network. As its identity Q_{PDA} has already been broadcast to the network as a valid identity, then the node Q_X sends data back to the PDA using Advanced Encryption Standard (AES). This message is encrypted by the pair-wise symmetric key (K_{XPDA}) and forwarded towards the extraction point, which is also known as a sink. The encrypted message is also appended with a Keyed-Hash Message Authentication Code (HMAC) generated using the local symmetric pair-wise key (K_{AX}) and sent to Q_A, which is along the path to Q_{PDA}. Q_A checks the HMAC and, if it is authentic, will generate a new HMAC using the key K_{AB} and forward the message to Q_B. This process continues until the message arrives at the PDA, thus ensuring that only devices that are members of the WSN can forward the message. It is possible that nodes on the path to the sink are compromised and could drop packets. This could be dealt with at the routing algorithm level (there could be multiple paths to the sink). A compromised node will not be able to decrypt the message as they do not have the pair-wise key (K_{XPDA}) between the source and the sink.

5 Security of the Scheme

Various different attacks on a WSN are discussed in the following section.

5.1 Erroneous Data Insertion

A compromised or malfunctioning node may introduce erroneous data into the network and this scheme does not protect against this attack. Instead it is envisaged that the end user of the WSN will have software that will ignore data from a node that is not collaborated by other nodes in the same location.

5.2 Sinkhole Attack

In a sinkhole attack, a compromised device advertises a high quality route to a data extraction point, when it is not near one. This causes data to be routed to this malicious node, which can then drop the packets. As the nodes are aware of their position, this attack can be easily countered. If the device injects false routing information, to say that it is close to a distant area of the network, then as the nodes in the next hop are aware of their own position they will know that this could not be the case, and drop the packet. Also, if routing information is replayed from another section of the network then the receiving device will ignore the communication as the device from which the routing information originally is not a member of $\mathbb{C}_X$, where X is the node's identity.

5.3 Wormhole Attack

The wormhole attack [30] is where two devices, that are not nodes, and are geographically distant, conspire with each other to provide a low latency, undetectable (to the other devices in the WSN) route between them that is known as a wormhole. All communication between the source and sink would go through this wormhole as it appears to be a short path to the sink. The adversary could exploit this traffic to drop packets. The scheme defends against this attack as a node will only accept messages from a list of devices, $\mathbb{C}_X$, that it is authorized to communicate with.

5.4 Sybil Attack

Sybil attacks [31] can be mounted by compromised devices. In this attack the nodes present multiple identities to neighbouring devices in order to disrupt routing, or provide multiple readings to the network to make the local aggregated data value erroneous. Under the scheme presented, this attack is no longer feasible, as during normal operation the nodes only accept packets from their neighbours in $\mathbb{C}_X$. During the node addition phase they will only accept communication from devices in $\mathbb{E}_{KGC}$.

5.5 Identity Replication Attack

Unlike the Sybil attack, the identity replication attack [32] is based upon giving the same identity to different physical devices. This attack can be mounted because in a WSN there is no way to know that a node is compromised. If this device is cloned and placed in different parts of the network with the intention of disrupting the routing schemes then this attack can be overcome with the security scheme, since the nodes are only allowed to communicate with other devices that are members of $\mathbb{C}_X$. Hence, if a compromised device is placed in another part of the network it would not be able to join the WSN at that point.

6 Software profile of the Scheme

In order to evaluate whether a scheme based on SOK ID-NIKDS and BMLQ IBS is an appropriate choice for a WSN, the most computationally demanding components are implemented in software using the Miracl library [33]. The components that are profiled are exponentiation in the field $GF(2^{283\times4})$ (power), elliptic curve point multiplication (mult) and the Tate pairing (tate). In Table 1 the Tate pairing contribution to the total is counted twice as it is used in signature verification and symmetric key generation. The target device is the ARM920T [34] as a similar processor is used on the Imote2 device.

The code used to implement the scheme was compiled for the ARM using the ARM Development Suite (ADS) v1.2. As well as generating an executable that can be downloaded to the ARM using the JTAG inputs, it also gives timing figures for these executables.

The total energy dissipated is $35.4mJ$ and the power consumed is $0.05W$ at $140MHz$. The time required to run the scheme algorithm at $200MHz$ is $444.5ms$. Due to the nature of the experimental setup these figures are a lower bound of the energy dissipated by these component parts. The energy measurements should be taken at $200MHz$ as this will be the clock speed of the final system. This could not be achieved as the fastest clock speed that the board on which the measurements are taken on can run at is $140MHz$. It can be seen that a software implementation requires too much energy and its latency is unacceptably large for implementation on a node, therefore we believe that a hardware implementation of this scheme should be investigated.

TABLE 1
Timing and energy figures for main components of the scheme

code	algorithm	time (ms)	energy (mJ)
tate ($\times2$)	$e_l(P,Q)$	177.1	14.1
power	g^x	39.8	3.2
mult	rQ	50.6	4.0
	Total	444.5	35.4

From analysis of table 1 it is clear that the Tate pairing calculation is the most computationally demanding component of the scheme. It requires $14.1mJ$ to run which is considerable when the total energy budget of the nodes is of the order of $1000J$. A key design goal of nodes is that they operate on a low duty cycle and the fact that the Tate pairing takes $177.1ms$ is counter to this goal. A hardware implementation of the Tate pairing is therefore merited, as it will reduce the time it requires and also the energy it dissipates.

The software implemention of the scheme was undertaken in order to investigate whether a software solution of the Tate pairing would suffice for a WSN application and, if not, to identify key components that should be ported to hardware. Recent results has shown that the software implementation could be improved upon and significantly lower figures for the latency and energy arrived at, though they also conclude that a hardware implementation of pairings would be beneficial [10].

7 Arithmetic Operations

All operations that are used for the various algorithms in the hardware accelerator take place in binary extension fields; either $GF(2^{283})$ or $GF(2^{283\times4})$. If the Tate pairing calculation is rewritten as in Algorithm 1, then the arithmetic operations that are required are addition, multiplication, and inversion in both fields. In addition, a circuit is required to perform the squaring and square root operations in $GF(2^{283})$, and exponentiation in $GF(2^{283\times4})$.

For a design of this nature there is no real time constraint and so there can not be a latency that has to be met. Power is not important to this design, what is critical is the amount of energy that the device consumes. As a Lithium battery has a energy density of $2880J/cm^3$, which translates into $90\mu W/cm^3/year$ [35], then this figure of $2880J$ could be used as an energy constraint. It has been discussed previously that the device must operate on a low duty cycle to conserve energy; this requires a circuit that completes its operation quickly. At the same time the device should be as cheap as possible, and this would mean that the techniques of parallelism might not be appropriate as they will increase the area and hence the cost. Finally, when the circuit is operating it should consume as little energy as possible. These sometimes conflicting design goals of latency, area and energy are combined into a single metric known as area*energy*time (AET) which will be used to evaluate the circuits outlined in this work.

In the subfield $GF(2)$ addition is carried out using modulo two arithmetic, and hence can be performed in hardware using an XOR gate. Addition is equivalent to subtraction in $GF(2)$. Also, multiplication is performed using an AND gate in hardware.

