Practical Identity-Based Key Agreement For Secure Communication in Sensor Networks

Piotr Szczechowiak
School of Electronic Engineering
Dublin City University
Glasnevin, Dublin 9, Ireland
Email: piotr@eeng.dcu.ie

Martin Collier
School of Electronic Engineering
Dublin City University
Glasnevin, Dublin 9, Ireland
Email: collierm@eeng.dcu.ie

***Abstract*—Despite much research effort key distribution in Wireless Sensor Networks (WSNs) still remains an open problem. In this paper we address this issue by proposing a simple identity-based key agreement scheme. Our protocol uses Identity-Based Cryptography (IBC) and secret key pre-distribution. We argue that IBC is in many ways a perfect solution for WSNs. It reduces the number of required keys, simplifying the key management in the network, and has a lower communications overhead than traditional public key protocols.**

We evaluate our proposal on a broad range of sensor platforms to show its efficiency on different CPU architectures. A complete key agreement procedure takes less than 3s on a resource-constrained Tmote Sky node without requiring any communication between two parties. To our knowledge this work is the first practical implementation of a complete IBC scheme on sensor devices.

We identify a range of WSN applications which would benefit from the incorporation of a security architecture, and show that the scheme described here makes it feasible to deploy these applications in the real world.

I. Introduction

A Wireless Sensor Network (WSN) can be considered as one of the most constrained pervasive systems with minimal resources. One of the main challenges in this research area is to provide reliable security services that overcome the limitations in terms of available power, computational capabilities and storage resources. A typical WSN node has an 8-bit micro-controller, 4KB of RAM, 128KB of program space, and is battery operated. Providing security in a distributed network comprising of such devices is a challenging task that needs new cryptographic solutions.

Sensor networks are often deployed over wide areas that in many cases are open to the public. This environment poses a threat of physical node capture. Typical low-cost devices will not have secure storage for cryptographic keys or tamper-proof hardware. An active attacker may easily subvert a node, intercept messages and decode them using the derived secret key.

WSNs also pose additional challenges that need to be addressed in the proposed security solution. One is the limited bandwidth available for communication. Devices are usually low-powered with a short battery life span. Wireless transmission is very expensive in terms of energy usage. Current radio transceivers use as much as ten times more energy than the node's CPU [14]. Hence the cryptographic scheme should minimize the communication overhead needed to provide security.

Additionally networks may consist of a large number of nodes which requires a scalable security protocol. The number of necessary cryptographic keys must be small due to very limited memory resources and to facilitate the scalability of the security scheme. For the same reason the key length should be short, but on the other hand long enough to provide a sufficient level of security.

Wireless communication between sensor nodes is insecure by its nature and requires cryptographic methods to ensure data confidentiality and message integrity. WSNs need also entity authentication and access to the network resources should be restricted allowing only genuine nodes to participate in data exchange. In order to satisfy all of these security goals we need to use Public Key Cryptography (PKC) methods for providing security.

Typical security solutions for WSNs use basic symmetric key algorithms [10] due to their simplicity and efficiency in resources utilisation. The exclusive usage of fast and energy efficient symmetric cryptography primitives may seem an obvious choice for constrained sensor devices, but in reality it does not solve the security problem. In typical real-life WSN deployments a network-wide key is used (e.g. in the residential mode in ZigBee WSNs [24]), which can be easily compromised. Such a security system is insufficient for many WSN applications, especially those that deal with critical data. Our ID-based key agreement scheme addresses the above issues and solves the key distribution problem in sensor networks.

II. Key distribution problem

The key distribution and management problem is one of the main security issues in sensor networks. In general there are three classes of key agreement schemes: trusted server mechanisms, public-key algorithms and key predistribution schemes. The first type uses central servers to issue certificates and to setup the whole public key infrastructure. This method is certainly far too complex for constrained systems with low power and low computing capabilities. Public key techniques use asymmetric cryptography which is more heavyweight than

978-1-4244-4581-3/09/$25.00 ©2009 IEEE

TABLE I
IDENTITY BASED CRYPTOGRAPHY VS OTHER SECURITY SCHEMES FOR WSNS.

