SeNsIM-SEC: security in heterogeneous sensor networks

Valentina Casola, Alessandra De Benedictis, Antonino Mazzeo and Nicola Mazzocca
Dipartimento di Informatica e Sistemistica
University of Naples Federico II, Napoli, Italia
Telephone : +39 0817683907
Fax : +39 0817683916
Email: {valentina.casola,alessandra.debenedictis,antonino.mazzeo,nicola.mazzocca}@unina.it

***Abstract*—Wireless sensor networks are widely used in several application domains thanks to their data acquisition and processing capabilities and their decentralized and self-organizing nature. A widely distributed monitoring system is typically characterized by the need to integrate a large amount of data; if considering complex critical environments such as hospitals, these data often have different security requirements, to be addressed by means of specific security protocols and architectures. In such a distributed system, security must be addressed at different levels, namely the physical node level, the inter-node communication level, and the application level. In this paper, we focus our attention on security problems related to the data exchange between sensor nodes and propose an hybrid cryptosystem based on Elliptic Curves, aimed to ensure confidentiality, integrity ad authentication requirements to the inter-nodes communication. The integration issue has been addressed by proposing an extension of the SeNsIM integration platform, in order to enable the management of heterogeneous networks having different security requirements. At this aim, we have developed a flexible wrapper to connect the whole system to the secured network, and carried out a performance analysis of the overhead introduced by security mechanisms, showing the feasibility of the proposed cryptosystem and the platform scalability and extensibility features.**

I. Introduction

Wireless Sensor Networks (WSNs) are widely used in several application domains as environmental monitoring [13], detection and classification of objects in military and civil settings [6], agriculture procedures [2], automotive [14] and health monitoring [12]. Their decentralized and self-organizing nature makes the deployment very easy and this facilitates their adoption in any context without requiring the existence of any supporting infrastructure.

Furthermore, they are widely employed in critical scenarios and security issues are becoming a fundamental concern to be addressed by means of proper security policies and mechanisms. WSNs can be considered as belonging to wireless ad hoc networks, but they present a lot of distinctive features making the well-known security solutions not directly applicable. Unlike ad-hoc networks nodes, WSN nodes are typically provided with constrained processing and storage capabilities and limited energy resources; they are prone to failures due to harsh deployment environments and are easy to be compromised due to typically unattended operations. Finally, a WSN is often characterized by a dynamic topology due to node joining, mobility or failure, thus introducing further security and reliability issues.

The main problems that we want to face in this paper are primary related to two different aspects: (i) interoperability of different sensor networks (in terms of technologies and security mechanisms), (ii) security mechanisms that can be enforced within a sensor network to ensure confidentiality, authentication and integrity of exchanged messages.

The problem of sensor networks integration and interoperability has been extensively discussed in literature; we handled it with an integration platform named SeNsIM [3], a scalable software architecture which enables the development of applications based on multiple heterogeneous sensor systems providing a standard way to manage, query, and interact with them. By extending the SeNsIM architecture, we can build different applications to manage data with different security requirements, too. Security must be addressed at different architecture layers, namely at the physical node level, at the inter-node communication level, and at the application level. In this paper, we will focus our attention on security problems related to the data exchange between sensor nodes; we want to satisfy confidentiality, integrity and authentication requirements, but traditional network security protocols cannot be applied because of the limited resources and capabilities available on the sensor nodes.

At this aim, we will present an hybrid cryptosystem based on Elliptic Curves and we illustrate the implementation of key management and message signature protocols and their integration within the SeNsIM framework. Such cryptosystem has been implemented on top of the well known TinyOS sensor middleware by exploiting the WM-ECC library, a suite developed for wireless sensor motes turning out to be the most efficient publicly available ECC implementation. In order to integrate our secured network into SeNsIM we have developed an extensible wrapper,

able to interface with any TinyOS-based network via the UART interface.

The introduction of security requirements opens new possible scenarios of adoption of sensor networks but can introduce a huge overhead to the whole architecture; at this aim, in the last part of the paper, we will present a performance analysis on the overhead introduced by security mechanisms and we discuss about the feasibility and scalability of the proposed extensions.

