On Application of Host Identity Protocol in Wireless Sensor Networks

Andrey Khurri, Dmitriy Kuptsov, and Andrei Gurtov
Helsinki Institute for Information Technology and
Department of Computer Science and Engineering, Aalto University
{firstname.lastname}@hiit.fi

***Abstract*—Recent advances in development of low-cost wireless sensor platforms open up opportunities for novel wireless sensor network (WSN) applications. Likewise emerge security concerns of WSNs receiving closer attention of research community. Well known security threats in WSNs range from Denial-of-Service (DoS), Replay and Sybil attacks to those targeted at violating data integrity and confidentiality. Public-key cryptography (PKC) as a countermeasure to potential attacks, although originally treated infeasible for resource-constrained sensor nodes, has shown its eligibility for WSNs in the past few years. However, different security and performance requirements, energy consumption issues, as well as varying hardware capabilities of sensor motes pose a challenge of finding the most efficient security protocol for a particular WSN application and scenario. In this paper, we propose to use the Host Identity Protocol (HIP) as the main component for building network-layer security in WSNs. Combining PKC signatures to authenticate wireless nodes, a Diffie-Hellman key exchange to create a pairwise secret key, a puzzle mechanism to protect against DoS attacks and the IPsec protocol for optional encryption of sensitive application data, HIP provides a standardized solution to many security problems of WSNs. We discuss how HIP can strengthen security of WSNs, suggest possible alternatives to its heavy components in particular WSN applications and evaluate their computational and energy costs on a Linux-based *Imote2* wireless sensor platform.**

***Index Terms*—Public key cryptography, wireless sensor networks, lightweight security, energy consumption, Imote2**

I. Introduction

With proliferation of advanced wireless sensor platforms featuring multiple connectivity options, input/output interfaces, scalable processing units and large memory storage, the popularity of wireless sensor networks (WSNs) grows opening up space for new applications for civil and military use. Security concerns of WSNs emerge, too, and despite active research in this area, a number of issues remain open.

A variety of well known attacks in WSNs aiming to violate confidentiality and integrity of sensitive data, to replicate node identity and to cause a denial of service, call for reliable and efficient security measures. Numerous research initiatives have investigated the question how different means of cryptography can be exploited in communications of wireless sensor nodes. Public-key cryptography (PKC) well known for its advantages over, e.g., symmetric algorithms in terms of better key management and scalability has been for a long time thought of as an infeasible cryptography class due to its computational complexity for low-power sensor motes (e.g., [1]). However, substantial research that followed has disproved earlier claims and shown that PKC and especially Elliptic-Curve Cryptography (ECC) are sensible even for weak sensor devices [2], [3], [4].

Although advanced sensor platforms such as *Imote2* [5] have further diminished the issue of computational complexity of cryptographic operations, energy consumption in WSNs is still a big concern. In addition, as related work shows [2], [6], despite availability of perfect security mechanisms applying them standalone is in most cases insufficient as it leaves a possibility for attacks. Hence, for better security a WSN often requires a combination of different solutions.

In this paper, we propose to use the Host Identity Protocol (HIP) [7] as the main building block for network-layer security in WSNs. Standardized by the IETF, HIP uses public keys for identification of nodes, PKC signatures for their authentication, a Diffie-Hellman (DH) key exchange to generate a common session key for a pair of nodes, a puzzle mechanism for protection against Denial-of-Service (DoS) attacks and the IPsec protocol for optional encryption of application data. By integrating several mechanisms HIP solves many security problems of WSNs.

The contributions of this study are the following. First, we introduce HIP and show how it can strengthen security of WSNs. Second, we suggest how HIP can be adapted to particular WSN scenarios that do not tolerate heavy computations or simply do not require PKC. We describe several alternative security mechanisms and show how they can be incorporated in HIP. Third, we port an existing HIP implementation to the Linux-based Imote2 sensor platform [5] and measure computational and energy cost of individual protocol components, both in existing and modified HIP. Finally, we analyze requirements of three different WSN applications, identify necessary for them protocol features and compare their energy consumption.

The rest of the paper is organized as follows. In Section II we give a short background on HIP for unfamiliar readers. In Section III we survey related work on the use of cryptography in WSNs. Section IV shows how HIP can be applied in WSNs. In Section V we discuss several alternatives to existing HIP components able to make it more computation and energy efficient. Section VI contains performance evaluation of selected protocol components on the Imote2 platform and Section VII concludes the paper.

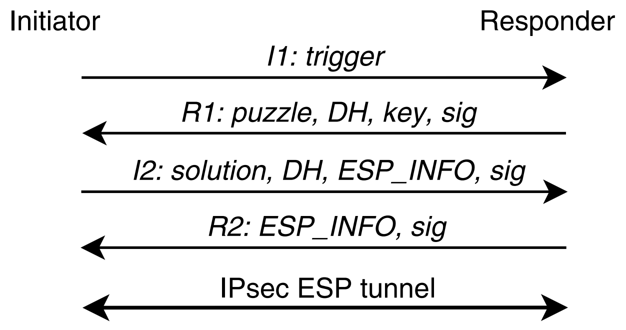


Fig. 1. HIP base exchange. Adapted from [8].