	Symmetric key cryptography	**Public key cryptography**	**Identity based cryptography**
Computational complexity	Low	High	High
Communication overhead	Low	High	Low
Key distribution	Problematic	Complex	Simple
Number of keys	$O(n^2)$	O(n)	n
Key directory	At each node	At each node or key center	No
Non-repudiation	No	Yes	Yes
Forward encryption	No	No	Yes

symmetric algorithms and requires authentication of public keys. The third approach to key establishment is via predistribution, where nodes are loaded with keying material before their deployment [6]. Those schemes are mainly based on random key distribution - neighbouring nodes share a common key in terms of probability. This is a major drawback of such mechanisms, because we do not have a guarantee that a perfect connectivity between communicating parties can be established. This approach is also impractical for large scale sensor networks.

A. Identity-Based Cryptography

In this paper we propose a practical identity-based key distribution mechanism which does not require interaction between nodes to agree upon session keys. We use Identity-Based Cryptography (IBC) [19] and secret key predistribution to solve the key distribution problem. IBC is a public key technique that uses identities as public keys. Nodes are uniquely identified (by means of network or physical addresses) and this information is used to exchange keys and encrypt data to secure communication in the network. This allows the implementation of a practical public key encryption without the use of a complex public key infrastructure.

In many ways an identity-based scheme is a perfect solution for sensor networks. There is no need to maintain a public key directory, as the public keys can be derived from node's identities that are widely known in the network. Nodes generate a public key for a given node only in case when they want to communicate with it for the first time. After agreeing upon a shared session key, nodes can use cheap symmetric key mechanisms (like TinySec [10]) to encrypt the messages and communicate in a secure manner.

IBC has clear advantages over traditional PKC systems, but also has some inherent problems. Identity-based schemes assume the existence of a trusted authority that issues users' secret keys. This authority is often called the Private Key Generator (PKG) and can use its master key to decrypt any user message. It also has the ability to impersonate anybody in the system. This feature introduces the key escrow problem, where the security of the whole system depends on the PKG security. In many cases a single unconditionally trusted entity in the network simply does not exist.

Fortunately in sensor networks the original network deployer is obviously a trusted authority that can play the role of PKG. It can generate a unique secret key based on each node's identity and preload this information to the node's memory before the deployment phase. At this stage a secure channel clearly exists which allows careful configuration of the network. We can assign identities to nodes and load all the required public parameters.

Another problem associated with IBC is key revocation. When a private key is compromised the owner should change his identity information corresponding to that private key. This might be especially problematic in case where identities are based on the node's unique physical address (e.g. the transceiver serial number). That is why we propose to use network addresses (e.g. TinyOS or IPv6 addresses) for our identity based cryptosystem. This allows the PKG to assign new identities to nodes and generate appropriate private keys to replace the compromised ones.

B. IBC vs other security schemes

Table I summarizes the main advantages of IBC when compared with other security schemes. It also shows our main motivations behind choosing an ID-based mechanism to ensure secure communication in sensor networks.

Symmetric key system is obviously the most lightweight solution, but it does not address the key distribution problem and it cannot provide all the required security services. In addition it requires large number of pair-wise keys to achieve perfect connectivity. Symmetric key cryptography scheme in the ZigBee commercial mode needs $O(n^2)$ master keys in a sensor network of n devices [24]. The addition of new nodes in such a system is also cumbersome, as it requires new keying material to be added to each existing node. Identity based scheme clearly decreases the number of necessary keys in the network and provides a scalable solution that reduces the storage overhead.