The reminder of the paper is structured as follows: in Section 2 a brief overview of the context and motivation behind our research activity is drawn, and in Section 3 a reference architecture that includes both the heterogeneous network integration aspects and the proposed security system is presented. In Section 4 we will illustrate significant details on the implementation of our proposal and evaluation results will be presented and discussed. Finally in Section 5 some conclusions and future works will be drawn.

II. Security in Wireless Sensor Networks

As illustrated in Figure 1, a typical monitoring system is made of different sensor networks that can be heterogeneous in the technology aspects, in the data formats, in synchronization and localization standards and so on. They can be connected in different ways and their data should be elaborated by the same application to enrich the knowledge of observed complex phenomena. Providing security services in wireless sensor networks is a technical challenge, due to hostile deployment environments and resource limitations.

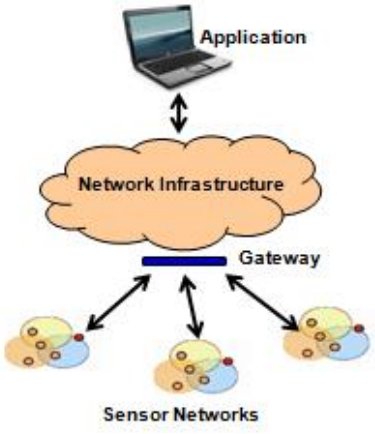

Fig. 1. A typical monitoring system

As illustrated in Figure 2, the monitoring infrastructure can be considered as structured into two main layers, namely the *sensor network layer* and the *distributed application layer*.

The *sensor network layer* can be further divided into two levels:

- Physical level: is responsible of the processing of the locally generated data at the node level.
- Transport level: controls the communication between the nodes of the network.

The *application layer* deals with the fusion and high level management of the data sensed by different heterogeneous networks; it can be considered as structured into two levels:

- Integration level: is responsible of the integration of data belonging to different sensor networks; it typically enforces a translation in a common data model [1].
- User level: executes the user distributed applications, which typically query the underlying networks and sensor features and manipulate the retrieved results for aggregation and decision purposes.

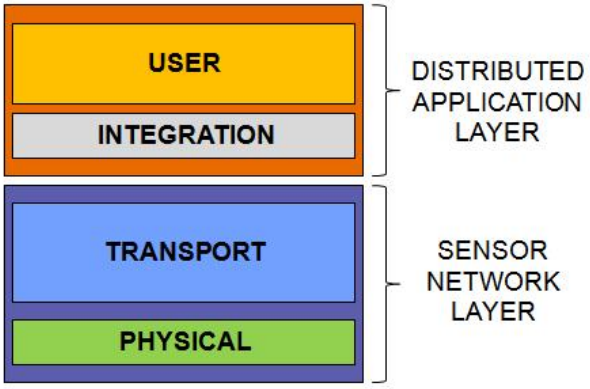

Fig. 2. A monitoring system layered view

In such scenario, security issues arise at different levels: at the *application layer*, data retrieved from the different networks are typically processed in a distributed manner, thus raising well-known issues dealing with secure network communication and access control; as this kind of processing is usually done by PC-class devices, the application layer does not suffer of the problems related to the limited resources of sensor nodes, and the well-known security protocols can be directly applied.

As for the *sensor network layer*: at the *transport level* it is necessary to secure data exchanged between nodes, and this can be achieved with the adoption of proper security protocols and mechanisms that take in consideration the limited resources; at the *physical level*, it is necessary to provide mechanisms for protecting nodes against physical tampering and DOS and jamming attacks.

The specific features of the sensor nodes make difficult the direct application of the existing security approaches to the area of wireless sensor networks: most security protocols are based on cryptographic operations, which massively involve the adoption of keys and complex mathematical functions that require dedicated computational resources and turn out to be critical from a performance and power consumption point of view.

