Towards Adaptable Security for Energy Efficiency in Wireless Sensor Networks

Nikos Fotiou, Giannis F. Marias, George C. Polyzos, Pawel Szalachowski, Zbigniew Kotulski,
Michael Niedermeier, Xiaobing He and Hermann de Meer

***Abstract*— Modern sensors are portable, embeddable, they offer multiple connectivity options and enough processing power that allows the performance of advanced operations. Multiple sensors can be used together forming a wireless sensor network (WSN). Ubiquitous WSNs are expected to play a significant role in the future, assisting users in their everyday life. In this paper we present an intriguing application of WSNs: health monitoring of hospital patients. We focus on the security aspects of this application and identify security threats and requirements. Moreover, we argue that existing security solutions are energy hungry, therefore they are inappropriate for WSNs and we propose a new security design approach: adaptable security. Our design approach advocates that security mechanisms should be able to adapt their complexity by ranking the security requirements of each operation, achieving this way better energy efficiency.**

***Index Terms*—Energy efficiency, Security, Wireless sensor networks**

I. INTRODUCTION

Wireless sensors are expected to become a core component of future ubiquitous networks. As technology evolves, sensors become smaller, smarter, with even more processing and interconnection capabilities and their application field becomes wider and wider. Sensors are already embedded in daily usage devices—such as smartphones—and the fact that they are being used on regular basis reveals their great potentials. Due to their size, sensors, can be very pervasive which combined with their advanced sensing and inteconnection features may jeopardize user's privacy. On the other hand security mechanisms require complex computations which lead to increased energy consumption and therefore limit the operational scope of sensors. As security mechanisms have not been designed with energy efficiency in mind, they cannot always be applied in sensor networks without firstly being reconsidered and modified. In this paper we introduce the concept of adaptable security, i.e, how security mechanisms can be adapted in order to satisfy the security and energy requirements of a specific application. The use case application for this paper is the health monitoring of hospitals' inmates

A. Use case secnario

WSNs can be used in hospitals in order to monitor patients' health status and notify doctors in case of emergency. A hospital-based WSN used for this purpose, will ameliorate patients' stay in the hospital as they will not have to be restricted in a room, but will be able to move around the hospital freely and safely. Sensors deployed on patients' body will be able to monitor their vital signs as well as their position. Moreover by using their processing and interconnection capabilities those sensors will be able to detect an emergency situation and notify doctors, updating them at the same time with the patient's health status and position.

To prolong autonomy and battery life and even to limit interference to other hospital devices, wireless communication should use the minimum power level possible. In order to achieve this target, sensors can create small clusters coordinated by low emission wireless access points acting as Cluster Heads (CHs). CHs will be responsible for handling sensor nodes that enter and leave the cluster, as well as for assuring the proper and secure operation of each node. Moreover CHs are chosen to be the gateways to a wired backbone network that will be used to transfer data from sensors to a central control system. Not all members of the cluster will have direct communication with the CH; in order to assure as low power transmission as possible, sensors will communicate in ad-hoc mode and the nodes that are closer to the CH will act as transit nodes for the nodes that are further away. As patients will freely move around, clusters are expected to be dynamic, with high churn.
The security requirements of this system are the following:

- Data should be confidential and only authorized users should have access to it. As the data captured and transmitted by the sensors is related to the health status of a patient, it is protected by national and EU legislation. Therefore data confidentiality is a key condition for the deployment of such a system.
- Data integrity should be protected and any data modification should be immediately detected.
- The system should have high availability; it should be robust and fault tolerant. Any system disruption may be lethal.
- The system should preserve users' privacy and it should not be possible for a 3rd party to infer any information about the users' identity, health status, or location.
- The system should be able to detect security breaches and react immediately.

In addition to the above security requirements the following adversary model is considered in our approach:

- Attackers have unlimited energy resources.
- Attackers have more computational power than a sensor or a CH.
- Attackers are able to sniff and capture the transmitted data.