In our IBC scheme we use a unique identity as a public key rather than relying on certificates and revocation lists, as used in traditional PKC systems. We do not have to store so many public keys and the communication overhead related to key agreement is minimized. We can also send encrypted information to nodes that have not received their secret keys yet (forward encryption). With identity based cryptography the whole key distribution is simplified and easier to manage. IBC allows us to provide all the security services of a traditional public key system in a more elegant way and at lower price.

IBC seems to fit perfectly as a solution for the key distribution problem in WSNs. However before we proceed we need to ask a basic question. Is such a system even viable on constrained sensor devices? And if so, can we achieve a

sufficient level of security? There is a clear correlation between the level of security achieved, and the processing power required. We have to set the security parameters of the system beyond the current cryptanalysis records and demonstrate that implementation at acceptable speeds is possible on sensor nodes.

The evaluation results presented in section V show that our key agreement mechanism is suitable even for the most constrained sensor nodes like the 8-bit MICAz. This makes our scheme practical for many different WSN applications, especially those that are more demanding in terms of security (see section VI).

III. Related work

Research results obtained in recent years reject the popular belief that PKC is infeasible for sensor nodes by showing that public key primitives can be implemented on embedded devices [22], [8]. These results show that Elliptic Curve Cryptography (ECC) is a far more efficient PKC method than RSA in a resource constrained environment.

Although we can apply PKC in WSNs through the use of ECC primitives, we still need to distribute secret keys and somehow agree upon mutual keys between pairs of nodes in the network. We can use the Elliptic Curve Diffie-Hellman key exchange protocol (ECDH) [9], but it involves interaction between the nodes which consumes precious energy. Moreover this key exchange is not authenticated and can be subjected to a problematic man-in-the middle attack. It is clear that such a scheme is not the best security solution for sensor networks.

In the literature there are papers that propose the use of identities to distribute keys in sensor networks. The authors in [5], [23], [16] envisioned the use of Identity-Based Encryption (IBE) as a security solution for sensor networks. All three papers proposed the Boneh and Franklin identity-based encryption scheme [3] to distribute keys in the network. IBE is based on cryptographic pairings and it was the first practical IBC scheme.

In [23] the IBE scheme was evaluated through a simulation on a desktop class computer. The relevance of the achieved results to WSNs is not clear, as simulation details were not presented. Doyle *et al.* in [5] showed the energy consumption results for the Tate pairing calculation on the ARM7 processor. This platform, however, is considerably more powerful than the devices that are currently in use in WSNs. Finally Oliveira *et al.* [16], [15] proposed the IBE scheme for sensor networks and presented the Tate pairing implementation figures for the MICAz mote. The pairing calculation on its own took 31s making the whole IBE scheme infeasible for practical sensor network applications.

In this paper we propose a simple identity-based key agreement that is not as complex as Boneh and Franklin IBE and does not require interaction between parties to agree upon mutual keys. Our scheme uses the η_T pairing and takes around 4.3s to complete on a MICAz WSN platform. To our knowledge this work is the first practical implementation of a complete IBC scheme on sensor devices.

IV. Identity based key agreement

Our key agreement scheme for WSNs is based on the protocol proposed by Sakai, Ohgishi and Kasahara in [17]. We also use secret key pre-distribution using the secure channel that exists during the network pre-deployment phase. The whole scheme is based on the concept of a cryptographic pairing on elliptic curves. Before we can describe the key agreement procedure we have to introduce basic definitions related to pairings.

A. Concepts

A pairing is a bilinear map between two groups. The Tate or η_T pairings on elliptic curves are examples of such a map. A bilinear pairing can be defined as follows. Let ℓ be a positive integer. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be additively-written groups of order ℓ with identity $\mathcal{O}$, and let $\mathbb{G}_T$ be a multiplicatively-written group of order ℓ with identity 1. A bilinear pairing is a computable, non-degenerate function

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T.$$

The most interesting property of a cryptographic pairing is bilinearity. We can say that a map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is bilinear if

$$\hat{e}(aP, bQ) = \hat{e}(P, bQ)^a = \hat{e}(aP, Q)^b = \hat{e}(P, Q)^{ab}$$

for all $P \in \mathbb{G}_1$, $Q \in \mathbb{G}_2$ and all $a, b \in \mathbb{Z}_q$. Pairings that are evaluated over supersingular elliptic curves have additional property of symmetry $\hat{e}(P, Q) = \hat{e}(Q, P)$.