The main problem to face with when setting up a secure communication between nodes is the way cryptographic keys are established at each node. There are two main well-known mechanisms to handle this problem: in Symmetric Key Cryptography (SKC) a unique secret shared key is used for both encrypting and decrypting messages, while in Public Key Cryptography (PKC) each node manages a couple of keys.

Several implementations of Symmetric Key Cryptography (SKC) algorithms in WSN have been proposed in literature (TinySec [7], MiniSec [8], ZigBee [19] and SNEP [10]), thanks to their low computational costs that make them well suited for realization on sensor devices. Even if symmetric schemes are very attractive for their energy and memory efficiency, they require complex and resource expensive key distribution and management protocols, resulting in a heavy traffic in the network and in complex and not scalable architectures. Moreover, symmetric cryptography only fulfills confidentiality requirements, while not considering other security issues such as authentication and integrity. As a matter of fact, an important security requirement which arises within the sensor network domain is the broadcast authentication, that is the capacity of a sender to broadcast messages to multiple nodes in an authenticated way, which can be achieved only via asymmetric schemes.

The use of asymmetric schemes in sensor networks has been usually considered as "nearly impossible" because they are power consuming and require a large amount of computational and storage resources. However, such schemes are very attractive, because they can ensure a higher degree of security while guaranteeing a greater flexibility and manageability than symmetric ones: thanks to them, any two sensors can establish a secure channel between themselves; moreover, as nodes do not share the same common key for encrypting/decrypting messages, the "capture" of some sensor devices will not affect the security of others. Rivest-Shamir-Adelman (RSA) algorithm [11] and Elliptic Curve Cryptography (ECC) [4] are among the most well known public key algorithms used in security systems, the latter being an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields.

Many protocols have been developed based on the ECC operations, as the Elliptic Curve Diffie-Hellman (ECDH) key agreement technique [4], which provides two communicating nodes with the possibility of achieving the same secret key without physically exchanging it across the network, and the Elliptic Curve Digital Signature Algorithm (ECDSA) [4] a variant of the Digital Signature Algorithm (DSA) that operates on elliptic curve groups, which can be used for signature generation and verification.

In order to overcome the drawbacks of both PKC and SKC schemes, a hybrid approach could be adopted; In [5] a hybrid security system is presented, relying upon a symmetric key function to encrypt the communication channel between each couple of nodes, and a public key function to ensure authentication of the base station and to establish the symmetric keys between the base station and each of the motes: this way, a higher level of security can be achieved, while limiting as much as possible the cryptographic computational load.

III. SeNsIM-SEC Architecture

As previously said, in many cases a monitoring infrastructure should provide the integration of critical data with other types of information retrieved by the surrounding environment, with different security requirements, too.

The architecture we propose in this paper is built upon SeNsIM (Sensor Networks Integration and Management) [3], a framework that was designed for integration of heterogeneous sensor networks based on the wrapper-mediator paradigm [18]. It provides a unified interface by which users can easily execute queries on the system to retrieve network information and elaborate sensor data. In SeNsIM each different network of the system is managed by a dedicated wrapper that is able to communicate with the specific underlying technology and acts as a connector for the mediator component; the mediator is responsible to properly format user requests and forward them to the different wrappers, this translates the incoming queries and injects them into the underlying networks, retrieves the results and passes them back to the mediator. The communication between the mediator and the wrappers is carried out by means of XML files, written according to a standard format and containing information about the structure of the underlying networks, the user-defined query parameters and the retrieved results. The retrieved results are then formatted according to new emerging standards on Sensors and related Observations [9].

In order to make the integration effective, each wrapper explores and monitors the local sensor networks (discovery phase) and sends to the mediator an appropriate description of the related information according to a common data model, in a struct.xml file. The mediator organizes such information and keeps a unique view of all systems in order to satisfy user or application queries.
By means of the mediator GUI, a user can specify a query by indicating the desired values and other relevant parameters such as sample period and query duration; the mediator translates user requests and builds a query, sending it to the wrappers responsible of the target network/s. The retrieved results are collected and written in a result.xml file by each involved wrapper, and sent back to the mediator, which encodes and stores sensed data into its database in order to make them available for elaboration or data fusion. In order to reduce the TCP communication overhead, a *retrieval interval* can be specified by the end user, identifying the time interval by which the wrappers may collect query results and send them back to the mediator (in a single result.xml file), thus not having to wait until the query duration elapses.