The aforementioned attackers are trying either (a) to block the system's operation (either by causing a denial of service communication attack, or by saturating various sensors' energy resources) or (b) to extract information about patients. Attackers of category (a) are using the following types of attack:

- Fault data injection
- Jamming attacks
- Replay attacks
- Modification attacks

Attackers of category (b) are using the following types of attack:

- Data analysis
- Traffic analysis

Attackers of both categories also try to

- Impersonate legitimate entities

II. ADAPTABLE SECURITY

Each security service (such as integrity, confidentiality, and authentication) can be realized using various security mechanisms. Each mechanism has different properties and security levels. For example, authentication can be achieved using both symmetric message authentication codes (MAC) and digital signatures (which also ensure non-repudiation). The security level of these mechanisms is different as well as their efficiency. This is especially important in the case of resource-constrained environments like Wireless Sensor Networks (WSNs).

Adaptable security models, often called Quality of Protection (QoP) models, allow the calculation of different versions of the mechanism that protects the transmitted data, achieving different security levels. The security level depends on parameters such as:

- parameters of used cryptographic element (e.g., key length, block length)
- importance of protected content
- message priority,
- probability of an attack
- assets gained during successful attack

Examples of various QoP models are presented in [3,4,5]

It is clear that traditional cryptographic protocols are not suitable for WSNs as they require significant energy in order to process and transmit data. The security each protocol achieves has to be modeled as a function of its energy consumption. For example, key exchange messages should be protected using the highest security level while signaling or diagnostic traffic can be protected using faster (and less secure) methods.

In the rest of this section we shortly describe the QoP model presented in [2]. This model uses three primary parameters in order to calculate the security level of a mechanism:

1. L: the protection level,
2. P: the probability of an incident occurrence,
3. ω: the impact of a successful attack.

The protection level is defined in percents for each security service. It describes the contribution of the protection of a particular service to the global protection level and it is associated with risk. Each security mechanism (e.g. symmetric encryption) is characterized by the following main parameters:

- LZ – assets gained during successful attack on a given security element (100% = compromising the whole protocol),
- LK – the knowledge required for an attack (100% = expert),
- LP – cost of an attack (100% = the highest cost),
- C – communication overhead of an attack, $C \in [0,0.1]$ (0.1 = the highest threat),
- M – implementation complexity. The difficulty in implementing increases the probability of incorrect configuration. Error reports are an additional source of information, etc. $M \in [0,0.1]$ (0.1 = the highest threat).

The impact of a successful attack is calculated for each service and in each step of the protocol. The parameters used for the calculation of the impact of a successful attack are presented below:

- F – financial losses during a successful attack on given security elements (100% is the total financial loss),
- α – necessary financial costs for repairing the damages gained during a successful attack (100% is the maximal cost),
- β – losses of the value of the company shares or the company reputation (100% is the maximal market loss).

Additionally, LZ parameter is used to compute ω.

The security level realized by a given version of a cryptographic protocol is based on the three main parameters (L,ω,P) and can be described as:

$$L=(1-\omega)(1-P)$$

A. SPOT

The Security Protocol Optimizing Tool (SPOT) [1] is a tool that optimizes cryptographic protocols using the model presented above. This tool is composed by the following four modules.

The presentation module
It presents a user friendly graphic interface to the tool.
Core module
It is responsible for the calculation of the protection level, of the probability of incident occurrence, of the impact of a successful attack and of the global security level. The Core module accepts as input XML files that contain the protocol parameters. It outputs data that includes instructions on how to configure the system which realize the cryptographic protocol. The output data from the core module can be visualized using the visualization mode.

Visualization module
It is responsible for the presentation of the output data from the Core module in user friendly ways. The user can compare different versions of the cryptographic protocol in the separate tables.
Optimization module
It calculates all possible versions of the protocol and find these which fulfill defined conditions.

B. TESLA in the adaptable security model (case study)

Timed Efficient Stream Loss Tolerant Authentication (TESLA) is a light-weight hash chains-based solution that allows all receivers to check the integrity and authenticity of each packet they received [6].

Even lightweight protocols, like TESLA, provide different levels of security. TESLA uses three kinds of authenticators:

- Authentication Tags
- Digital Signatures
- Group MAC Tags

Each of them plays a different role in secure communication:

- authentication tag (MAC tag) is used in data packets,
- digital signature is used in TESLA signaling packets

An optional group authentication tag, can be added to all the packets to mitigate attacks coming from outside of the group.