II. Background on Host Identity Protocol

The Host Identity Protocol [7], [8], [9] was proposed to overcome the problem of using IP addresses simultaneously for host identification and routing. The idea behind HIP is based on decoupling the network layer from the higher layers in the protocol stack architecture. HIP defines a new cryptographic *Host Identity* name space, thereby splitting the double meaning of IP addresses. In HIP, Host Identities (HI) are used instead of IP addresses in the transport protocol headers for establishing connections. IP addresses at the same time act purely as locators for routing packets towards destination. For compatibility with IPv6 legacy applications, each Host Identity is represented by a 128-bit long hash, the Host Identity Tag (HIT).

Prior to communication over HIP, two hosts must establish a HIP association. This process is known as the HIP base exchange (BEX) [8] and it consists of four messages transferred between the initiator (I) and the responder (R) (see Figure 1). A successful BEX authenticates hosts to each other and generates a DH shared secret key used in creation of two IPsec Encapsulated Security Payload (ESP) Security Associations (SAs), one for each direction. All subsequent traffic between communicating nodes is optionally encrypted by IPsec. Further details of the protocol can be found in RFC 5201 [8].

III. Related work

A number of projects have studied security issues in WSNs. An extensive survey of security requirements, threats and corresponding defensive measures is presented by Walters *et al.* [10]. Liu and Ning [1] claim infeasibility of using PKC for establishing a pairwise key in sensor networks and propose their own key predistribution scheme based on polynomials. We apply a similar scheme, as well as ideas from [11] to HIP. The same authors in another study [2] describe design, implementation and evaluation of the *TinyECC* library for applications running on TinyOS-based sensors. Besides the library code, the authors provide measurement results that clearly indicate performance of selected ECC primitives on four sensor platforms. However, as the authors note, without additional security mechanisms, TinyECC is susceptible to DoS attacks that can deplete sensors' batteries [2]. This conclusion exemplifies the need for a combination of techniques to achieve better security in WSNs.

Piotrowski *et al.* [3] provide an estimation for the lifetime of different sensor models running PKC primitives. After comparing RSA and ECC algorithms, the authors conclude that the use of ECC is feasible even on the weakest of the tested motes, MICA2DOT. Roman *et al.* [4] survey a wide combination of cryptographic primitives implemented in software and hardware and evaluate their applicability to different wireless sensor models, ranging from "weak" PIC12F675 (uPart) motes to "heavy-duty" ARM920T and PXA271 (Imote2) devices. The comparison of various primitives and implementations presented by the authors points to a number of potential optimizations that can improve suitability of cryptography for different WSN platforms.

Nachman *et al.* [12] present technical advances of the Imote2 platform over previous sensor models and introduce its prospective applications. By running a set of measurements of PKC while ranging the Imote2's CPU clock frequency, the authors draw an interesting conclusion that the advanced Imote2 device consumes less energy for an ECC key generation than its predecessor TelosB only when its CPU is clocked at high frequencies (above 208 MHz). Sun *et al.* [6] design an access control system for WSNs that consists of an ECC-based admission module (combined with a polynomial authentication), an Advanced Encryption Standard (AES)-based hardware security interface implemented in $CC2420$ radio components on TinyOS and a scheme for updating the network-wide symmetric key. Implemented on Imote2, the system includes the components similar to our prototype.

Comparing to related work, our work is different in a number of ways. First, while many studies assess performance of individual cryptography primitives for particular sensor platforms, we show how a standardized secure communication protocol can be applied in WSNs. Second, while acknowledging suitability of PKC for WSNs shown by previous research, we provide measurements of computational and energy cost of the protocol components performed on a real Imote2 sensor platform that is running Linux unlike many TinyOS-based devices. We emphasize that Linux has a great potential for future generations of sensors. Finally, we recognize the vast difference in hardware resources of sensor devices and WSN application requirements and accordingly do present and evaluate several lightweight security mechanisms that are able to enhance computational performance and prolong the lifetime of a WSN in particular scenarios, while providing sufficient security level.

IV. How a WSN can benefit from HIP

In this section we discuss what benefits HIP in its "standard" or existing form can provide for WSNs. Before discussing how certain protocol components would apply to WSNs, we explain the reason of selecting the Imote2 sensor platform for our experiments and describe a set of WSN scenarios where, in our opinion, an Imote2 running HIP can be particularly useful.

A. Scenarios with Imote2

The choice of Imote2 as the target hardware in our work was a result of the following observations. On the one hand, most of the research projects in the WSN field to date concentrated on extremely constrained TinyOS-based sensor motes with 8-bit processors (clocked at few MHz) and kilobytes of RAM. Unlike previous studies, we focused on the Imote2 sensor platform that, due to its advanced features, enables novel resource-demanding applications and represents a new generation of ”smart” sensors. Support of Linux adds potential value of porting existing protocols, libraries and applications. Platforms such as Imote2 provide high potential for medical, industrial and environmental monitoring applications [13], [14]. One of the Imote2's advantages over less powerful sensors is the ability to decrease the WSN overall energy and bandwidth consumption by processing as much data on board and transmitting as little data over radio as possible [12]. All this makes Imote2 a prospective candidate for future WSN deployments in a number of areas.