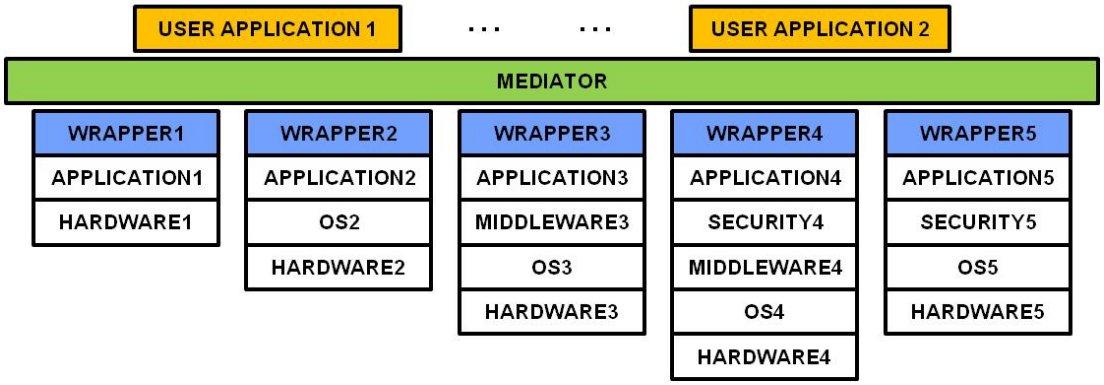

Fig. 3. The extended SeNsIM-SEC architecture

The retrieval interval can be tuned in order to control the trade off between system performances and the acceptable delay in obtaining the results from wrappers.
As illustrated, the SeNsIM architecture is able to de-couple different networks and manage them in the same way: this enables us to adopt it to design an heterogeneous sensor network infrastructure where the heterogeneity is not only in the technology aspects but also in the different security requirements. At this aim we are working on the extension of SeNsIM to cope with this new security features (SeNsIM-SEC) and developed ad-hoc wrappers matching the underlying security mechanisms and protocols, too. In Figure 3 the SeNsIM-SEC architectural model is represented, showing several types of sensor networks to be integrated, with their different features in terms of hardware platform, operating system, middleware and security requirements.

In the reminder of this paper, we will consider the scenario on the right of Figure 3 (denoted as wrapper5) where there is not an intermediate middleware that manages the sensor network, and security mechanisms can be directly enforced on the top of the sensor operating system by implementing proper security components to meet confidentiality, integrity and authentication requirements. At this aim, a specific wrapper was designed to communicate with sensor nodes via low level interfaces exploiting operating system primitives.

We have considered one of the most adopted OS for sensors, TinyOS [15], installed on each simple motes (sensor nodes) of the network and on the sync node (master node) that acts as a connector towards the application level, forwarding the incoming queries to the motes and returning results back to the querying wrapper.

In order to provide the integration in SeNsIM-SEC, we have implemented a flexible wrapper able to interface with any TinyOS-based network. The security requirements of the sensor network have been achieved by implementing a hybrid cryptosystem based on the ECC primitives for key agreement operations and digital signature generation and verification. In order to realize such operations we have implemented two different applications to be run respectively on the master and the motes nodes, based on the operating system components and on the WM-ECC library primitives. Further details on the cryptosystem and the flexible wrapper component are given in the following section.

IV. The System Implementation

In this section we will give some implementation details about the cryptosystem adopted to secure our reference sensor network and the wrapper developed in order to integrate such network into the SeNsIM framework.

A. The WM-ECC library and the developed cryptosystem

Our cryptosystem is based on the WM-ECC library [16], a publicly available open source implementation of a 160-bit ECC cryptosystem targeted to MICAz, TelosB and Tmote Sky platforms, based on recommended 160-bit SECG elliptic curve parameters [secp160r]. Fundamental ECC operations are based on large integer arithmetic operations over finite fields as multiplication, division and modular reduction; in order to improve the performances of encrypting/decrypting operations, authors of WM-ECC library have exploited several optimizations and implemented parts of such operations directly in assembly language, in order to have a complete control of the register utilization.