Authentication in TESLA can be implemented using different cryptographic mechanisms with different protection levels that depend on parameters such as key length. Some recommended key-lengths for different scenarios can be found in [7].

The key-lengths that provide the maximum security cannot be used in wireless nodes as they introduce significant computational overhead and, what it follows, need more energy to perform calculations.

For energy saving a different approach can be used for algorithms selection. It is based on a concept of security level which depends not only on the protection level of cryptographic mechanisms applied but also on the actual risk of attack on a service [2]. Depending on changing external conditions the protection level can be adaptably changed to keep minimal security level and save energy not overestimating the protection level when the risk of attack is small.

To calculate the protection level for different cryptographic mechanisms used in TESLA authenticators SPOT is applied. For each case we calculate the energy required in order to make calculations (for selected hardware platforms) and we show how adaptable approach can save battery resources without loss of the system protection level.

In our example we focus on the authentication service which is realized by three mechanisms: digital signatures, MAC and group MAC. Each mechanism can be implemented using various cryptographic techniques (with varying input parameters) such as those mentioned below:

1. Digital Signature (LZ=80%):
(a) NTRU, ECDSA-160, RSA-1024 (80 bit secure, LK=60%, LP=60%)
(b) RSA-2048, ECDSA-192 (96 bit secure, LK=60%, LP=75%)
(c) RSA-3072, ECDSA-224 (128 bit secure, LK=60%, LP=95%)

2. MAC (LZ=40%):
(a) CMAC,HMAC-MD5 (64 bit secure, LK=50%, LP=50%)
(b) HMAC-SHA1 (80 bit secure, LK=50%, LP=60%)

Security of methods and LK, LP factors are estimated using recommendations from [7]. Bootstrapping is the most important phase, so assets protected by digital signature are very high (LZ=80%). Assets protected by MAC operation are assessed as 40%.

Using SPOT we calculate security level for given implementations. Results are presented in Table 1.

Digital Signature	MAC scheme	Security Level
1.a	2.a	0.394
1.a	2.b	0.440
1.b	2.a	0.523
1.b	2.b	0.566
1.c	2.a	0.641
1.c	2.b	0.716

Table 1: Used cryptographic mechanisms with corresponding security levels

From table 1 it can be seed that security level varies between 0.394 and 0.716. To make our example complete we add efficiency results. Table 2 presents performance results for the presented methods in a WSN environment. All results were achieved on standard node hardware: MICAz (ATmega128L 8Mhz) and for standard WSN traffic (up to 64 byte message).

	Sign	Verify
NTRU Sign	619ms	78ms
ECDSA (160)	918ms	918ms
RSA (1024)	10990ms	430ms
RSA (2048)	83260ms	1940ms
ECDSA (192)	1240ms	-
RSA (3072)	-	-
ECDSA (224)	2190ms	-
CMAC (AES)	1.4ms	1.4ms
HMAC (MD5)	3.7ms	3.7ms
HMAC (SHA1)	4.8ms	4.8ms

Table 2: Performance results of used methods in WSN environment

Implementations' details of these results are presented in [8, 9, 10]. Using Table 1 and Table 2 we can select appropriate mechanisms for our communication profile.

III. Energy consumption model for wireless sensor networks

To model the energy consumption of a WSN, several aspects of the overall system have to be carefully reflected, to achieve a flexible, yet precise formula. This can be done by modeling on one hand the static base consumption (meaning the idle power consumption of the sensor nodes) in both awake (P_{base_a}) and sleep (P_{base_s}) states. On the other hand, during the awake state, there are several dynamic functions which combined make up the dynamic consumption part ($E_{dynamic_t}$), which have to be investigated apart from the base consumption. Examples of dynamic consumption are sending and receiving of packets, application of encryption algorithms and other security-related tasks. These have to be separately evaluated in an energy consumption model. Therefore, our first approach in modeling the overall energy consumption ($E_{complete_t}$) for a WSN in a communication round (t) would be:

$$E_{complete_t} = E_{base_t} + E_{dynamic_t}$$

where:

$$E_{base_t} = t_{a_t} * P_{base_a} + t_{s_t} * P_{base_s}$$

$$E_{dynamic_t} = \sum_{j=1}^{k} \#j_t * E_j$$

where:

$t = t_{a_t} + t_{s_t}$
$E_j = t_j * P_j$
$P_{base_a} = \sum_{i=1}^{n} P_{i_a}$
$P_{base_s} = \sum_{i=1}^{n} P_{i_s}$

where:

$t =$ Time of one communication round
$t_{a_t} =$ Time spent in awake mode during communication round t
$t_{s_t} =$ Time spent in sleep mode during communication round t
$P_{i_a} =$ Power consumption of a component i in its awake mode
$P_{i_s} =$ Power consumption of a component i in its sleep mode
$t_j =$ Time needed for one execution of task j
$P_j =$ Power consumption of a single execution of task j
$j =$ A task performed in addition to the base load (encryption, decryption, sending / receiving messages, ...)
$\#j_t =$ Number of times task j is executed during communication round t
$k =$ Number of overall tasks that are relevant to the dynamic energy consumption
$i =$ Number of components on the sensor node (sensors, GPS, RAM, CPU, ...)
$n =$ Number of overall components that are relevant to the base energy consumption

It has to be mentioned that parts of the model's input parameters need to be provided by empirical test results. Specifically, this applies to P_{i_a}, P_{i_s}, t_j and P_j. To implement such an energy consumption model in a real-life system would in addition require a precise clock on the sensor board, to enable a high-resolution time measurement for calculation purposes. The complexity of the model can easily be in- or decreased, depending on the hardware capabilities of the sensor nodes it should be used for. While the model explained above is quite fine granular, values like $E_{dynamic_t}$ could also be approximated by not counting each execution of a certain method during a communication round *t*, but instead using average values based on empirical or simulation data. This would result in a much less complex model, as only the awake and sleep times of a node would be actively measured. Of course, the accuracy of the model would suffer in the course of this. To explain the model's usage a bit further, we use two example scenarios to show the impact of higher security standards on the energy consumption. For both scenarios, we assume that the same sensor node components, energy consumptions for tasks as well as communication round times are used:

t_{a_t}	150 ms
t_{s_t}	100 ms
P_{i_a}	5.00 mW
P_{i_s}	0.01 mW
$P_{encryption}$	0.01 mW
$P_{decryption}$	0.01 mW
$P_{send\_message}$	0.30 mW
$P_{receive\_message}$	0.15 mW
$P_{authentication}$	0.07 mW
$P_{verification}$	0.40 mW
$t_{encryption}$	2.00 ms
$t_{decryption}$	2.00 ms
$t_{send\_message}$	6.00 ms
$t_{receive\_message}$	4.00 ms
$t_{authentication}$	1.00 ms
$t_{verification}$	7.00 ms

The difference in both scenarios is therefore limited to the dynamic part of the energy consumption ($E_{dynamic_t}$). The specific differences between the test scenarios will be show in the following.

Scenario 1: The first scenario shows a low-security WSN implementation that only uses basic encryption on some messages without authentication.

j	{*encryption,* *decryption,* *send_message,* *receive_message*}

k	4
$\#encryption_t$	4
$\#decryption_t$	8
$\#send\_message_t$	6
$\#receive\_message_t$	12

Using these values, the energy consumption for the communication round t is:

$$E_{base_t} = 0.15\ s * 5\ \text{mW} + 0.1\ \text{s} * 0.01\text{m W} = 0.75\ \text{mWs} + 0.001\ \text{mWs} = 0.751\ \text{mWs}$$

$E_{dynamic_t} = \sum_{j=1}^{4} \#j_t\ *\ E_j = 4\ * 0.002\ s * 0.01\ mW + +8 * 0.002\ s * 0.01\ mW + 6 * 0.006\ s * 0.3\ mW + 12 * 0.004\ s * 0.15\ mW = 0.01824\ mWs$

$$E_{complete_t} = E_{base_t} + E_{dynamic_t} = 0.751\ mWs + 0.01824\ mWs = 0.76924\ mWs$$

Scenario 2: The second scenario shows a high-security WSN using encryption and authentication for all messages.