On the other hand, having a combination of scalable CPU, a MMX coprocessor, considerable RAM and storage space, small device form-factor, multiple I/O interfaces and connectivity options [5], Imote2 can become an essential architectural component of a multi-layered security framework for different WSNs. It can serve, e.g., as a sink or an aggregator of sensed data within a cluster of less capable nodes, performing data processing and secure forwarding to a remote site by employing strong cryptography. It can also act as an edge router in a medical patient area network (PAN), forming a secure gateway between the PAN and a remote service (e.g., a doctor). WSNs composed exclusively of powerful Imote2s can form reliable backbone networks.

Considering the above scenarios as primary ones for prospective use of Imote2, we assume that HIP can be used between Imote2 and a remote server or another communication device, between two Imote2 nodes (e.g., being cluster heads), and, in certain cases, between Imote2 and more-constrained sensor nodes (feasibility of PKC on tiny sensor nodes has been shown previously [2], [3], [4]).

B. Public key certificates

Most of the emerging commercial WSN applications will be used by closed groups with a set of different rights, thus requiring appropriate access control mechanisms. Each member of a WSN needs to prove its identity to all other members, which can be done using certificates. In case of HIP, the Host Identity (or the public key) of a node has to be signed by a certificate authority (CA) all other nodes trust. In addition to a CA, the public key infrastructure (PKI) of WSNs may include a repository that stores the list of untrusted/revoked public keys or identities. The selection of PKI components depends upon desired application requirements. For instance, in case of a medical WSN, we assume the presence of a central CA. If a patient's node does not trust in the certificate it received in a HIP BEX from a doctor, it may want to contact the CA for checking the state of the certificate in a certificate revocation list (CRL). In this way the sensor node can verify the validity of the doctor's certificate and check whether it was actually revoked before the expiration date or upon an event of compromising or stealing the device holding the public key [15]). Communication with the CA has to be itself secure, which can be achieved by establishing a separate HIP association between the sensor node (the patient in this case) and the CA (provided the CA is the entity trusted by each participant).

In case of an emergency situation, when the patient's node in the above example cannot contact the CA prior to communication with the doctor due to time limits, insufficient network coverage, the CA being offline or other constraints, the doctor (while approaching the patient) can request from the CA a token, which it can include into the BEX with the patient. The token would be a temporary certificate valid for a limited time period (as needed by the case), which will prove that the doctor's certificate was not revoked.

PKC certificates (e.g., X.509) can be added to HIP signaling traffic as described by Heer and Varjonen [16], enabling identity (public key) authorization and verification of the certificate status. However, to build a complete PKI, a sensor node will need additional signaling with a server storing CRLs. This will require protocol modifications, further details of which we leave out of the paper's scope. In case of a fully autonomous remote WSN with no central CA, CRLs have to be stored in a distributed manner. Additional cooperative security mechanisms should be applied in such scenarios to check the status of certificates and perform revocation of malicious or malfunctional nodes (see, e.g., [17]).

C. Authentication of sensor nodes

With the proven identity by means of PKC certificates, a sensor node will be able to authenticate itself to other WSN nodes, which is needed during the node admission phase. HIP uses PKC signatures for authentication of communicating parties. To improve efficiency of these operations and preserve energy for sensor nodes, “traditional” RSA (Rivest, Shamir and Adleman) or DSA (Digital Signature Algorithm) algorithms can be replaced with Elliptic Curve Cryptography (ECC) algorithms. We present performance of RSA, DSA and ECDSA (Elliptic Curve DSA) operations on the Imote2 sensor platform in Section VI.

D. Role management

Applications of WSNs may require separation of roles to cluster nodes in the network or delegate specific rights to certain sensor motes. For instance, there can be several intermediate sink nodes in a WSN that would have more computational and battery resources or a special hardware module to store sensitive information securely. Depending on a particular architecture, the delegation of rights and the separation of roles can be added to HIP again in a form of PKC certificates containing bindings between a sensor node identity and a subset of its roles and rights in this WSN.

E. Key exchange and secure channel establishment

To securely exchange sensed information between each other and with a sink or a central entity, sensor nodes should be able to establish a secure communication channel. For this purpose, HIP utilizes a DH key exchange protocol that generates a shared secret key for two nodes over an insecure channel. As will be seen in Section VI, a DH key exchange makes a major impact on duration of a HIP base exchange and, hence, it is wise to replace it with a much more efficient ECDH (Elliptic Curve Diffie-Hellman) exchange for constrained wireless sensor nodes.

The shared keying material resulted from a key exchange serves as an input to creation of IPsec ESP security associations used for optional network-layer encryption of sensitive application data. The applicability of IPsec in WSNs, however, has to be considered carefully as it adds extra bytes to the protocol header and increases energy cost per byte (we evaluated the impact of IPsec encryption on data throughput and energy consumption in our previous work [18]). Given the limited frame size of the IEEE 802.15.4 radio commonly used in WSNs (102-bytes on the medium access control layer [19]), as well as battery constraints of sensor nodes, we assume that the use of IPsec in WSNs will be limited, though not completely infeasible. One option is to utilize the keying material derived from a DH exchange for the link-layer AES-based encryption implemented in hardware on most IEEE 802.15.4 chips. Another option is to use the IPsec ESP mode with the AES CCM mode as suggested by Housley [20].