The groups $\mathbb{G}_1$ and $\mathbb{G}_2$ are implemented using a group of points on certain special elliptic curves (pairing-friendly curves) and the group $\mathbb{G}_T$ is implemented using a multiplicative subgroup of an extension of the underlying finite field. For certain families of supersingular elliptic curves we have $\mathbb{G}_1 = \mathbb{G}_2$.

Every cryptographic construction must rely on some hard problem based on number theory to be secure. Most of the pairing applications rely on the hardness of the following problem for their security [7]: given P, aP, bP, and cP for some $a, b, c \in \mathbb{Z}_q$, compute $\hat{e}(P, P)^{abc}$.

This problem is known as the Bilinear Diffie-Hellman Problem (BDHP). The hardness of the BDHP depends on the hardness of the Diffie-Hellman problems both on $E(\mathbb{F}_q)$ and in $\mathbb{F}_{q^k}$. $E(\mathbb{F}_q)$ is an elliptic curve defined over a finite field $\mathbb{F}_q$, where q is a prime power and $\mathbb{F}_{q^k}$ is a k-th extension field.

B. Pre-deployment phase

The original network deployer takes the role of a trusted authority in our system. He needs to load secret keys into each node's memory together with all the public parameters. Initially he generates the master key s which has to be kept secret. He also assigns a unique identity to each node that will participate in the network. For this purpose he can use TinyOS or IP addresses (in the case of an IPv6 addressing scheme).

In the next step the PKG calculates each node's private key. This operation can be performed by the use of ECC primitives.

The publicly available hash function H is used to derive a hash value based on the node's identity $N_X = H(ID_X)$. The N_X value is mapped to an elliptic curve point via a mapping function. This allows us to perform the point multiplication operation with the use of the master key as a scalar, namely sN_X. The result of this operation is a node's private key. From the science of elliptic curve cryptography we know that it is not feasible to derive the s value based on sN_X, when the size of N_X in bits is bigger than 160, as this would require the solution of an intractable discrete logarithm problem[1].

Each node is issued with a single secret key sN_X, its identity ID_X, hashing function H, mapping function and a Key Derivation Function (KDF) that is based on a one-way hash function. KDF is required to derive the session key of a size that is suitable for a particular symmetric cipher. All these parameters are preloaded into each node's memory before the deployment phase.

C. Key agreement in sensor networks

After the setup process nodes are ready for deployment. During network operation two nodes **A** and **B** that know each other's identities can exchange information in a secure way without any prior interaction. Node **A** has a private key sA and node **B** a key sB. Both sides can independently obtain the required public keys A and B by calculating $A = H(ID_A)$ and $B = H(ID_B)$.

When **A** wants to setup a pair-wise session key $K_{A,B}$ with **B** he calculates the bilinear pairing function $\hat{e}(sA, B)$. The KDF function is used to derive the session key from the calculated pairing value $K_{A,B} = KDF(\hat{e}(sA, B))$. Now **A** can use the key $K_{A,B}$ to encrypt the message and send it over the radio channel to **B**. Node **B** receives the message and obtains the decryption key $K_{B,A} = KDF(\hat{e}(sB, A))$ to read the packet payload. Both **A** and **B** will end up with the same key since

$$K_{A,B} = KDF(\hat{e}(sA, B)) = KDF(\hat{e}(A, B)^s)$$
$$= KDF(\hat{e}(A, sB)) = KDF(\hat{e}(sB, A)) = K_{B,A}.$$

This follows from the bilinearity and symmetry properties of our pairing function. Our protocol allows two nodes to agree upon a common pair-wise key without any prior interaction with each other. There is no extra bandwidth overhead associated with the cryptography, and subverting one node does not reveal anything about communication between other pairs of nodes. In case an attacker steals an identity of a node or takes fake identities he still cannot establish a shared key, because he does not have the PKG's master key s and is not issued with a private key sN_X. Our scheme guarantees that the master secret s is not revealed even if all the nodes are subverted.