WM-ECC provides support for all the ECC operations and gives an optimized implementation of the ECDSA protocol for digital signature generation and verification, relying upon techniques such as sliding-window and Shamir trick; it does not provide an implementation of the ECDH protocol, which we have supplied by exploiting the basic ECC primitives and the main TinyOS components.
WM-ECC has been proved to be more computationally efficient than its major counterparts like TinyECC and EccM2.0: for example, on MICAz platform, TinyECC is 42% slower in signature generation, and on TelosB platform, the performance gap increases to 180%.

From an implementation point of view, WM-ECC is composed of 3 modules:

- Bint - provides optimized subroutine for large integer operations;
- ECC - based on the Bint module, implements all ECC operations;
- ECDSA - provides digital signature generation and verification primitives.

The WM-ECC library has been exploited in order to:

- implement the ECDH protocol for achieving a common secret key to be used for establishing a secure communication channel between the master and each of the motes;
- provide digital signature generation at the master side and verification at the mote side, by using the ECDSA primitives;

The secret shared key achieved via the ECDH protocol is used as a symmetric key for encrypting and decrypting the messages exchanged between the master and the motes, by exploiting the Skipjack cypher, with 80 bit keys and 64 bit blocks. The cypher has been used in such a way that the motes encrypt the result messages' payload after verifying the signature sent by the master, and the master decrypts such messages in order to get the results and forward them via the UART interface to the overlying application.

Let's first consider the implementation of the key agreement protocol. In order to implement the ECDH protocol the EcdhC component has been added to the WM-ECC library: this component uses the key_agree function implemented in the EcdhM module and accessible via the Ecdh interface; it is connected to the EccC component, providing all the ECC operations, and to the SHA1M_ncsu module, which provides the implementation of SHA-1 used by the Key derivation function (KDC).
The main steps performed by the protocol are the following:

1) after an initialization phase, the master generates a random key KM in the [1,n-1] interval, and performs a scalar product to calculate its own public point Q1=KM*P, where P is the base point on the curve.
2) at the same time the mote, after an initialization phase, generates a random key Km in the [1,n-1] interval, and calculate its own public point Q2=Km*P.
3) the master inserts Q1 into a message and sends such message to the mote.
4) the mote receives the message from the master, extracts Q1 and performs a scalar product in order to obtain the secret S=km*Q1; then it sends its public point to the master in a message.
5) the master receives the message from the mote, extracts Q2 and calculates the shared secret S=KM*Q2.

As for digital signature operations, we have used the ECDSA primitives in order that:

1) the master node, at the arrival of a query from the UART interface, constructs a query packet with the received parameters, digitally signs it and then broadcasts it to the motes via the radio channel;
2) when receiving a query packet, a mote first verifies the digital signature and, if it turns to be successful, starts to sample the required physical values according to the query parameters; the retrieved results are collected and inserted into the payload of the response packet, which is previously encrypted before being sent back to the master with the secret key obtained after the ECDH protocol.
3) the master receives the response packet, extract its payload and decrypts it by using the shared secret obtained after the ECDH protocol. Then, the master returns the result values to the querying wrapper through the UART interface.

The encryption/decryption operations are carried out by means of the Skipjack cypher, realized by a custom TinyOS component implementing the BlockCypher TinyOS interface; such cypher uses the key derived by the ECDH protocol as the cryptographic key for encrypting the communication channel between the master and the motes. In order to reduce the overhead resulting by the cryptographic operations, we have chosen to encrypt only response packets sent by the motes to the master, and let only the master do decrypting operations to get the requested data.