F. Protection against attacks

HIP contains a number of mechanisms that make it resistant to different attacks. However, certain protocol techniques provide no direct benefits to WSNs. For instance, the puzzle mechanism, that protects HIP responder from potential DoS attacks and allows to postpone creation of the state until receiving a valid I2 packet [8], would likely not be able to prevent a battery-exhausting DDoS (Distributed Denial-of-Service) attack, given the vast difference in resources of the attacker (e.g., a network of zombies) and the victim (i.e., a lightweight sensor node). To eliminate DoS and DDoS attacks, we suggest to use additional access control mechanisms (e.g., a HIT-based firewall) that would allow to filter out unwanted or unregistered clients already at an early stage prior to processing of HIP control packets. Limited number of legitimate clients is a fair assumption for the above-mentioned scenarios where an Imote2 acts as a cluster head or a gateway for less capable nodes. HIP can be used in such scenarios to authenticate the Imote2 and a remote host (an “operator”, a sink etc.) to each other and to set up a secure communication tunnel between them. However, provided “non-public“ nature of the network, the benefits of the puzzle mechanisms in such cases are doubtful.

Please note that we only imply attacks on the network layer that are a result of retransmissions or intentional flood of spoofed HIP packets. We do not consider other possible types of DoS attacks such as, e.g., jamming radio channel with a signal of interfering frequency. HIP packet protection against replay attacks is described thoroughly in its specification [8], please refer for details to it. Sybil attacks (or taking on multiple identities) are not feasible in HIP scenarios as long as HITs (public keys) of communicating hosts are signed with the private key of a CA, which should be a prerequisite as stated earlier. Among other methods, the use of a tamper-proof device for storing private keys of WSN nodes can prevent attempts of their ”cloning”. In general, we consider the ”standard” HIP to be useful in various WSN scenarios but particularly in those presented in the beginning of this section.

V. Lightweight alternatives for HIP

Depending on requirements, certain WSN applications either are not capable of performing computationally intensive PKC operations or just do not require PKC at all. In such cases, certain HIP components can be either omitted or replaced with lightweight alternatives, thereby extending the potential protocol use to more constrained sensors and applications that require fast communications and low energy consumption. We discuss two such alternatives in this section.

A. Lightweight certificates for HIP

Although public key certificates allow to fulfill the desired security requirements of WSNs in a scalable and a flexible way, in certain applications, such as a static unattended WSN or an on-body network communicating through a gateway, introducing a complex PKI does not bring additional benefits. Omitting a PKI allows to avoid cost of public key certificates, while still having authorization and authentication of host identities.

We use a notion of a *lightweight certificate* as defined in [11]. One of the representations is based on a Merkle tree [21], [22]. Such certification structure is a binary tree, where each of n leaves L_i is calculated as the hash of a value $HIT_i \| Role_i \| Time_{validity}$, and each internal node m_{ij} is calculated as the hash of the concatenation of its two sibling nodes $m_{ij} = H(m_i \| m_j)$. The root of the tree together with $log_2(n)$ elements can be used to verify any leave in the tree.

The above information from the binary tree combined with the corresponding path is sufficient to allow any node (which has the root element of the tree) to efficiently perform (i) authentication of an identity (a public key), (ii) authorization of any other node that holds the corresponding identity, and (iii) verification of any other additional information bound to the identity. Therefore, the advantage of lightweight certification is that it provides efficient authentication and authorization of an identity and allows to build access control mechanisms without heavy computations typical for PKC approach.

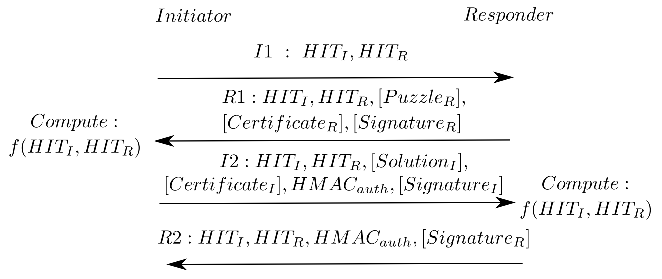
In practice, such lightweight certificates (generated similarly to PKC certificates by a trusted CA) can be easily integrated in HIP, as the approach described in [16] has no limitation on the type of certificates carried in $R1$, $I2$, or $R2$ signaling packets.

B. Lightweight key exchange for HIP

So far HIP was specified for using asymmetric cryptography only, which on the one hand provides scalability for key negotiation, but on the other hand requires more computational resources. As a less scalable but more lightweight alternative for particular types of WSNs, we suggest to use a polynomial key exchange presented in [1], [11].

The approach in question makes use of a preloaded on each sensor node set of bivariate polynomials $f(x, y)$, each of degree α over a finite field $GF(q)$, such that q is a large enough prime to accommodate a cryptographic key. The technique allows any pair of nodes to derive a secret key by evaluating the polynomial using corresponding identities as points. In case of a small WSN, e.g., a body network, each sensor node i can directly share a polynomial $K_0 = f(HIT_i, HIT_j)$ with any other node j in the network. Since the exchange uses preshared polynomials, it is sufficient to evaluate polynomials at points $HIT'_i = HASH(HIT_i \| Role_i \| Time_{validity})$, so that (i) nodes negotiate a secret key, and (ii) if a correct key was derived, authorization succeeds (without usage of a Merkle tree). The approach is similar to the one proposed in [11].