The above key agreement scheme is a simple way to bootstrap security in a sensor network. However we are aware that this mechanism on its own is not sufficient to secure the network against any given attack. Our scheme might not be also appropriate for all kinds of WSN applications (see section VI for suitable applications). The main purpose of this paper is to propose a secure and practical key agreement mechanism for sensor networks that will be a foundation for other network protocols built on top of it.

[1]this level of security in ECC system is equivalent to 80 bits of security for symmetric key algorithms [9].

V. Performance evaluation

Our identity based key agreement mechanism is based on the mathematical theory of elliptic curves which involves operations that are computationally intensive. The implementation of the whole scheme is quite difficult and requires cryptographic primitives used in ECC and Pairing-Based Cryptography (PBC). The key distribution is performed mainly at the beginning of network operation to establish session keys with neighbouring nodes, and thus the expensive pairing calculations are very infrequent. We also expect that the network will use an energy efficient cluster-based routing protocol. In this case majority of nodes will have to perform the key agreement only once, when establishing a common key with their cluster heads.

A. Security parameters

In ECC systems it is not enough to rely only on large key sizes to provide a high level of security. Other domain parameters like the selected curve E and an appropriate finite field $\mathbb{F}_q$ are equally important. All these parameters should be chosen so that the Elliptic Curve Discrete Logarithm Problem (ECDLP) is resistant to all known attacks. In the case of the binary field $\mathbb{F}_{2^m}$ we should choose our pairing parameters as $k \cdot m > 1024$ [12], where k stands for the embedding degree. These countermeasures would make any index calculus attack [9] on the BDHP infeasible for the time being. According to the above security policy we chose a supersingular elliptic curve $y^2 + y = x^3 + x$ over the binary field $\mathbb{F}_{2^{271}}$ with the embedding degree $k = 4$.

B. Implementation

In the beginning we have to perform the setup process and pre-load all the necessary information to sensor nodes. In our evaluation we do not consider the operations performed by the trusted authority in the pre-deployment phase. This is done off-line, and so is not time-critical, and it does not have any influence on the network performance. In what follows we focus only on the operations that are carried out by the sensor nodes during the key agreement scheme.

The overall performance of the identity-based key distribution mechanism relies on the efficiency of the pairing calculation $\hat{e}(P, Q)$. The η_T pairing [2] is one of fastest known pairings that can be evaluated very efficiently. It uses a variant of Miller's algorithm to calculate the pairing. One of its advantages is that it requires only half the number of iterations of the Miller's loop compared with typical pairing algorithms. That is why we chose the η_T algorithm for all our implementations. Due to space constraints we cannot present all the details regarding our pairing implementations. In depth information can be found in a recent paper [20].

The η_T pairing evaluates as an element in the $\mathbb{F}_{2^{4*271}}$ extension field. As a result we get a 1084-bit value that can be used to calculate the mutual session key. The KDF function derives the appropriate key by hashing the pairing result. In our implementation we used 128-bit symmetric keys suitable for the AES block cipher.

To evaluate the efficiency of our scheme we need also to consider the cost of mapping node identities to elliptic curve points. One viable method is to hash the identity to the x-coordinate, and then solve a quadratic equation to find y. We use the following fast algorithm to solve the $y^2 + y = c$ equation (based on [9]).