The polynomial basis representation is used for the elements of the two finite field such that for $\alpha \in GF(2^{283})$

$$\begin{aligned}\alpha = A(x) = a_{282}x^{282} + a_{281}x^{281} + \cdots + a_0 \pmod{f(x)},\\ \forall a_j \in GF(2).\end{aligned} \tag{9}$$

When $\alpha \in GF(2^{283\times 4})$ then

$$\begin{aligned}\alpha = A(x) = a_3x^3 + a_2x^2 + a_1x + a_0 \pmod{p(x)},\\ \forall a_j \in GF(2^{283}).\end{aligned} \tag{10}$$

An irreducible polynomial (11) is chosen

$$f(x) = x^{283} + x^{119} + x^{97} + x^{93} + 1, \tag{11}$$

such that it has an odd exponent polynomial which means that the square root operation can be carried out in one clock cycle.

The polynomial for generating $GF(2^{283\times 4})$ is

$$p(x) = x^4 + x + 1, \tag{12}$$

and it is defined over $GF(2^{283})$.

7.1 Addition

Addition in a binary extension field is trivial to implement in hardware. It is an array of XOR gates, one for every two bits of the operands that are to be added. Hence for $GF(2^{283})$ 283 XOR gates are required, and for $GF(2^{283\times 4})$ 1132 are required.

7.2 Multiplication in $GF(2^{283})$

WSNs will be deployed in practice only if the devices that make up the network are cheap. In terms of multiplication in $GF(2^{283})$ a fast bit-parallel multiplier is approximately 300,000 gates in area. This would be prohibitive in terms of manufacturing cost for a wireless sensor node. Thus, a bit-serial approach to designing the multiplier is warranted. There are two approaches to a bit-serial multiplier – an Most Significant Bit (MSB) first design or Least Significant Bit (LSB) first design [36].

7.2.1 The MSB multiplier

The MSB multiplier is based on the following observation.

$$\begin{aligned}C(x) &= A(x)B(x)\\ &= (a_{282}x^{282} + \cdots + a_1x + a_0)(b_{282}x^{282} + \cdots\\ &+ b_1x + b_0) \pmod{f(x)}\\ &= b_0 + x(b_1A(x) + \cdots x(b_{280}A(x) + x(b_{281}A(x)\\ &+ (xb_{282}A(x)) \pmod{f(x)}))\end{aligned} \tag{13}$$

From algorithm 2 it can be seen that this circuit will require at least 283 clock cycles to complete.

Algorithm 2 MSB Multiplication in $GF(2^{283})$

```
Require: $C(x) = A(x)B(x) \pmod{f(x)}$
  $C(x) = 0$
  $count = 0$
  while $count < 282$ do
    if $B(282) = 1$ then
      $C(x) = C(x) + A(x)$
    end if
    left shift $B(x)$
    $C(x) = xC(x) \pmod{f(x)}$
    $count = count + 1$
  end while
  if $B(282) = 1$ then
    $C(x) = C(x) + A(x)$
  end if
  return $C(x)$
```

7.2.2 The LSB multiplier

Another approach to bit-serial multiplication in $GF(2^{283})$ is to use a LSB multiplier. The LSB multiplier is based on the following observation.

$$\begin{aligned}C(x) &= A(x)B(x)\\ &= (a_{282}x^{282} + \cdots + a_1x + a_0)(b_{282}x^{282}\\ &+ \cdots + b_1x + b_0) \pmod{f(x)}\\ &= (b_{282}x^{282}A(x) + \cdots + b_1xA(x)\\ &+ b_0A(x)) \pmod{f(x)}\\ &= (b_{282}x^{282}A(x) \pmod{f(x)}) + \cdots\\ &+ (b_1xA(x) \pmod{f(x)}) + (b_0A(x) \pmod{f(x)})\end{aligned} \tag{14}$$

$C(x)$ can be calculated using a shift and add algorithm where the first partial product is $b_0A(x)$. $B(x)$ is then shifted right one bit while at the same time $A(x)$ is multiplied by x and reduced mod $f(x)$. It is added to the previous product if b_1 is equal to one. The algorithm will terminate when the value of the right shift register is equal to zero (see Algorithm 3). This is an early exit mechanism, as it could finish after one clock cycle or 283 clock cycles.

From an analysis of the algorithm it can seen that the addition of right shift and linear feedback barrel shift registers can be used to improve the performance of the circuit. Two,

Algorithm 3 LSB Multiplication in $GF(2^{283})$

```
Require: $C(x) = A(x)B(x) \pmod{f(x)}$
  $C(x) = 0$
  while $B(x) \neq 0$ do
    $C(x) = C(x) + b_0 A(x)$
    right shift $B(x)$
    $A(x) = xA(x) \pmod{f(x)}$
  end while
```

three, four or five consecutive zero bits are searched for, and the registers shifted accordingly. As there is a cost in terms of extra area for every extra bit searched for, it was decided that five would be the most bits considered. This is because the probability of five zeros is $\frac{1}{32}$ and the probability of more than five zeros is low. The datapath circuitry is shown in Figure 1.

7.3 Multiplication in $GF(2^{283\times 4})$

Multiplication of two elements

$$\gamma = \alpha\beta, \quad \forall \alpha, \beta \in GF(2^{283\times 4})$$

is required. As the multiplication circuitry will exist for $GF(2^{283})$, it can be used to perform the multiplication for $GF(2^{283\times 4})$ using Karatsuba and Ofman's algorithm [37]. This sharing of resources will lead to a decrease in the monetary cost of the system.

As the elements are represented using the polynomial basis (10). Then the multiplication is as follows.

$$\left(\sum_{i=0}^{3} c_i x^i\right) = \left(\sum_{i=0}^{3} a_i x^i\right)\left(\sum_{i=0}^{3} b_i x^i\right) \pmod{p(x)} \quad (15)$$

By applying the Karatsuba algorithm the resultant equation is

$$\begin{aligned} c_0 &= a_2b_2 + (a_1+a_3)(b_1+b_3) + a_0b_0 + a_3b_3 + a_1b_1 \\ c_1 &= (a_2+a_3)(b_2+b_3) + (a_1+a_3)(b_1+b_3) \\ &\quad + (a_0+a_1)(b_0+b_1) + a_0b_0, \\ c_2 &= (a_2+a_3)(b_2+b_3) + a_1b_1 + (a_0+a_2)(b_0+b_2) + a_0b_0 \\ c_3 &= a_0b_0 + (a_0+a_1)(b_0+b_1) + a_1b_1 + (a_0+a_2)(b_0+b_2) \\ &\quad + (a_0+a_2+a_1+a_3)(b_0+b_2+b_1+b_3) \\ &\quad + (a_1+a_3)(b_1+b_3) + a_2b_2 + (a_2+a_3)(b_2+b_3) \end{aligned} \quad (16)$$

Using terms common to more than one equation i.e. $a_0b_0 + a_1b_1$ it can be seen that 12 additions are required in $GF(2^{283})$. In total, 9 multiplications and 22 additions are required in $GF(2^{283})$ when the Karatsuba algorithm is employed.

The datapath circuitry is shown in Figure 2. The datapath width is 283 bits wide. In order to reduce dynamic energy dissipation, wires are held at a constant value when not in use. This is accomplished through the signals enadd10 and enadd12 (not shown), which gate the inputs and the combinational logic, respectively. The LSB multipliers clocks are to be gated with their "done" signals. This technique takes advantage of the early exit of the LSB multipliers due to their structure.