Storing all polynomials on each sensor node provides (i) a good secrecy, and (ii) ensures successful key negotiation. However, if a network comprises several thousands sensor nodes, a grid-based assignment [1] can be used to minimize memory footprint. This will guarantee (i) that any pair of sensors can agree on a common secret, and (ii) resilience to a node compromise when certain conditions are met. For further details please refer to [1]. A corresponding lightweight HIP base exchange with the use of polynomials as a way to derive a common secret is shown in Figure 2.



$Initiator$
$Responder$
$I1\ :\ HIT_I, HIT_R$
$R1 : HIT_I, HIT_R, [Puzzle_R],$
$[Certificate_R], [Signature_R]$
$Compute:$
$f(HIT_I, HIT_R)$
$I2 : HIT_I, HIT_R, [Solution_I],$
$[Certificate_I], HMAC_{auth}, [Signature_I]$
$Compute:$
$f(HIT_I, HIT_R)$
$R2 : HIT_I, HIT_R, HMAC_{auth}, [Signature_R]$

Fig. 2. Lightweight HIP base exchange.

During the BEX in Figure 2 both initiator and responder use a modified key derivation function (KDF) [8] shown in Equation (1). Here, a postfix is defined as $sort(HIT_I \| HIT_R) \| I \| J \| 0x01$, where I and J are nonce values, and *sort* is a function that sorts HITs as specified in [8]. Thus, to obtain the k^{th} key, nodes evaluate (1) k times starting from the initial key K_0.

$$K_k = HASH(HASH(K_{k-1}) \| postfix) \qquad (1)$$

Note that the parameters in square brackets in Figure 2 are optional meaning that (i) PKC signatures can be present if compatibility is required, (ii) puzzle parameters are only needed when there is a DoS threat or for negotiating random values I, J used in KDF, and (iii) lightweight certificates can be carried if the network needs to support a node authorization. The use of a polynomial key exchange in HIP represents a trade-off between performance and scalability for a WSN. While providing efficiency in computation of a shared key, the mechanism requires preloading a set of bivariate polynomials onto sensor nodes before their deployment, which makes adding new sensors to the network afterwards problematic, However, the approach is still valid for many WSN scenarios where the network growth is not part of the design.

VI. Performance evaluation

This section presents a performance evaluation of HIP and its alternatives applied to WSNs. Our analysis is based on measurements in a pilot sensor network, as well as related work. We first describe methodology for our experiments, then present measurement results and finally evaluate performance of a combination of security components in different WSN applications.

A. Hardware and methodology

In the following paragraphs we introduce the *Imote2* sensor platform served as a target hardware in our experiments and describe tools and utilities used to conduct particular tests.

1) Imote2 specifications: Technical specifications of Imote2 can be found from its product datasheet [5]. Comparing to other well known sensor models (e.g., TelosB, IRIS, MICAZ), Imote2 is an advanced platform that enables computation-intensive WSN applications. Dynamic voltage scaling allows to change frequency of the *Intel XScale* processor from 13 to 416 MHz and to rationalize power consumption of each application. Different sleep modes of the processor allow to save energy when applications are inactive [5].

2) Tools and methods: To measure computational cost of different cryptographic algorithms and operations that were of our interest, we used several utilities and methods including *OpenSSL speed* (version 1.0.0-beta3), a benchmarking program from the HIPL (HIP for Linux) code, as well as timestamps inside the protocol code.

To measure current draw (and then calculate energy cost of security primitives) on the Imote2 sensor platform we used a Fluke 189 digital multimeter with a DC (Direct Current) accuracy of $\pm$ $(0.15\% + 2)$. The multimeter was able to calculate and display an average DC draw on a minimum 1-second interval. For short events lasting for microseconds and milliseconds (e.g., RSA signature verification), we thus needed to do more repetitions in a row, so that a sufficient number of multimeter readings for a particular event can be collected and analyzed statistically. For convenience, all multimeter readings were transferred and stored on a PC using *FlukeView Forms* software.

B. Measurement results

We present only a subset of the most important measurement results including computational complexity of individual

TABLE I
DURATION OF SIGNATURE OPERATIONS ON IMOTE2.

		Average time (ms)					
Key length (bits)		**RSA**		**DSA**		**ECDSA**	
R(D)SA	ECC	*sign*	*verify*	*sign*	*verify*	sign	verify
512	**N/A**	4.1	0.3	4.2	3.8	N/A	N/A
1024	**160**	20.0	1.1	11.2	12.0	3.5	9.4
2048	**224**	127.1	3.9	38.2	43.9	6.7	26.4
3072	**256**	N/A	N/A	N/A	N/A	8.5	34.5

TABLE II
ENERGY CONSUMPTION ON A 416-MHZ IMOTE2. VOLTAGE 4 V.