Algorithm 1 Solve $y^2 + y = c$ (basic version)

INPUT: $c = \sum_{i=0}^{m-1} c_i z^i \in \mathbb{F}_{2^m}$ where m is odd, trace $Tr(c) = 0$ and $H(z^i)$ is a half-trace function
OUTPUT: A solution s of $y^2 + y = c$
1: Precompute $H(z^i)$ for odd i, $1 \leq i \leq m-2$
2: $s \leftarrow 0$
3: **for** $i \leftarrow (m-1)/2$ **to** 1 **do**
4: **if** $c_{2i} = 1$ **then do**: $c \leftarrow c + z^i, s \leftarrow s + z^i$
5: **end for**
6: $s \leftarrow s + \sum_{i=1}^{(m-1/2)} c_{2i-1} H(z^{2i-1})$
7: **return** s

When the quadratic equation has no solution we have to increment x and repeat the process described in Algorithm 1. Otherwise we pick one of the two solutions for y and multiply the elliptic curve point (x, y) by the large cofactor, to obtain a point of an appropriate order. This point multiplication is the most expensive operation in the whole ID mapping scheme. The performance of ID mapping depends also on the efficiency of the one-way hash function.

In order to check the performance of our security scheme, we implemented it on some typical WSN sensor nodes. We chose three popular hardware platforms used in real-life deployments: the MICAz [4] and Imote2 [1] platforms developed by Crossbow Technology, and the Tmote Sky [14] developed by Moteiv corporation. The MICAz device is build upon the 8-bit ATmega128L processor, whereas the Tmote Sky uses the MSP430F1611 microcontroller. Imote2 is a far more powerful sensor node with a 32-bit Marvel PXA271 CPU.

To implement pairings we used the MIRACL [18] library which provides all the necessary tools to perform operations on elliptic curves. MIRACL was designed mainly for desktop class computers, and we needed to optimize it for our constrained 8, 16, and 32-bit platforms. All memory allocation in our programs was taken directly from the stack. This means that after the pairing calculation almost all of the RAM memory could be re-used for different purposes. As a one-way hash function we used the popular SHA-1 algorithm. An efficient implementation of this function for different sensor devices can be found in the TinyECC [13] package.

C. Results

Table II presents the evaluation results for all the basic operations that are performed by a single node in our key agreement scheme. Numbers in brackets tells us how many times a given routine needs to be performed. As we can see the pairing calculation is the most expensive operation in terms of time consumption and memory utilization on all three platforms. ID mapping also takes considerable amount of time to complete.

The fact that we were working with a fixed field size ($\mathbb{F}_{2^{271}}$) allowed us to greatly optimize our code. We used mainly C code and some assembly language to speed up our time critical arithmetic routines (in particular binary polynomial multiplication). On our most constrained 8-bit MICAz device we were able to compute the pairing in 2.66s – a time comparable with the scalar point multiplication operation presented in [21]. The whole key agreement takes as little as 0.1s on Imote2 and 4.27s in the worst case on the most constrained mote. All the results presented in Table II assume 7.3828MHz clock rate on MICAz, 8.192MHz on Tmote Sky and two different default CPU frequencies on Imote2. The timings for the pairing calculation are acceptable, given that these operations are performed very rarely and mainly at the beginning of network operation.

One of the most important issues for sensor devices is efficient memory utilization. The pairing calculation takes a significant amount of ROM on all platforms. This is mainly due to the large size of the ECC library (more than 40KB on MICAz, around 20KB on Tmote Sky and 25KB on Imote2). The code needed to perform point multiplication for ID mapping is also included in this library. Currently we are working on different ways to optimize the MIRACL library in order to further decrease its memory footprint. The SHA-1 algorithm takes less than 4KB of ROM on all platforms.