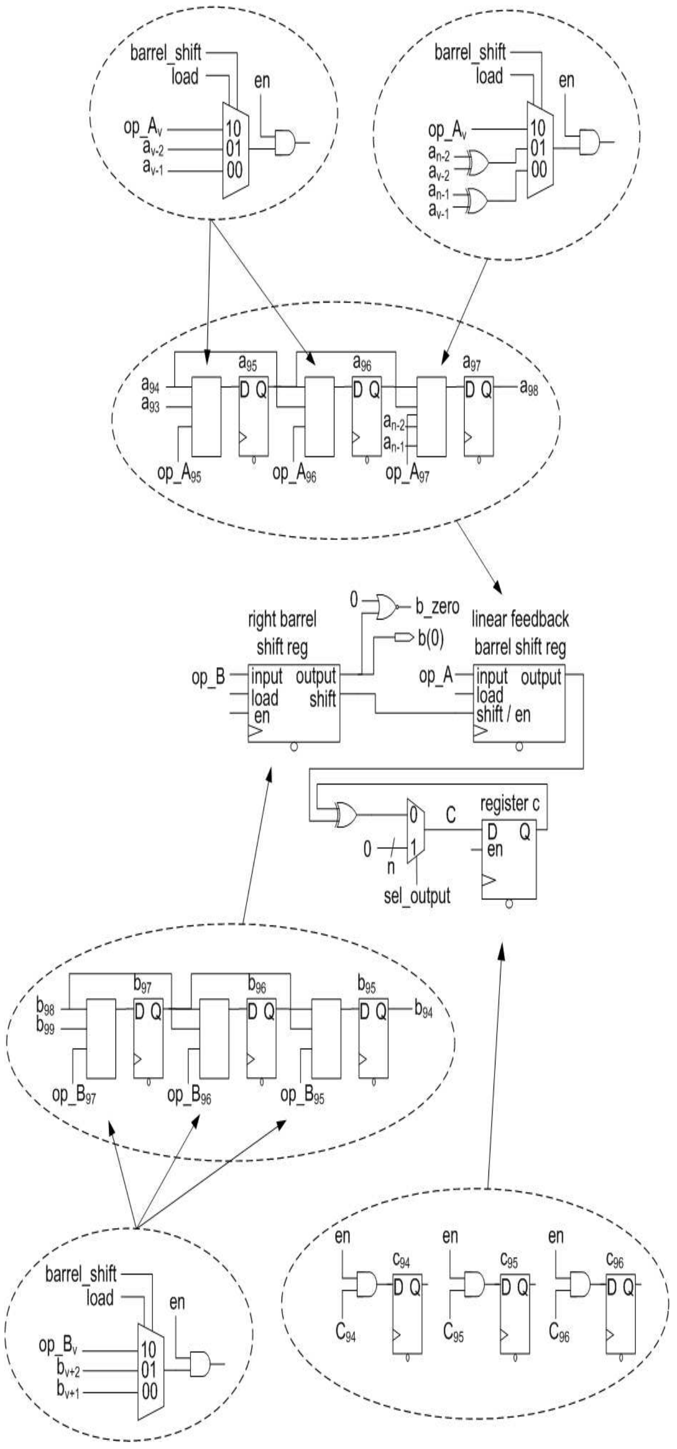

Fig. 1. mult_lsb: Datapath circuit for the LSB multiplier in $GF(2^{283})$

7.4 Squaring

The bit-serial multiplier described in Section 7.2 could be used for squaring, but as squaring is used 283 times in each loop and in the inversion circuitry, this is not the optimum choice. Instead, a bit-parallel squaring circuit has been implemented.

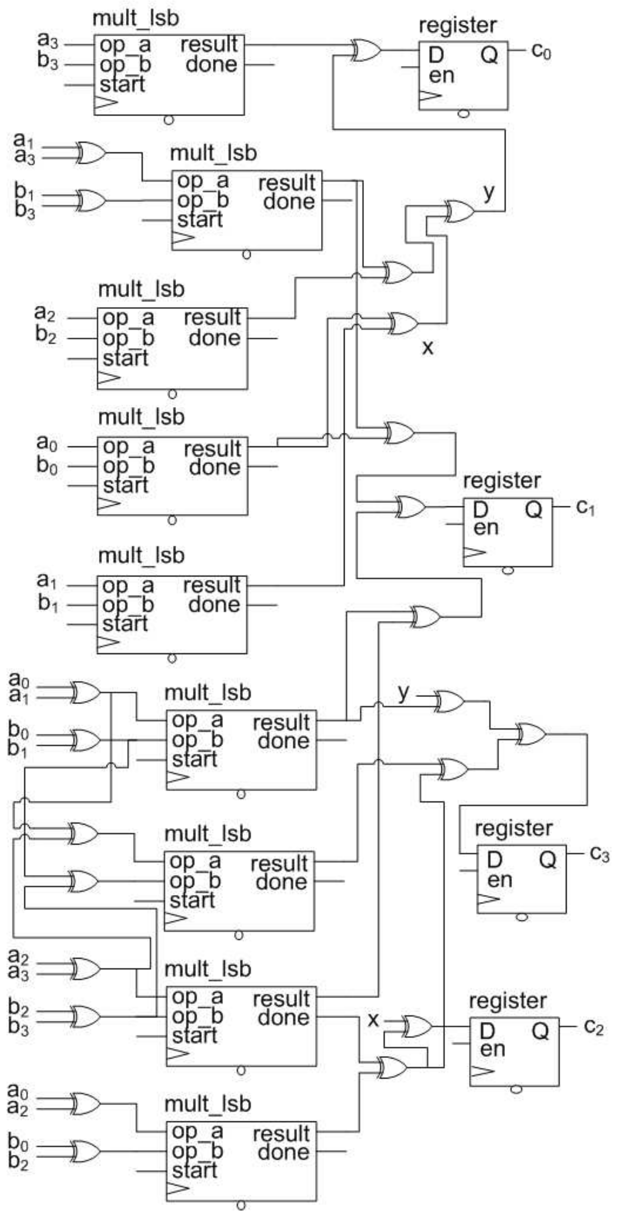

Fig. 2. mult_koa: Datapath circuit for the multiplier in $GF(2^{283\times4})$

For example, if given $C(x), A(x) \in GF(2^4)$ then

$$\begin{aligned}C(x) &= (A(x))^2 \pmod{x^4+x+1}\\ &= (a_3x^3+a_2x^2+a_1x+a_0)^2 \pmod{x^4+x+1}\\ &= a_3x^3+(a_1+a_3)x^2+a_2x+(a_0\\ &\quad +a_2) \pmod{x^4+x+1}\end{aligned} \tag{17}$$

This can be implemented with two XOR gates and a reordering of the inputs. With the aid of a C++ program this technique can be applied to elements from $GF(2^{283})$. The resulting matrix can be converted into hardware using Very High Speed Integrated Circuit Hardware Description Language (VHDL).