OPERATION	CURRENT	DURATION	ENERGY
Idle, radio on	73 mA	N/A	N/A
Idle, radio off	52 mA	N/A	N/A
RSA-1024 create key	144 mA	2300 ms	1324.8 mJ
RSA-1024 sign	154 mA	20.0 ms	12.3 mJ
RSA-1024 verify	144 mA	1.1 ms	0.6 mJ
DSA-1024 create key	151 mA	12100 ms	7308.4 mJ
DSA-1024 sign	153 mA	11.2 ms	6.9 mJ
DSA-1024 verify	150 mA	12.0 ms	7.2 mJ
ECDSA-160 sign	141 mA	3.5 ms	2.0 mJ
ECDSA-160 verify	147 mA	9.4 ms	5.5 mJ
DH-1536 create	158 mA	349.5 ms	220.9 mJ
DH-1536 shared	158 mA	348.4 ms	220.2 mJ
ECDH-192 shared	147 mA	16.1 ms	9.5 mJ
SHA1 hash create	142 mA	0.03 ms	0.02 mJ
Polynomial $GF(2^{16}-1)$	150 mA	0.45 ms	0.27 mJ
Puzzle solving $K = 10$	142 mA	28.5 ms	16.2 mJ

cryptographic components and their energy cost for an Imote2 sensor node.

1) Computational complexity of HIP: To see how the total duration of the HIP base exchange is affected by its individual components we have performed a set of isolated measurements. The results allow to see what elements make the most critical impact in terms of computational complexity and energy cost to an Imote2 sensor platform. The components include asymmetric key generation, signature operations (RSA and DSA versus ECDSA), Diffie-Hellman key exchange (DH versus ECDH) and puzzle mechanism. The results of the measurements not presented separately are summarized in Table II.

a) Signatures and verifications: Table I shows computational complexity of operations on public-key signatures. It contains average times needed to generate and verify a signature on an Imote2 sensor platform using RSA, DSA and ECDSA keys of different length. The comparison of "traditional" asymmetric algorithms such as RSA and DSA with ECC is especially interesting since ECC achieves the same level of security with notably shorter keys [23], thus providing considerably better performance while saving storage space and network bandwidth.

Benefits of ECC are increasing with raising the level of security, i.e the key length (see Table I). It is important to point to the difference in performance of *sign* and *verify* operations for different algorithms. While verification of RSA signatures takes significantly less time than their generation, DSA sign and verify operations are of the same order of magnitude. ECDSA, in turn, has a different behavior with verification of signatures consuming more resources than their generation.

b) DH shared key computation: Diffie-Hellman (DH) key exchange is used to establish a shared secret key between two communicating parties. In HIP base exchange it accounts for a major delay. On an Imote2 with a default 1536-bit DH group it takes on the average 349.5 ms to create a DH public value and 348.4 ms to generate a shared secret key, resulting in the overall 697.9 ms-delay for a complete DH exchange. In contrast, a corresponding (in security level) ECDH exchange with a 192-bit ECDH group is accomplished on an Imote2 in 32.2 ms on the average. Our measurements show that with increasing a DH key length, the benefits of ECC are growing.

c) Processing of puzzle: Although we have found no particular WSN scenarios where the HIP puzzle mechanism would be worth using, we have measured performance of Imote2 in solving puzzles of various difficulties. The time to solve a cryptographic puzzle by an Imote2 rises exponentially from 20 ms with no puzzle to solve (difficulty $K = 0$) to 143.6 seconds with $K = 25$. The numbers are the median values of collected samples of puzzle processing times.

d) BEX total duration: The total duration of a HIP BEX between an Imote2 and a server (connected via a USB network with the average RTT=3.7 ms) in our tests took 1.1 seconds. The BEX between two Imote2 sensors communicating over IEEE 802.15.4 radio lasted 1.7 seconds with RTT varying from 13 to 127 ms on the average for packets ranging from 64 to 1028 bytes in size. The above BEX times are the medians of the corresponding samples of measured BEX durations. In a real WSN, the delay for an association establishment should not become a concern unless such exchanges are too frequent. Instead, the total energy cost of the communication system is more important as it will depend on security requirements and vary from scenario to scenario.

2) Energy consumption: Energy consumption is one of the crucial issues for battery-powered wireless sensor nodes. To maximize lifetime of, e.g., remotely placed sensors, all operations running on them should aim at minimizing the amount of consumed energy. Considering a WSN security protocol, it is important to assess how efficient in terms of energy consumption its components are. Energy analysis has to evaluate the cost of both on-sensor processing and data transmission.

Table II presents energy consumption for different operations on an Imote2 sensor node running at 416 MHz. By measuring the duration of particular operations and the current draw during their execution we were able to calculate the amount of energy spent on these operations. The respective

TABLE III

Protocol components needed for different WSN applications (based on identified requirements) and their energy cost per BEX for HIP initiator and responder. Numbers in the round brackets – the cost of BEX using RSA and DH ("standard" HIP). Numbers before the brackets – the cost of BEX using ECDSA and ECDH. Numbers for "Bodynets PAN" and "Monitoring" – the cost of BEX based on LW certificates and a polynomial exchange.