Finally, we developed two different applications, respectively for the master and the mote side. The master application has been implemented so that it starts the ECDH protocol (step 1) when a timer expires: at the system setup a timer starts to run, and after 5 seconds an event is generated, handled by the master application which will calculate the master's public point and send it to the motes. The master application has been configured in order to digitally sign outgoing query packets addressed to the motes and decrypt the incoming response packets before sending the results to the wrapper. The mote application in turn, has been configured in order to be able to perform the ECDH protocol initiated by the master, verify the digital signature of the incoming query packets, and encrypt all outgoing response packets.

B. A flexible wrapper for secure networks

In order to introduce security management in the SeNsIM architecture we have developed a flexible wrapper able to connect a secured network to the whole system. Each wrapper is able to discover the network topology through some information retrieved about its nodes and, above all, it is able to access sensor data by querying single sensors, groups of nodes or the entire network. In particular, it can perform the following tasks:

- classification - discovery, extraction and management of information about sensors and local network;

- query processing - management of the mediator queries sent to the local network;
- communication - management of the communication process with the mediator;

As we have shown in the previous section, our secured network runs TinyOS opportunely modified by the introduction of security primitives based on the WM-ECC library. In order to connect the SeNsIM system with such network, we have developed a wrapper component interfacing with the master node by means of a C application which communicates with the sensor nodes by exploiting the serial forwarder tool; thanks to its generality, our wrapper can be adapted to any network which is not equipped by a middleware layer providing a high level programming abstraction, by just customizing the modules which realize functions depending on the specific network interfaces.

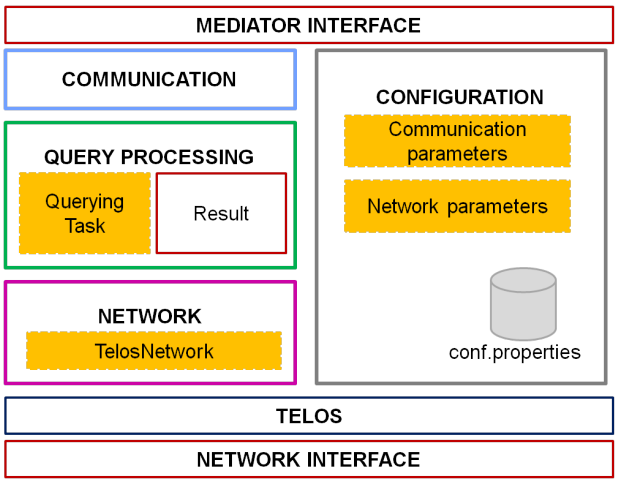

Fig. 4. The wrapper architecture

Figure 4 shows the architecture of our wrapper, made of several components between the mediator and the network interfaces.

- communication component: is responsible of managing wrapper-mediator communication, carried out through socket interfaces.
- query processing component: is responsible of performing query execution and result retrieval; this component can be configured by properly specializing the wrapper Query interface; the querying function is implemented by the *inject* method, which sends a query to the network, prepares a query packet and sends it to the master node via UART; it also collects results formatting them according to the structure expected by the overlying components.
- network component: this component can be configured by properly specializing the Network interface, which describes some network parameters and implements the discovery function, necessary to define the topology on the basis of information as node IDs, parent IDs and node depth in the routing tree.
- configuration component: is responsible of retrieving some communication and network parameters from a configuration file; it allows for a flexible reconfiguration of the wrapper component when it has to be adapted to different networks.
- platform component: specifies the values which can be retrieved by the employed sensor platforms.

The introduction of security requirements in SeNsIM is quite easy, once the protocols have been designed; nevertheless it opens new possible scenarios of adoption of sensor networks that can introduce a huge overhead in terms of performance and sensor life (read battery consumption) into the whole architecture. In next section we will discuss about the feasibility and overhead of our solution.

V. Security and Performance Analysis

The introduction of security mechanisms within the sensor networks is a desirable feature but it may introduce a very heavy overhead that must be addressed by any sensor networks developer and deployer. The analysis we conducted in this section aims at analyzing the security level introduced in the system, the latency introduced in the whole monitoring architecture, the overhead introduced by the protocol and, finally, the resource occupation to elaborate cryptographic functions on single nodes. This information, in fact, should be taken into account by any designer to meet his security and performance requirements.