7.4.1 Square root circuit

Using the techniques of Fong et al [38], it is possible to reduce the latency of the square root operation to one clock cycle. Given

$$\begin{aligned}\sqrt{\alpha} &= \alpha^{2^{m-1}} \pmod{g(x)},\\ &= \left(\sum_{i=0}^{m-1} a_i x^i\right)^{2^{m-1}} \pmod{g(x)}\end{aligned} \tag{18}$$

where

$$g(x) = x^m + x^t + x^u + x^v + 1, \tag{19}$$

and

$$\begin{aligned}\alpha &= a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \cdots\\ &\quad + a_1x^1 + a_0 \pmod{g(x)}\end{aligned} \tag{20}$$

All of the exponents in Equation (19) are odd. Equation (18) can be further developed as below;

$$\begin{aligned}\sqrt{\alpha} &= \left(\sum_{i=0}^{m-1} a_i x^i\right)^{2^{m-1}}\\ &= \sum_{i=0}^{m-1} a_i \left(x^{2^{m-1}}\right)^i\\ &= \sum_{i=0}^{(m-1)/2} a_{2i}\left(x^{2^{m-1}}\right)^{2i} + \sum_{i=0}^{(m-3)/2} a_{2i+1}\left(x^{2^{m-1}}\right)^{2i+1}\\ &= \sum_{i=0}^{(m-1)/2} a_{2i}x^i + \sum_{i=0}^{(m-3)/2} a_{2i+1}x^{2^{m-1}}x^i\\ &= \alpha_{even} + \alpha_{odd}\sqrt{x}\end{aligned} \tag{21}$$

From Equation (19) it can be seen that

$$\begin{aligned}1 &= x^m + x^t + x^u + x^v \pmod{g(x)}\\ x &= x^{m+1} + x^{t+1} + x^{u+1} + x^{v+1} \pmod{g(x)}\\ \sqrt{x} &= x^{(m+1)/2} + x^{(t+1)/2} + x^{(u+1)/2}\\ &\quad + x^{(v+1)/2} \pmod{g(x)}\end{aligned} \tag{22}$$

Therefore

$$\sqrt{\alpha} = \alpha_{even} + \alpha_{odd}(x^{(m+1)/2} + x^{(t+1)/2} + x^{(u+1)/2} + x^{(v+1)/2}) \tag{23}$$

In the case of $\alpha \in GF(2^{283})$

$$\sqrt{\alpha} = \alpha_{even} + \alpha_{odd}(x^{142} + x^{60} + x^{49} + x^{42}) \tag{24}$$

and the exponents are taken from Equation (11). This can be implemented in hardware using XOR gates in one clock cycle.

7.5 Exponentiation

The only exponentiation that is required for the Tate pairing calculation is $\beta = \alpha^{2^{283}}$ where $\alpha, \beta \in GF(2^{283\times4})$. This is also known as the Frobenius map.

Using Equation (10) the exponentiation is as follows;

$$\begin{aligned}\sum_{i=0}^{3} b_i x^i &= \left(\sum_{i=0}^{3} a_i x^i\right)^{2^{283}} \\ &= \sum_{i=0}^{3} a_i^{2^{283}} x^{i2^{283}} \\ &= \sum_{i=0}^{3} a_i x^{i2^{283}} \pmod{p(x)}. \\ &= (a_0 + a_1) + (a_2 + a_3)x + a_1 x^2 + a_3 x^3\end{aligned} \tag{25}$$

For a proof see [39].The Frobenius map can therefore be implemented in hardware with two additions in $GF(2^{283})$ and reordering of the coefficients.

7.6 Inversion in $GF(2^{283})$

There are two well-known techniques for inversion of $\beta \in GF(2^{283})$. One approach is based on Fermat's little theorem and the other uses the extended Euclidean algorithm.

7.6.1 *Inversion by Fermat's Little Theorem*

Fermat's little theorem (see Equation (26)) can be used to invert an element of $GF(2^{283})$

$$\beta^{2^{283}-1} \equiv 1 \pmod{f(x)}. \tag{26}$$

This means that $\beta^{2^{283}-2}\beta \equiv 1 \pmod{p(x)}$ and therefore $\beta^{2^{283}-2}$ is the inverse of β. The inverse of β can be calculated with the square and multiply technique using the following observations.

$$\begin{aligned}\beta^{-1} &= \beta^{2^{283}-2} = \beta^{2^1}\beta^{2^2}\beta^{2^3}\cdots\beta^{2^{282}} \\ &= (\cdots(((\beta)^2\beta)^2\beta)^2\cdots\beta)^2\end{aligned} \tag{27}$$

This algorithm requires 282 squarings and 281 multiplications in $GF(2^{283})$. From section 7.2 and 7.4, it can be seen that multiplication in $GF(2^{283})$ is a very expensive operation in terms of time and energy. It would be beneficial to reduce the number of multiplications. This can achieved using the techniques of Itoh and Tsujii [40].

As

$$\beta^{-1} = \beta^{2^n-2} = \left(\beta^{2^{n-1}-1}\right)^2$$

for a field $GF(2^n)$ then we can apply the following recursive formula to reduce the number of multiplications. When n is odd then

$$\beta^{2^{n-1}-1} = \left(\beta^{2^{\frac{n-1}{2}}-1}\right)^{2^{\frac{n-1}{2}}} \beta^{2^{\frac{n-1}{2}}-1} \tag{28}$$

and when n is even

$$\beta^{2^{n-1}-1} = \left(\beta^{2^{n-2}-1}\right)^2 \beta \tag{29}$$

β^{-1} can now be decomposed using Equations (28) and (29) resulting in only 11 multiplications and 282 squarings are required to obtain the inverse of β. The datapath circuitry is shown in Figure 3. 3.

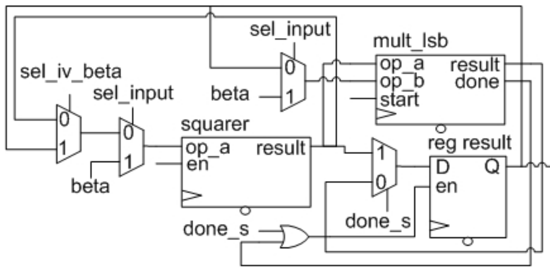

Fig. 3. Datapath circuit for the inverter in $GF(2^{283})$ using Fermat's little theorem

Algorithm 4 Extended Euclidean Algorithm for inversion in $GF(2^{283})$[16]

```
Require: $C(x) = A(x)^{-1} \pmod{f(x)}$
  $u = A(x), v = f(x), g_1 = 1, g_2 = 0$
  while $u \neq 1$ do
    $j = 0$
    if $v > u$ then
      $u = v, v = u, g_1 = g_2, g_2 = g_1$
    else
      $j$ is equal to degree of $u$ minus degree of $v$
    end if
    $u = u + x^j v$
    $g_1 = g_1 + x^j g_2$
  end while
  return $C(x) = g_1$
```

7.6.2 *Inversion by the Extended Euclidean Algorithm*

The Extended Euclidean Algorithm is implemented using algorithm 4. The datapath circuitry is shown in Figure 4. This block uses the degree sub-block to measure the degree of the polynomials u and v.

7.7 Inversion in $GF(2^{283\times 4})$

Fermat's little theorem (see Equation (26)) can also be used to get the inversion of an element $\alpha \in GF(2^{283\times 4})$ where the extension field of $GF(2^{283})$ is obtained using the irreducible polynomial given in Equation (12). The technique below, that has been used by Paar and Guajardo [41], is used as it makes use of circuits that are already designed.