Protocol components							**Requirements**	**WSN application types**			
Certificates		*Signatures*		*Puzzle*	*Key exchange*			*Battlefield*	*Bodynets*		*Monitoring*
PKC	LW	RSA	ECDSA		DH	Poly		(critical)	PAN	BEN	(lightweight)
[x]	[x]	[x]	[x]			(x)	Authentication	x	x	x	x
[x]	[x]						Authorization	x	x	x	x
					[x]	[x]	Confidentiality	x	x	x	(x)*
		[x]	[x]				Integrity	x	x	x	x
				x			DoS resistance	x		x	
x		[x]	[x]		x		WSN scalability	x		x	
	x					x	WSN lifetime		x		x
							Energy/I (mJ) $\rightarrow$	53.8 (471.4)	1.5	53.8 (471.4)	1.5
							Energy/R (mJ) $\rightarrow$	34.1 (467.0)	1.5	34.1 (467.0)	1.5

formula is

$$E = I \times U \times T,$$

where I is the current draw, U is the voltage (4 V in our case), and T is the execution time of an operation.

A reader should take into account that presented values correspond to the energy consumed by all applications running on a device, i.e. the whole Imote2 system. The term *idle* is used to account for an operation mode, when no applications besides system processes are running. As results show, the current draw in the idle state equals to 73 mA with the active (module inserted into kernel and radio interface brought up) and 52 mA with the deactivated radio module. Thus, a real overhead of an operation should be seen as the difference between the current draw value for the operation in question and the current draw in the idle state.

Table II does not contain current draw in a *sleep* and a *deep sleep* mode. Although Imote2 can operate in these low-power modes consuming as little current as 390 μA [5], our Linux-based Imote2 platform missed actual implementation of power management. In practice, each Imote2 should be provided with due support of low-power modes and be able to deactivate its radio and turn to a sleep state whenever it is not processing and transmitting data over a network.

Our results do not show a separate current draw number for Imote2 data transmission. The reason is that according to our observations, sending or receiving packets over radio does not incur an additional cost by itself but rather draws the same level of current as with enabled radio module. In other words, the value of 73 mA should be interpreted as the current draw level during Imote2's radio communications. Absolute amount of consumed energy will then depend on the duration of data transmission (or data size), which, in turn, is defined by a protocol in question.

C. *Analysis and applications*

In this section we present an analysis of HIP "standard" and alternative components applied to three WSN application types (see Table III): (1) a battlefield sensor network representing a "critical" WSN class with a maximum set of requirements; (2) a body sensor network representing a "hybrid" WSN class and consisting of two sub-networks, a PAN (Patient Area Network) and a BEN (Back-End Network), as they defined in [11]; and (3) a sensor network used for non-critical environmental monitoring (e.g., to sense humidity, temperature etc.) that in our example represents a "lightweight" class of WSNs.

Table III contains a summary of our analysis by listing in the middle a set of requirements for WSNs, on the left side - a set of HIP components and their alternatives, and on the right side - our three example WSN applications. The main possible alternatives we suggest in this paper are ECC-based algorithms for signature operations and shared secret negotiation, lightweight (LW) certificates instead of PKC certificates and a polynomial (Poly) exchange as a further possible replacement to a DH or a ECDH key exchange.

To show how WSN requirements correspond to protocol components and application types, in Table III we use "x" to indicate a mandatory option; "[x]" to indicate that either of two components with this sign can be used to satisfy a correspondent requirement; "(x)" to indicate that a "polynomial exchange" alone is sufficient to meet "authentication" requirement; and "(x)*" to indicate that "confidentiality" is not mandatory for "environmental monitoring" (in our scenario) but is already fulfilled by the use of "polynomial exchange".

As a part of our analysis, for each application we estimate energy cost per BEX for both HIP initiator and responder (each being Imote2). The results presented in Table III assume the use of PKC certificates and Diffie-Hellman (either traditional or ECC-based) for the most critical "battlefield" application and, in turn, LW certificates and a polynomial exchange for the least critical "monitoring" application. For estimations we relied on BEX versions depicted in Figures 1 and 2 and used energy consumption values from Table II. We did not include into BEX estimation two round trip times (RTT) since their value varies for different packet sizes and, in our experience, also depends on the quality of the radio driver and the signal level. Although the puzzle in our scenarios is of little usage, we kept it in calculations.

Overall, our measurements (Table II) and estimations (Table III) demonstrate a significant difference between the computation and energy cost of PKC and LW approaches. The energy cost of the most "lightweight" BEX variant in our scenarios (using LW certificates and a polynomial exchange) is equal to only 2.8% (for the HIP initiator) and 4.4% (for the HIP responder) of the cost of the BEX using ECDSA and ECDH, and to just 0.3% of the "standard" HIP BEX cost using conventional RSA and DH algorithms. The presented combination of security mechanisms can serve as a solid framework useful in design, development and evaluation of flexible communications security protocols applicable to a variety of WSN scenarios. Given a real WSN deployment case with knowledge of particular security requirements and the use of certain protocol components, the presented results would allow easy assessment of the protocol performance and estimation of the network lifetime.

VII. Conclusion

In this paper we proposed to use the Host Identity Protocol to enhance security of wireless sensor networks. To increase computational and energy efficiency of HIP in particular WSN applications, we suggested lightweight alternatives to its expensive PKC components in a form of ECC, a polynomial-based key establishment and lightweight certificates based on one-way hash functions.

We analyzed three particular WSN applications with different requirements and identified a set of protocol components required for each scenario. Our measurement results on the Imote2 sensor platform showed that in certain applications HIP can be applied with lightweight components allowing to speed up critical secure communications and to increase longevity of WSNs. Our measurement and estimation results can serve as a useful reference of Imote2 performance in possible future WSN deployments involving HIP and other presented security mechanisms.