j	{*encryption, decryption, send_message, receive_message, authentication, verification*}
k	6
$\#encryption_t$	6
$\#decryption_t$	12
$\#send\_message_t$	6
$\#receive\_message_t$	12
$\#authentication_t$	6
$\#verification_t$	12

Using these values, the energy consumption for the communication round t is:

$$E_{base_t} = 0.15\ s * 5\text{mW} + 0.1\ \text{s} * 0.01\ \text{mW} = 0.75\ \text{mWs} + 0.001\ \text{mWs} = 0.751\ \text{mWs}$$

$E_{dynamic_t} = \sum_{j=1}^{6} \#j_t\ *\ E_j = 6\ * 0.002\ s * 0.01\ mW + 12 * 0.002\ s * 0.01\ mW + 6 * 0.006\ s * 0.3\ mW + 12 * 0.004\ s * 0.15\ mW + 6 * 0.001\ s * 0.07mW + 12 * 0.007\ s * 0.4\ mW = 0.05238\ mWs$

$$E_{complete_t} = E_{base_t} + E_{dynamic_t} = 0.751\ mWs + 0.05238\ mWs = 0.80338\ mWs$$

IV. Towards an Application Specific Solution

From the identified security requirements and threats of our use case, it can be understood that multiple security mechanisms have to be applied. Some of the security mechanisms that can be considered in our approach are the following:

Elliptic curve cryptography (ECC)
ECC is a public-key encryption scheme which can achieve the same security levels as traditional public-key security schemes—such as RSA—with much smaller encryption keys (but various trade-offs exist).

RSA
RSA is one of the most widespread public-key cryptography algorithms. It is used as public key encryption as well as digital signature scheme. The security of the RSA depends on factoring problem of $n=pq$, where p,q are large prime numbers. Unfortunately RSA requires heavy computations, thus it is not recommended for limited devices.

ECDSA
ECDSA is a lightweight variant of the Digital Signature Algorithm (DSA. The efficiency of ECDSA is caused by using Elliptic Curve Cryptography (ECC). The size of key in ECDSA is smaller (with reference to RSA) and because of this, the scheme is more efficient. ECDSA with 160 bits key (ECDSA-160) achieves the same security level as RSA-1024.

NTRUSign
NTRUSign is a public key cryptography digital signature algorithm. It is lattice-based scheme using close vector problem as security basis. NTRUSign is designed for constrained environments but it is not widespread and well-known because that algorithm is patented. The security of NTRUSign is comparable with ECDSA-160 and RSA-1024.

HMAC
HMAC is variant of message authentication code (MAC) based on cryptographic hash function. The security of HMAC is connected with used hash function. The most used variants are HMAC based on MD5 (HMAC-MD5) and HMAC based on SHA1 (HMAC-SHA1). MD5 produces 128bit tag while SHA1 produces 160bit tag, so HMAC based on SHA1 is considered as more secure.

CMAC
CMAC is special mode of block cipher, designed for message authentication. It uses encryption function of block cipher and symmetric key. The security of this mechnisms is comparable to HMAC-MD5. CMAC can be very efficient on constrained devices and for specific data, like short messages.

eXtensive Access Control Markup Language (XACML)
XACML is a declarative XML-based access control policy language that can be used in order to define access control rules on items. Access control rules specify the actions that each identifiable actor can perform to an item.

Timed Efficient Stream Loss Tolerant Authentication (TESLA)
TESLA is a light-weight hash chains-based solution that allows all receivers to check the integrity and authenticity of each packet they received.

Data Obfuscation
Data obfuscation is a technique that prevents traffic analysis through the addition of extra “noise” traffic.

Trusted Platform Module (TPM)
TPM is tamper resistant, trusted hardware that is used to digitally sign the hardware generated data, e.g., the signals that a sensor receives. TPM can assure data fidelity, i.e., that the captured data is the same as the data transmitted by the user.

Moreover each solution requires the existence of sub-systems, such as key revocation and key renewal mechanisms. Finally we assumed that there is an authentication system, as well as a centralized intrusion detection system.