If the general case $\alpha \in GF(2^{mk})$ is considered, then the inverse is

$$\begin{aligned}\alpha^{-1} &= \alpha^{2^{mk}-2} = \alpha^{\frac{2^{mk}-1}{2^m-1}(2^m-1)-1} \\ &= \alpha^{r(2^m-2)+r-1} = (\alpha^r)^{-1}\alpha^{r-1}\end{aligned}$$

where $r = \frac{2^{mk}-2}{2^m-1}$. The technique is based on the fact that

$$\alpha^r \in GF(2^m), \qquad \forall \alpha \in GF(2^{mk}) \tag{30}$$

When working in the field $GF(2^{283\times 4})$ the first stage of the inversion algorithm (5) is obtained by the following equations.

$$r - 1 = 2^{283} + (2^{283})^2 + (2^{283})^3.$$

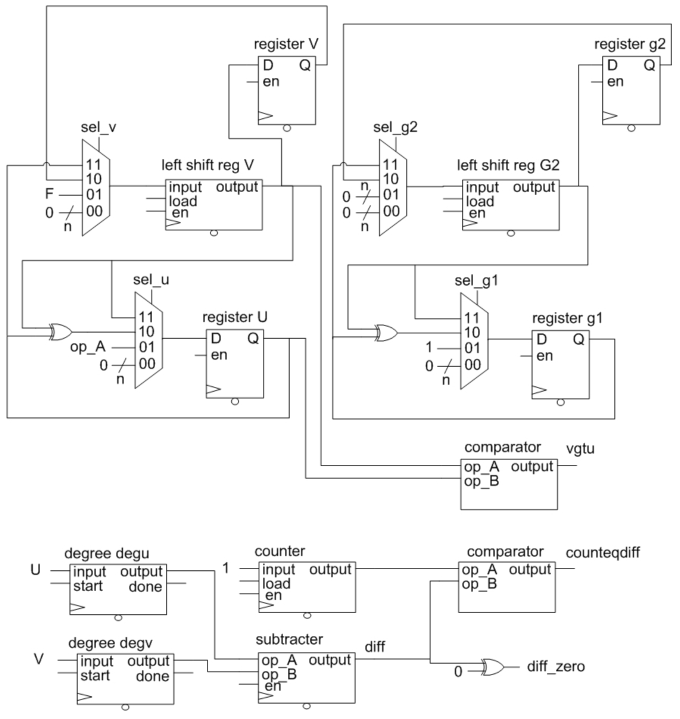

Fig. 4. Datapath circuit for the inverter in $GF(2^{283})$using the Extended Euclidean Algorithm

Algorithm 5 Inversion in $GF(2^{mk})$

```
Require: $\beta = \alpha^{-1}$
  $\alpha^{r-1}$
  $\alpha^r = \alpha^{r-1}\alpha$
  $(\alpha^r)^{-1}$
  $(\alpha^r)^{-1}\alpha^{r-1}$
```

If we let

$$\beta = \alpha^{r-1} = \alpha^{2^{283}+(2^{283})^2+(2^{283})^3}$$

where $\alpha, \beta \in GF(2^{283\times4})$ this can be rewritten as

$$\beta = \left(\left(\alpha^{2^{283}}\alpha\right)^{2^{283}}\alpha\right)^{2^{283}}$$

Three 2^{283} exponentiations (see section 7.5) and two multiplications in $GF(2^{283\times4})$ (see section 7.3) are required to perform this operation. The next stage is multiplication in $GF(2^{283\times4})$. As $\alpha \in GF(2^{283})$, $(\alpha^r)^{-1}$ is inversion in $GF(2^{283})$ as outlined earlier in this section, and finally the last step is multiplication in $GF(2^{283\times4})$.

The datapath circuitry is shown in Figure 5.

7.8 Results

Table 2 presents the results of all the circuits that have been discussed in terms of their latency, energy, area and AET metric. The operations are listed as follows; mult_lsb* is the LSB $GF(2^{283})$ multiplier, mult_msb is the MSB $GF(2^{283})$ multiplier, square is the squaring circuit, sqroot is the square root operation, mult_koa is the Karatsuba algorithm multiplier, inv_ferm is the $GF(2^{283})$ inverter and inv_gf2m4 is the $GF(2^{283\times4})$ multiplier.

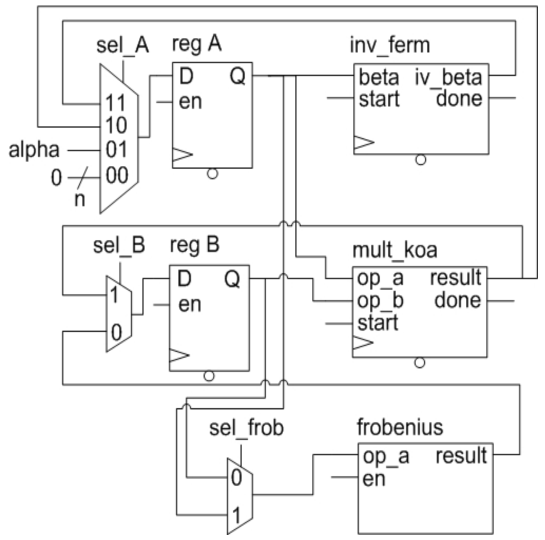

Fig. 5. Datapath circuit for the inverter in $GF(2^{283\times4})$

The choice of the $GF(2^{283})$ multiplier is crucial as it is the building block upon which most of the other arithmetic operations are based. The area of the MSB multiplier (mult_msb) is $0.024mm^2$, it has a latency of $2.32\mu s$ and consumes $3.94nJ$ of energy. The area of the LSB multiplier (mult_lsb1) is $0.022mm^2$, it has a latency of $2.31\mu s$ and consumes $4.46nJ$ of energy. Therefore, this multiplier has 11% less area and uses 13% more energy than the MSB multiplier. Using the AET metric it can be seen that the differences between the two approaches is less than 1%. From an analysis of the multipliers presented in Table 2, where mult_lsb2 represents the search for the two LSB being zero etc., it can be seen that mult_lsb3 has the lowest AET figure. This is the multiplier that is used in this work.

It can be noted that the Karatsuba algorithm multiplier has a latency of $1.93\mu s$, which is comparable to the LSB multiplier due to its parallel architecture. Also the results for the squaring and sqroot circuit are of the same order.

The inverter based on Fermat's little theorem (inv_ferm) is $0.044mm^2$. It uses $68.1nJ$ of energy and require $22.65\mu s$ to run to completion. From Table 2 it can be seen that the area of the Extended Euclidean inverter (euclid) is 43% greater. It is also an order of magnitude slower and uses two magnitudes more of energy. Thus, by all the metrics used to evaluate designs in this paper, the inverter based on Fermat's little theorem is the preferred choice for inversion in $GF(2^{283})$.