Acknowledgments

We thank our colleagues from Data Communications Software research group, Aalto University School of Science and Technology, for providing a testbed for measuring power consumption.

References

[1] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *CCS '03: Proc. of the 10th ACM conference on Computer and communications security*, (New York, NY, USA), pp. 52–61, ACM, 2003.

[2] A. Liu and P. Ning, "Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks," in *IPSN '08: Proc. of the 7th international conference on Information processing in sensor networks*, (Washington, DC, USA), pp. 245–256, IEEE Computer Society, 2008.

[3] K. Piotrowski, P. Langendoerfer, and S. Peter, "How public key cryptography influences wireless sensor node lifetime," in *SASN '06: Proc. of the 4th ACM workshop on Security of ad-hoc and sensor networks*, (New York, NY, USA), pp. 169–176, ACM, 2006.

[4] R. Roman, C. Alcaraz, and J. Lopez, "A survey of cryptographic primitives and implementations for hardware-constrained sensor network nodes," *Mob. Netw. Appl.*, vol. 12, no. 4, pp. 231–244, 2007.

[5] "Imote2 wireless sensor node platform. Product datasheet." Available online at: http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/Imote2_Datasheet.pdf. Accessed 18 May 2010.

[6] K. Sun, A. Liu, R. Xu, P. Ning, and D. Maughan, "Securing network access in wireless sensor networks," in *WiSec '09: Proc. of the 2nd ACM conference on Wireless network security*, (New York, NY, USA), pp. 261–268, ACM, 2009.

[7] R. Moskowitz and P. Nikander, "Host Identity Protocol architecture." IETF RFC 4423, May 2006.

[8] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson, "Experimental Host Identity Protocol (HIP)." IETF RFC 5201, Apr. 2008.

[9] A. Gurtov, *Host Identity Protocol (HIP): Towards the Secure Mobile Internet*. Wiley and Sons, 2008.

[10] J. P. Walters, Z. Liang, W. Shi, and V. Chaudhary, "Wireless sensor network security: A survey," in *Book chapter of Security in Distributed, Grid, and Pervasive Computing, Yang Xiao (Eds.)*, pp. 367–403, Taylor and Francis Group, 2007.

[11] O. Garcia-Morchon, T. Flack, T. Heer, and K. Wehrle, "Security for pervasive medical sensor networks," in *MobiQuitous'09: Proc. of the 6th Annual International Conference on Mobile and Ubiquitous Systems*, ICST/IEEE, July 2009.

[12] L. Nachman, J. Huang, J. Shahabdeen, R. Adler, and R. Kling, "Imote2: Serious computation at the edge," in *IWCMC'08: Proc. of the International Wireless Communications and Mobile Computing Conference 2008*, pp. 1118–1123, aug. 2008.

[13] R. Adler, J. Huang, R. Kong, P. Muse, L. Nachman, R. Shah, C.-Y. Wan, and M. Yarvis, "Edge processing and enterprise integration: Closing the gap on deployable industrial sensor networks," in *Proc. of the 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, 2007 (SECON'07)*, pp. 620–630, IEEE, June 2007.

[14] J. A. Rice and B. F. Spencer Jr., "Structural health monitoring sensor development for the imote2 platform," in *Proc. of the SPIE Smart Structures/NDE*, pp. 1–12, SPIE, 2008.

[15] D. Halperin, T. S. Heydt-Benjamin, K. Fu, T. Kohno, and W. H. Maisel, "Security and privacy for implantable medical devices," *IEEE Pervasive Computing*, vol. 7, no. 1, pp. 30–39, 2008.

[16] T. Heer and S. Varjonen, "HIP Certificates: draft-ietf-hip-cert-02," Oct. 2009. Work in progress.

[17] D. Kuptsov, O. Garcia-Morchon, K. Wehrle, and A. Gurtov, "Brief announcement: Distributed trust management and revocation," in *Proc. of the 29th Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC'10)*, (New York, NY, USA), pp. 1–3, ACM, 2010.

[18] A. Khurri, E. Vorobyeva, and A. Gurtov, "Performance of Host Identity Protocol on lightweight hardware," in *Proc. of the 2nd ACM/IEEE International Workshop on Mobility in the Evolving Internet Architecture (MobiArch'07)*, (New York, NY, USA), pp. 1–8, ACM, Aug. 2007.

[19] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Transmission of IPv6 packets over IEEE 802.15.4 networks." IETF RFC 4944, Sept. 2007.

[20] R. Housley, "Using Advanced Encryption Standard (AES) CCM mode with IPsec Encapsulating Security Payload (ESP)." IETF RFC 4309, Dec. 2005.

[21] R. C. Merkle, "A digital signature based on a conventional encryption function," in *CRYPTO '87: Proc. of a Conference on the Theory and Applications of Cryptographic Techniques on Advances in Cryptology*, (London, UK), pp. 369–378, Springer-Verlag, 1988.

[22] R. C. Merkle, *Secrecy, authentication, and public key systems*. PhD thesis, Stanford University, Dept. of Electrical Engineering, 1979.

[23] Certicom Research, *Standards for Efficient Cryptography. SEC 1: Elliptic Curve Cryptography*. Aug. 2008.