It can be understood that no matter how effective these mechanisms are, they cannot be applied directly in WSNs, as they will immediately saturate their energy resources. Those security mechanisms require complex operations and lead to the transmission of bigger data packets. Nevertheless it can be observed that various parameters of these mechanisms—such as encryption key length, ciphertext size, frequency of communication and many others—affect the sensor's energy consumption; thus, by modifying them, we can achieve better energy efficiency. On the other hand, not all data items transmitted have to be secured in the same way. As an example a communication protocol specific message—such as a ping message—does not require the same security level as a transmission that carries the patient's identity. Similarly an “everything goes well” message does not have to be transmitted as fast as an “emergency” message.

Our security design approach is based on those observations and it aims to achieve energy efficiency by estimating the energy consumption required in order to have *adequate* security. Towards this direction we estimate the energy consumption of each security mechanism, we define the mechanisms needed in order to achieve a security requirement and we refine the security requirements of each system operation. More precisely:

- For each security mechanism we create the respective adaptive security model.
- For each security requirement and attack scenario we decide which security mechanisms (and with which input parameters) have to be used. We use the energy consumption model to calculate the energy consumption of the defense mechanism
- For each system operation we decide the minimum security requirements needed to prevent an attacker to launch a successful attack by utilizing this operation. Thus, we can decide the security mechanisms that have to be used in order to protect this operation and, therefore, their energy cost.
- For each operation, we define its normal operation frequency as a function of the sensor energy level as well as the system state. (Each sensor energy level can e.g. have the values: low, normal, high. The system state can also e.g. be either normal or under attack.)

Using the above functions as input, each sensor is able to decide if it can operate normally, with adequate security and based on its current energy level and system state. In each time slot, a sensor calculates the actions it has to perform and their cumulative energy cost. If the energy cost of those actions is lower than the sensor’s energy level, then it can continue operating normally. With this technique a sensor can also estimate for how much time it can continue operating normally and, therefore, notify accordingly the user.

The estimation of the energy cost of each security mechanism is not a trivial task. Mathematic models for analyzing and evaluating security protocols have to be created. Those models have to be verified through simulations and emulations

ACKNOWLEDGMENT

The work reported in this paper was supported by the SJRP E-key-nets as part of the FP7 NoE Euro-NF.

REFERENCES

[1] P.Szalachowski, Z.Kotulski, SPOT: Optimization tool for network adaptable security, CN 2010, Communications in Computer and Information Science, Vol.79, pp.269-279,Springer-Verlag, Berlin Heidelberg 2010
[2] B. Ksiezopolski, Z. Kotulski, Adaptable security mechanism for the dynamic environments, Computers & Security 26 (2007) 246-255
[3] P. Schneck, K. Schwan, Authenticast: An Adaptive Protocol for High-Performance, Secure Network Applications, Technical Report GIT-CC-97-22 (1997)
[4] C.S. Ong, K. Nahrstedt, W. Yuan, Quality of protection for mobile applications, IEEE International Conference on Multimedia & Expo (2003) 137-140
[5] Y. Sun, A.Kumar, Quality od Protection(QoP): A quantitative methodology to grade security services, 28th confrence on Distributed Computing Systems Workshop (2008) 394-399
[6] RFC 5776, V. Roca, A. Francillon, S. Faurite, Use of Timed Efficient Stream Loss-Tolerant Authentication (TESLA) in the Asynchronous Layered Coding (ALC) and NACK-Oriented Reliable Multicast (NORM) Protocols, IETF April 2010.
[7] ECRYPT http://www.keylength.com/en/3/
[8] P.Szalachowski, B.Ksiezopolski, Z.Kotulski, On authentication method impact upon data sampling delay in Wireless Sensor Networks, In: A. Kwiecień, P. Gaj, and P. Stera [Eds.], CN 2010, Communications in Computer and Information Science, Vol.79, pp.280-289, Springer-Verlag, Berlin Heidelberg 2010
[9] B. Driessen, A. Poschmann, and C. Paar. Comparison of innovative signature algorithms for WSNs. In Proceedings of the first ACM conference on Wireless network security, WiSec ’08, pp. 30–35, New York, NY, USA, 2008. ACM.
[10] N. Gura, A. Patel, W. Arvinderpal, H. Eberle, and S. C. Shantz. Comparing elliptic curve cryptography and RSA on 8-bit CPUs. In: LNCS 3156 , pp 119-132, Springer-Verlag, Berlin Heidelberg 2004.