8 Tate Pairing Accelerator Architecture

In this section the arithmetic units presented previously are incorporated into a Tate pairing hardware accelerator. A top-level block diagram of the architecture of our proposed accelerator is given in Figure 6. The device is connected with the host using the Advanced Peripheral Bus (APB) scheme. The Host Interface block interfaces with the APB. It is responsible for

TABLE 2
Results for $GF(2^{283})$ and $GF(2^{283\times 4})$ arithmetic primitives

operation	area (mm^2)	time (μs)	time (cycles)	energy (nJ)	AET E-22mSJ
mult_lsb1	0.022	2.31	289	4.46	2.24
mult_lsb2	0.023	1.897	237	4.23	1.87
mult_lsb3	0.025	1.776	222	4.14	1.8
mult_lsb4	0.026	1.717	215	4.06	1.83
mult_lsb5	0.028	1.708	214	4.15	2.01
mult_msb	0.024	2.32	290	3.94	2.22
mult_koa	0.295	1.926	241	43.4	247
inv_ferm	0.044	22.65	2831	68.1	683
euclid	0.063	341.436	42680	1030	2.23E+05
inv_gf2m4	0.417	29.863	3733	541	67500
square	0.005	0.008	1	0.004	1.60E-06
sqroot	0.008	0.008	1	0.004	2.47E-06

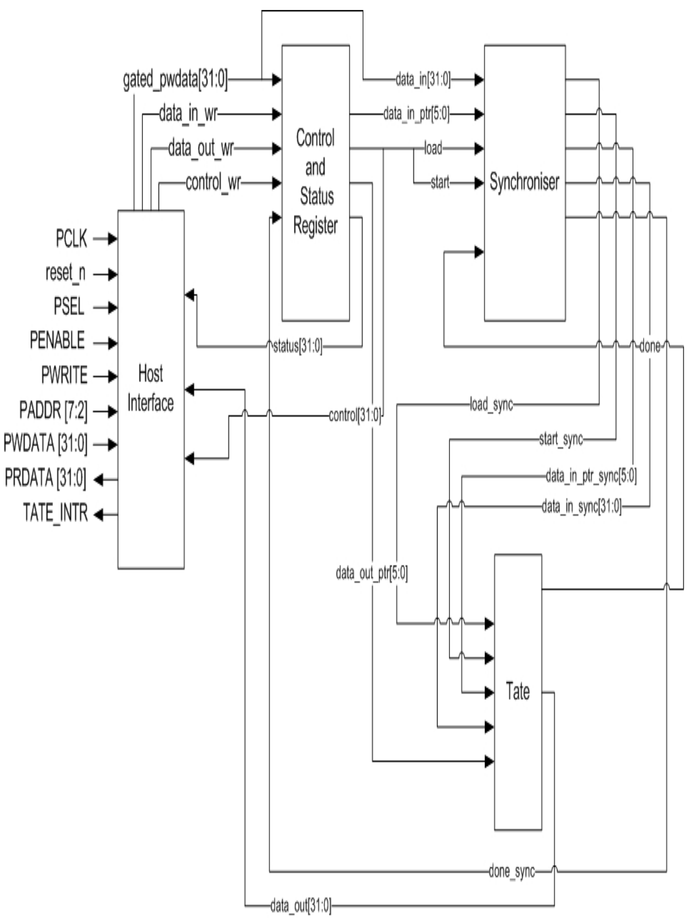

Fig. 6. Tate Pairing Hardware Accelerator Architecture

decoding the write data input signal and address signal to write to the internal registers. The read data output is muxed in this block with the control, status and resultant data from the Tate pairing accelerator. These internal buses are gated when not in use to conserve energy.

The control and status register block is used to implement the control register, status register, and the read and write data pointers. Accessing the data registers in this block has the effect of updating the write or read pointers, which are implemented as six bit counters. Writes and reads to the data registers are in fact writes and read to registers in the Tate block.

There are two clock domains in the design; the APB clock which is PCLK and runs at $50MHz$ and a system clock, tate_clk, which runs at $200MHz$. These clocks are assumed to be asynchronous. Signals that cross the clock boundary will require synchronisation and this is achieved in the synchronisation block.

Finally, the Tate block implements the algorithm for calculating the Tate pairing using algorithm 1. Two points on the curve are required to calculate the Tate pairing and these will have coordinates in $GF(2^{283})$. Therefore four 283 bit values have to be written into the device before the Tate block can be initiated. When it is finished it sends an interrupt back to the host interface (tate_intr).

8.1 Datapath

The datapath is presented in Figure 7. The databus is 283×4 bits wide. The tate_in register is arranged in 5×283 bit registers that can be written to in 32 bit or 283 bit mode. When the elliptic curve points are written into the accelerator they are accessed in 32 bit mode by the processor via the host interface block. When the operation is finished a "done" signal is asserted which is assigned to an interrupt. Upon detection of this interrupt the processor can read the output data, 32 bits at a time, from the tate_out register via reads to the tate_data_out register.

The zw_alu Arithmetic logic Unit (ALU) calculates $z = x_p + x_q$ and $w = z + x_p x_q + y_p + y_q + 1$. Internally, it has two squaring and square root circuits instantiated. It first is used to calculate $x_p = x_p^2$ and $y_p = y_p^2$. The tate_alu ALU then calculates $x_p x_q$. Then z and w can be arrived at. As well as the above operations, this ALU also calculates $x_q = x_q^{2^{283-1}}$ and $y_q = y_q^{2^{283-1}}$.

The tate_alu ALU calculates the multiplication in $GF(2^{283})$ and is also responsible for all other arithmetic operations required by the algorithm such as; multiplication, inversion and exponentiation – all in $GF(2^{283\times 4})$. As was discussed above, the inversion circuit in $GF(2^{283\times 4})$ requires an inverter in $GF(2^{283})$ and this is implemented in this ALU.

There are two sub-circuits implemented in the tate_alu; one for calculating exponentiation and the other for multiplication in both fields and inversion in $GF(2^{283})$. This last circuit is based upon the Karatsuba algorithm multiplier but with the modification that one of the $GF(2^{283})$ multipliers is replaced by a module that performs inversion and multiplication in $GF(2^{283})$.

The datapath is controlled by a finite state machine (FSM) with nineteen states. The data is written into the Tate block by a write to the tate_load bit in the control register, whilst at the same time data is written to the tate_data_in register. A write to the tate_start bit in the control register initiates the FSM. It executes 283 loops of the main body of the algorithm and this

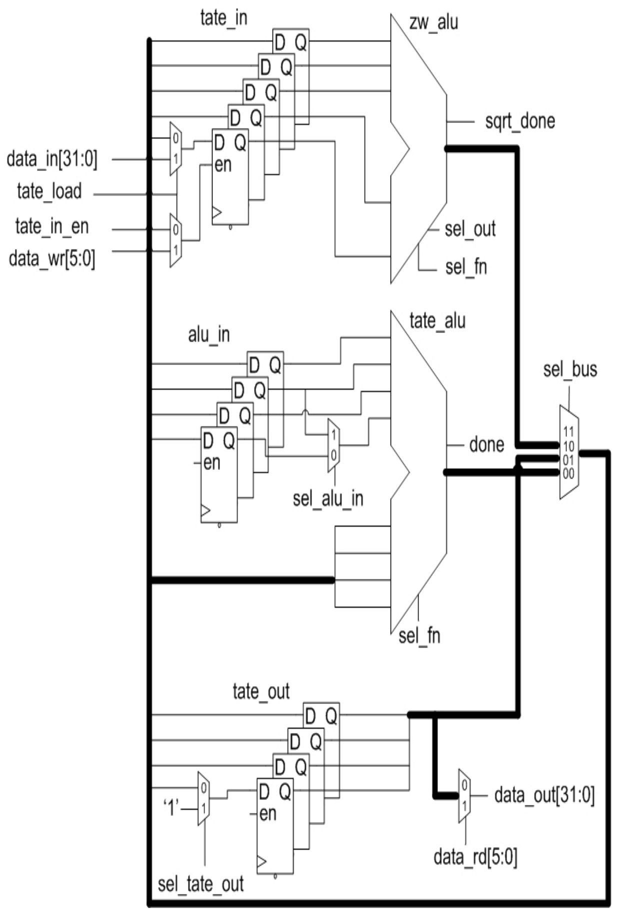

Fig. 7. Datapath for the Tate Pairing Hardware Accelerator

is controlled by a counter. When the FSM is finished then the tate_done signal is asserted and this is recorded in the status register and sent to the processor as an interrupt.