We have also measured the energy consumption of our basic cryptographic operations. An experimental setup was used for the MICAz and the Tmote Sky hardware platforms to measure the current drawn from the batteries during program execution. As we can see in Table II, the Tmote Sky node is far more efficient in terms of energy consumption than the MICAz platform. Imote2 uses even less energy especially when we set its CPU to 104MHz. Assuming two new AA alkaline batteries with 2850mAh capacity we can perform around 0.6 million and more than 2 million of key agreement operations for the MICAz and the Tmote Sky nodes respectively.

VI. PRACTICAL APPLICATIONS

There are many practical applications where sensor devices can control the operation of critical equipment, monitor assembly lines and perform condition based monitoring of critical structures. For example, sensor devices are deployed on the Golden Gate Bridge in San Francisco to monitor structure vibrations [11]. The importance of security in such an application justifies the use of a PKC technique to secure the sensor network. Our ID-based security scheme would facilitate the deployment of such systems on a larger scale.

TABLE II
PERFORMANCE EVALUATION OF OUR ID-BASED KEY AGREEMENT ON THREE WSN HARDWARE PLATFORMS.

	8-bit MICAz			16-bit Tmote Sky			32-bit Imote2 (13MHz)			32-bit Imote2 (104MHz)		
	Time	ROM	Energy	Time	ROM	Energy	Time	ROM	Energy	Time	ROM	Energy
Pairing(x1)	2.66s	47.41KB	62.73mJ	1.71s	23.66KB	17.70mJ	0.46s	29.55KB	12.12mJ*	0.06s	29.55KB	3.76mJ*
ID mapping(x1)	1.55s	0.72KB	36.55mJ	1.07s	0.48KB	11.07mJ	0.28s	0.66KB	7.38mJ*	0.03s	0.66KB	2.47mJ*
Hashing(x4)	15ms	3.68KB	0.35mJ	11ms	2.44KB	0.11mJ	1.4ms	2.81KB	37μJ*	0.17ms	2.81KB	10μJ*
Total	4.27s	51.81KB	99.63mJ	2.82s	26.58KB	29.21mJ	0.75s	33.02KB	19.65mJ*	0.1s	33.02KB	6.27mJ*

* - based on manufacturer data

We can identify a whole range of commercial applications that are deployed in public areas, where the threat of a physical attack is larger than usual. This application range include water quality monitoring, tunnel lighting control, street traffic and parking monitoring. In our security scheme each node employs asymmetric cryptography primitives. Thanks to that the effect of a node being compromised is strictly local and does not affect the communication between other nodes in the network.

In many cases commercial buildings waste vast amounts of energy by inefficient Heating, Ventilation and Air Conditioning (HVAC) usage. WSNs can provide intelligent control based on precise real-time measurements to reduce the energy consumption of those systems. Sensor networks can also provide energy monitoring and automatic meter readings in our homes. All such systems require reliable security solutions that will allow wider deployments in the real world. Our authenticated key agreement can establish secure network connectivity and may help to achieve appropriate security levels for those applications.

There are also other important applications in the military and health-care spheres, where security has the highest priority. Such systems must depend on public-key technology in order to fulfil all the security requirements. Our ID-based key agreement is a scalable PKC solution for dynamic sensor networks that sets the stage for an array of new and innovative applications.

VII. CONCLUSION

The application of ID-based cryptosystems to WSNs has a very promising future. This direction is especially interesting as more powerful nodes are being developed that allow more complex security protocols. In this paper we have shown that appropriately designed ID-based scheme might be a perfect solution for the key distribution problem in sensor networks. We have also identified particular WSN applications that may especially benefit from this security architecture. Our evaluation results have shown that this key agreement is feasible and practical on different WSN platforms and can be evaluated in around 4.3s on a tiny 8-bit sensor device.

In our future work, we plan to investigate techniques that can further speed up the execution and reduce the memory consumption for our scheme. We will also explore the possibilities of providing broadcast authentication in WSNs.

ACKNOWLEDGMENT

The authors would like to thank Mike Scott for his help and support in making this work possible.