Our testbed is shown in Figure 5: it is made of two sensor networks, both composed of Telosb motes, based on a 4.15 MHz MSP430 microcontroller and a CC2420 radio chip and having a 10 kBytes internal RAM and a 48 kBytes program Flash memory. The two sensor networks have different security requirements: the former is configured to work with the proposed secure protocols while the latter can work with the same applications deprived of any security mechanisms. Each sensor network relies upon a master node which is directly connected to a PC device running the wrapper software; each wrapper is connected to the mediator component via a TCP/IP network.

To analyze the performance, we have carried out several experiments by sending different queries to the underlying networks through the mediator interface. In order to estimate the overhead on the average response time of SeNsIM-SEC, we considered the sum of different delays (as shown in Figure 6) :

- T1: the mediator creates the query.xml file and forwards it to the wrappers through the TCP/IP network;
- T2, T3: the wrapper receives the query.xml file and creates the query packet to be forwarded to the master of the underlying network through the UART interface;
- T4: the sensor network manages the received query and returns results to the querying application;

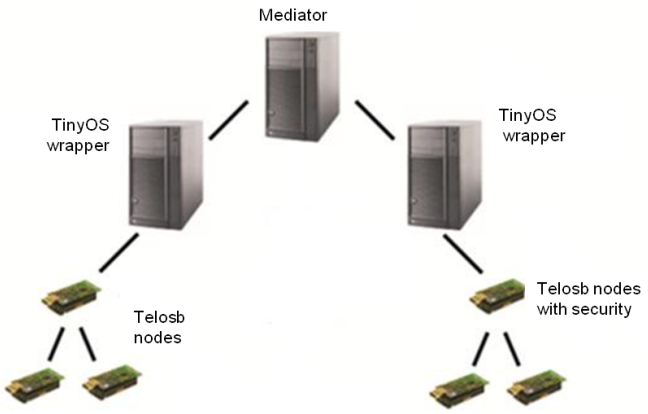

Fig. 5. The adopted testbed

- T5: the wrapper collects results belonging to the first retrieval interval and writes them in a result file; then it forwards such file to the mediator through the socket interface;
- T6: the mediator receives the first result.xml file, retrieves the contained values and stores them into its database;

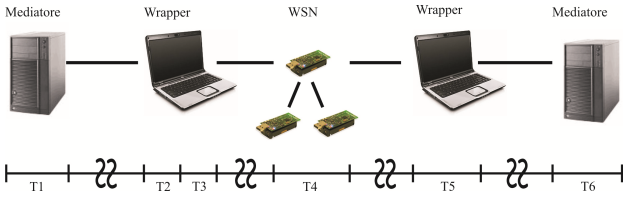

Fig. 6. The measured delays

In this analysis, we did not consider the delays related to wrapper-mediator communications over the TCP/IP network (as they depend on how the application is distributed to monitor a particular environment) and we just considered a one level deep tree sensor network topology; indeed, these choices may heavily affect the total response time but, in this experiments, we just want to measure the overhead introduced bu the security protocols and not the total response time.

The proposed cryptosystem introduces an almost constant delay of about 4 seconds respect to the network without security. So, the cryptographic operations performed at the master and mote sides for encrypting/decrypting packets produce a fixed latency in returning the first results to the wrapper for each executed query and this can certainly be acceptable in all monitoring applications where real-time is not a mandatory requirement. With ECC, in fact, we are able to guarantee the same security level provided with RSA protocols (in [4] it is stated that to provide equivalent security to 1024-bit RSA, an ECC scheme only needs 160-bit key size) but with less latency.

As for the packet overhead, in order to make the security application possible, we had to increase the payload length from the default 29 bytes size to 80 bytes, in order to allow the transmission of the public points and digital signatures. Such an overhead is very crucial as this also has a bad impact on battery consumption that is very high during the transmission phase; for this reason, it has been reduced by performing a little variation in the protocol and considering two different TinyOS packets, one for carrying cryptographic information, and the other for the normal communication.