8.2 Experimental Results

The design was implemented using VHDL to be incorporated in an Application Specific Integrated Circuit (ASIC). The target technology is a Taiwan Semiconductor Manufacturing Company (TSMC) 65nm low power CMOS process [42]. Worst case operating conditions are used and these are; voltage $1.08V$, process 1.000 and temperature $125°C$. It is synthesised for a clock of frequency of $200MHz$. Synthesis and physical synthesis is performed using Synopsys Design Compiler and Physical Compiler, respectively [43]. Synopsys PrimePower is used to arrive at a figure for the power consumption of the circuit. Clock power was not included in this figure. Results for the Tate pairing accelerator are shown in Table 3. From this it can be seen that it has a latency of $0.7ms$, an area of $0.574mm^2$ and consumes $29600nJ$.

9 Discussion

In terms of the overall system the inclusion of a hardware accelerator for the Tate pairing will lead to a 79% reduction in latency and energy when compared with running the scheme

TABLE 3
Results for the Tate pairing accelerator

operation	area (mm^2)	time (μs)	time (cycles)	energy (nJ)	AET E-22mSJ
tate	0.574	698.11	139622	29600	1.19E+08

in software. Even with this improvement in performance, it is evident, with the scheme still taking $91.7ms$ and consuming $7.3mJ$, that these percentage drops are not enough to justify the deployment of IBC in a WSN. Further significant improvements can be attained if the power operation (g^r) and the mult operation (rP) were also accelerated. Assuming that the same reductions in latency and energy consumption can be attained as for the Tate pairing calculation, then the figures are as in Table 4. As the time for running the scheme is now $1.75ms$

TABLE 4
Effect of accelerating on IBC scheme

	none	Tate		Tate/rQ/g^r	
	value	value	decrease	value	decrease
energy	$35.4mJ$	$7.26mJ$	79.5%	$0.08mJ$	99.8%
time	$444.5ms$	$91.7ms$	79.4%	$1.75ms$	99.6%

and it consumes $0.08mJ$ of energy, then it is probable that the latency and energy figures for the scheme would be in appropriate range for a WSN.

There are several groups which have implemented the Tate pairing in hardware but are targeting the latency metric and also Field Programmable Gate Arrays (FPGAs) rather than energy and an ASIC, therefore these contributions are not a valid comparison with this work [44], [44], [45], [46], [47]. The recent work of Szczechowiak el al. has the lowest reported energy figure for a software implementation of the Tate pairing which is $3.76mJ$ for the Imote2 [10]. This is, as would be expected, several magnitudes higher than what is achieved by implementing the pairing in hardware, and too high for the limited energy available to a node.

10 Conclusion

In this paper we presented a solution for distributing symmetric keys and network access control in a WSN using IBC. The proposed scheme was evaluated against well known attacks on a WSN and found to perform well. It was then profiled in software, and the most computationally demanding component of the scheme, the Tate pairing, was ported to hardware. We presented our hardware design for the Tate pairing and evaluated it against key metrics. Experimental results indicate that whilst this work represents significant progress in terms of implementing security in a WSN, further work is needed in order to implement other components of the scheme such as the elliptic curve point multiplication and exponentiation in the field $GF(2^{283\times4})$.

ACKNOWLEDGMENTS

This work is supported by Science Foundation Ireland under grant number 07/CE/I1147 and the Informatics Commercialisation initiative of Enterprise Ireland.

REFERENCES

[1] J. M. Rabaey, M. Ammer, J. L. da Silva Jr., D. Patel, and S. Roundy, "PicoRadio supports ad hoc ultra-low power wireless networking," *Computer Magazine*, pp. 42–48, Jul. 2000.

[2] J. L. Hill, "System architecture for wireless sensor networks," Ph.D. dissertation, University of California, Berkeley, 2003. [Online]. Available: http://www.cs.berkeley.edu/~jhill

[3] A. Perrig, J. Stankovic, and D. Wagner, "Security in wireless sensors networks," *Communications of the ACM*, vol. 47, no. 6, pp. 53–57, Jun. 2004.

[4] H. Chan and A. Perrig, "Security and privacy in sensors networks," *IEEE Computer*, vol. 36, no. 10, pp. 103–105, Oct. 2003.

[5] C. Savarese, J. M. Rabaey, and J. Beutel, "Locationing in distributed ad-hoc wireless sensor networks," in *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2001.

[6] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proc. Crypto '84*, Santa Barbara, California, USA, Aug. 1984, pp. 47–54.

[7] B. Doyle, S. Bell, A. F. Smeaton, K. McCusker, and N. O'Connor., "Security considerations and key negotiation techniques for power constrained sensor networks," *The Computer Journal (Oxford University Press)*, vol. 49, no. 4, pp. 443–453, 2006.

[8] H.-b. Cheng, G. Yang, J.-t. Wang, and X. Huang, "An authenticated identity-based key establishment and encryption scheme for wireless sensor networks," *The Journal of China Universities of Posts and Telecommunications*, vol. 13, no. 1, pp. 31–38, 2006.

[9] L. Oliveira, M. Scott, J. Lopez, and R. Dahab, "Tinypbc: Pairings for authenticated identity-based non-interactive key distribution in sensor networks," in *Networked Sensing Systems, 2008. INSS 2008. 5th International Conference on*, June 2008, pp. 173–180.

[10] P. Szczechowiak, A. Kargl, M. Scott, and M. Collier, "On the application of pairing based cryptography to wireless sensor networks," in *WiSec '09: Proceedings of the second ACM conference on Wireless network security*. New York, NY, USA: ACM, 2009, pp. 1–12.

[11] Y. H. Kim, H. Lee, J. H. Park, L. T. Yang, and D. H. Lee, "Key establishment scheme for sensor networks with low communication cost," in *Autonomic and Trusted Computing*, ser. Lecture Notes in Computer Science, vol. 4610. Springer, 2007, pp. 441–448.

[12] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Location-based compromise-tolerant security mechanisms for wireless sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 247–260, Feb. 2006.

[13] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," *SIAM J. of Computing*, vol. 32, no. 3, pp. 586–614, 2003.

[14] *MICA2 Wireless Measurement System*, Crossbow Technology. [Online]. Available: http://www.xbow.com

[15] R. Adler, M. Flanigan, J. Huang, R. Kling, N. Kushalnagar, L. Nachman, C.-Y. Wan, and M. Yarvis, "Intel mote 2: an advanced platform for demanding sensor network applications," in *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM Press, 2005, pp. 298–298.