REFERENCES

[1] Imote2 datasheet, 2008. http://www.xbow.com.
[2] P. S. L. M. Barreto, S. Galbraith, C. O'hEigeartaigh, and M. Scott. Efficient pairing computation on supersingular abelian varieties. *Designs, Codes and Cryptography*, 42:239–271, 2007.
[3] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003.
[4] Crossbow Technology, Inc., 41 Daggett Dr., San Jose, CA 95134. *MPR/MIB Mote Hardware Users Manual*, December 2003.
[5] 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*, 49(4), 2006.
[6] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *CCS '02*. ACM, 2002.
[7] S. Galbraith. Pairings. In I. Blake, G. Seroussi, and N. Smart, editors, *Advances in Elliptic Curve Cryptography*, London Mathematical Society Lecture Notes, pages 183–213. Cambridge University Press, 2005.
[8] N. Gura, A. Patel, A. Wander, H. Eberle, and S. C. Shantz. Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs. In *Workshop on Cryptographic Hardware and Embedded Systems (CHES'04)*, 2004.
[9] D. Hankerson, A. Menezes, and S. Vanstone. *Guide to Elliptic Curve Cryptography*. Springer, 2003.
[10] C. Karlof, N. Sastry, and D. Wagner. Tinysec: A link layer security architecture for Wireless Sensor Networks. In *2nd ACM SensSys*, 2004.
[11] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon. Health monitoring of civil infrastructures using wireless sensor networks. In *IPSN '07*, New York, NY, USA, 2007. ACM.
[12] A. K. Lenstra. Unbelievable security. Matching AES security using public key systems. In *Advances in Cryptology – Asiacrypt 2001*, volume 2248, pages 67–86. Springer-Verlag, 2001.
[13] A. Liu, P. Kampanakis, and P. Ning. Tinyecc: Elliptic Curve Cryptography for sensor networks (ver. 1.0), February 2007. *http://discovery.csc.ncsu.edu/software/TinyECC/*.
[14] Moteiv. Tmote Sky datasheet, 2006. http://www.moteiv.com.
[15] L. Oliveira, D. Aranha, E. Morais, F. Daguano, J. Lopez, and R. Dahab. Tinytate: Computing the tate pairing in resource-constrained sensor nodes. In *NCA 2007*, 2007.
[16] L. B. Oliveira, R. Dahab, J. Lopez, F. Daguano, and A. A. F. Loureiro. Identity-based encryption for sensor networks. In *PERCOMW '07*, 2007.
[17] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing. The 2000 Symposium on Cryptography and Information Security, Okinawa, Japan, 2000.
[18] M. Scott. MIRACL – Multiprecision Integer and Rational Arithmetic C/C++ Library, 2007. *http://ftp.computing.dcu.ie/pub/crypto/miracl.zip*.
[19] A. Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 47–53, New York, NY, USA, 1985. Springer-Verlag New York, Inc.
[20] P. Szczechowiak, A. Kargl, M. Scott, and M. Collier. On the application of pairing based cryptography to wireless sensor networks. In *WiSec '09, Second ACM conference on Wireless Network Security*, 2009.
[21] P. Szczechowiak, L. Oliviera, M. Scott, M. Collier, and R. Dahab. NanoECC: Testing the limits of Elliptic Curve Cryptography in Sensor Networks. In *EWSN 2008*, volume 4913 of *LNCS*. Springer-Verlag.
[22] R. J. Watro, D. Kong, S. Cuti, C. Gardiner, C. Lynn, and P. Kruus. Tinypk: securing sensor networks with public key technology. In *2nd ACM Workshop on Security of ad hoc and Sensor Networks*, 2004.
[23] G. Yang, C. Rong, C. Veigner, J. Wang, and H. Cheng. Identity-based key agreement and encryption for wireless sensor networks. *IJCSNS*, 6(5B):182–189, 2006.
[24] ZigBee-Alliance. Zigbee specification 1.1, 2005. http://www.zigbee.org.