[16] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2003.

[17] L. C. Washington, *Elliptic Curves, Number Theory and Cryptography*. London, UK: Chapman & Hall/CRC, 2003.

[18] I. Blake, G. Seroussi, N. Smart, and J. W. S. Cassels, *Advances in Elliptic Curve Cryptography (London Mathematical Society Lecture Note Series)*. New York, NY, USA: Cambridge University Press, 2005.

[19] I. M. Duursma and H.-S. Lee, "Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$." in *ASIACRYPT*, 2003, pp. 111–123.

[20] S. Kwon, "Efficient Tate pairing computation for elliptic curves over binary fields." in *ACISP*, 2005, pp. 134–145.

[21] P. S. L. M. Barreto, S. Galbraith, C. O. hEigeartaigh, and M. Scott, "Efficient pairing computation on supersingular abelian varieties," Cryptology ePrint Archive, Report 2004/375, 2004, http://eprint.iacr.org/.

[22] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*. New York, NY, USA: ACM, 2002, pp. 41–47.

[23] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Security and Privacy, 2003. Proceedings. 2003 Symposium on*, May 2003, pp. 197–213.

[24] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-secure key distribution for dynamic conferences," in *Advances in Cryptology*, ser. Lecture Notes in Computer Science, vol. 740. Springer, 1993, pp. 471–486.

[25] R. Blom, "An optimal class of symmetric key generation systems," in *Advances in Cryptology: Proceedings of EUROCRYPT 84*, ser. Lecture Notes in Computer Science, vol. 209. Springer, 1985, pp. 335–338.

[26] D. Liu, P. Ning, and R. Li, "Establishing pairwise keys in distributed sensor networks," *ACM Trans. Inf. Syst. Secur.*, vol. 8, no. 1, pp. 41–77, 2005.

[27] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili, "A pairwise key predistribution scheme for wireless sensor networks," *ACM Trans. Inf. Syst. Secur.*, vol. 8, no. 2, pp. 228–258, 2005.

[28] R. Sakai, K. Ohgishi, and M. Kasahara, "Cryptosystems based on pairing," in *Symposium on Cryptography and Information Security (SCIS2000)*, Okinawa, Japan, Jan. 2000, pp. 26–28.

[29] P. S. L. M. Barreto, B. Libert, N. McCullagh, and J.-J. Quisquater, "Efficient and provably-secure identity-based signatures and signcryption from bilinear maps," in *Advances in Cryptology - ASIACRYPT*, ser. Lecture Notes in Computer Science, vol. 3788. Springer, 2005, pp. 515–532.

[30] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Wormhole attacks in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 370–380, Feb. 2006.

[31] J. R. Douceur, "The sybil attack." in *IPTPS*, 2002, pp. 251–260.

[32] J. Newsome, E. Shi, D. Song, and A. Perrig, "The sybil attack in sensor networks: analysis & defenses," in *IPSN '04: Proceedings of the third international symposium on Information processing in sensor networks*. New York, NY, USA: ACM Press, 2004, pp. 259–268.

[33] "Multiprecision Integer and Rational Arithmetic C/C++ Library (MIRACL)," M. Scott. [Online]. Available: http://ftp.computing.dcu.ie/pub/crypto/miracl.zip

[34] "ARM922T," ARM. [Online]. Available: http://www.arm.com

[35] S. Roundy, D. Steingart, L. Frechette, P. K. Wright, and J. M. Rabaey, "Power sources for wireless sensor networks." in *EWSN*, 2004, pp. 1–17.

[36] E. D. Mastrovito, "VLSI architectures for computation in Galois fields," Ph.D. dissertation, Linköping University, Linköping, Sweden, 1989.

[37] A. Karatsuba and Y. Ofman, "Multiplication of many-digital numbers by automatic computers," *Translation in Physics-Doklady*, vol. 7, pp. 595–596, 1963.

[38] K. Fong, D. Hankerson, J. Lopez, and A. Menezes, "Field inversion and point halving revisited," *IEEE Transactions on Computers*, vol. 53, no. 8, pp. 1047–1059, Aug. 2004.

[39] R. J. McEliece, *Finite fields for computer scientists and engineers*. Kluwer Academic Publishers, 1987.

[40] T. Itoh and S. Tsujii, "A fast algorithm for computing multiplicative inverses in $GF(2^m)$ using normal bases," *Inf. Comput.*, vol. 78, no. 3, pp. 171–177, 1988.

[41] J. Guajardo and C. Paar, "Itoh-Tsujii inversion in standard basis and its application in cryptography and codes," *Des. Codes Cryptography*, vol. 25, no. 2, pp. 207–216, 2002.

[42] *TSMC 65nm Technology Platform*, Taiwan Semiconductor Manufacturing Company. [Online]. Available: http://www.tsmc.com

[43] "Synopsys," Synopsys. [Online]. Available: http://www.synopsys.com

[44] C. Shu, K. Gaj, and S. Kwon, "FPGA accelerated tate pairing based cryptosystems over binary fields," in *Field Programmable Technology, 2006. FPT 2006. IEEE International Conference on*, Bangkok, Dec. 2006, pp. 173–180.

[45] T. Kerins, C. Murphy, C. O hEigeartaigh, R. Ronan, and M. Scott, "FPGA acceleration of the tate pairing in characteristic 2," in *IEEE International Conference on Field Programmable Technology*, Bangkok, Dec. 2006, pp. 213–220.

[46] M. Keller, T. Kerins, F. Crowe, and W. Marnane, "FPGA implementation of a $GF(2^m)$ tate pairing architecture," in *International Workshop on Applied Reconfigurable Computing - ARC*, ser. Lecture Notes in Computer Science, vol. 3985. Springer, 2006, pp. 358–369.

[47] M. Keller, R. Ronan, W. Marnane, and C. Murphy, "A $GF(2^{4m})$ inverter and its application in a reconfigurable tate pairing processor," in *IEEE International Conference on Reconfigurable Computing and FPGA's*, San Luis Potosi, Mexico, Sep. 2006, pp. 1–10.

Kealan McCusker received the B.Sc. degree from the University of Manchester, Manchester, U.K. in 1994, the M.Sc. degree in Electronic Engineering from Queen's University, Belfast, U.K. in 1996 and the PhD degree from Dublin City University, Dublin, Ireland in 2008. From 1997 until 2003 he worked as an ASIC design engineer in industry. He is currently employed as a postdoctoral researcher in CLARITY: Centre for Sensor Web Technologies, Dublin City University, Dublin, Ireland. His research interests are in the field of identity based cryptography, with application to wireless sensor networks, and object recognition in computer vision.

Noel E. O'Connor Noel E. O Connor received his PhD in Dublin City University, Ireland in 1998 for work focusing on object detection and tracking in video sequences for compression applications. He is currently an Associate Professor in the School of Electronic Engineering in Dublin City University and a Principal Investigator in CLARITY: Centre for Sensor Web Technologies, with responsibility for the research strand on Contextual Content Analysis. The focus of his current research is in multi-modal content analysis leveraging mutually complementary sensor data sources, for applications in sports, ambient assisted living, digital media, gaming and environmental monitoring.