Finally, as for resource allocation, we expected that encryption and authentication operations performed by our cryptosystem came at an additional cost in terms of memory and CPU. In fact, the secure query application implies a higher RAM usage both on master (+ 81%) and mote side (+ 62%), compared to that of the simple query application without security; similarly, ROM usage turns out to be 54% and 44% higher then the not secured case. The total amount of program space and primary memory consumed by the query application (26KB on master side and 31KB on mote side of ROM, 2KB of RAM usage) is very low respect of the adoption of RSA-based protocols and, furthermore, it can be considered acceptable if we assume to deploy sensor nodes that just run a monitoring application at a time.

Up today, we did not considered the power consumption and sensor battery life associated with our proposal, yet. Nevertheless, we think that these results are very promising especially if compared with the adoption of RSA-based protocols that require longer keys and longer elaboration processing times as reported in [17]; as a direct consequence of measured parameters, we also expect positive results in power consumption.

VI. Conclusion and Future works

In this paper we addressed the security issues arising in a complex distributed monitoring infrastructure, focusing in particular on the transport level, responsible of the node-to-node communication in a sensor network. We proposed an hybrid cryptosystem relying upon the WM-ECC library in order to ensure confidentiality, integrity and authentication requirements: such cryptosystem implements key exchange mechanisms (through the implementation of the ECDH protocol), end-to-end encryption and broadcast authentication of query messages sent from the master to motes. We also proposed an extension of the SeNsIM integration platform, namely SeNsIM-SEC, in order to make it capable to manage networks with different security requirements by means of the development of a flexible wrapper component. We integrated our secured network into SeNsIM-SEC and carried out an evaluation of the overall system performances, in terms of resulting average response time, memory usage and packet length. Such evaluation showed the platform scalability and extensibility features, and highlighted the unavoidable introduction of an overhead due to cryptographic operations, which

resulted quite acceptable in the context of the overall system. As a future development, we plan to extend our cryptosystem implementation to other sensor platforms, such as Tmote Sky, as well as to exploit different block cyphers with longer keys for the communication channel encryption , in order to achieve a higher degree of security. Moreover, we plan to make a comparison of the proposed cryptosystem with other major solutions, such as TinySec, Minisec etc, by exploiting the SeNsIM-SEC framework, through their integration in the whole architecture in order to derive design criteria on the basis of the performance and security trade-off.

References

[1] F. Amato, V. Casola, A. Gaglione, A. Mazzeo,*A semantic enriched data model for sensor network interoperability*, In Simulation Modelling Practice and Theory, 2010, (in Press) Elsevier.

[2] T. Brooke, J. Burrel, and T. Beckwit, *Vineyard computing: Sensor networks in agricultural production*, In Pervasive Computing Magazine, pages 3845. IEEE, March 2004.

[3] V. Casola, A. Gaglione, and A. Mazzeo, *A reference architecture for sensor networks integration and management*, 2009. In IEEE Proceedings of GSN09, Oxford, July 2009.

[4] Certicom Research, *Standards for efficient cryptography*, SEC 1: Elliptic Curve Cryptography", Version 1.0, September 20, 2000. Certicom Research. Standards for efficient cryptography.

[5] A.De Benedictis, A. Gaglione and N. Mazzocca, *A Secure Architecture for Re-Taskable Sensing Systems*, In Journal of Information Assurance and Security, Volume 6 , pp. 240-247, 2011.

[6] Y. Hu D. Li, K.Wong and A. Sayeed,*Detection, classification, and tracking in distributed sensor networks*, In IEEE Signal Processing Magazine, pages 1729. IEEE, March 2002.

[7] C. Karlof, N. Sastry, and D. Wagner, *TinySec: A Link Layer Security Architecture for Wireless Sensor Networks*, In Proceedings of the 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys 2004), Baltimore, MD, November 2004.

[8] M. Luk, G. Mezzour, A. Perrig, and V. Gligor, *MiniSec: A Secure Sensor Network Communication Architecture*, In Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN 2007), April 2007.

[9] Open Geospatial Consortium Inc., *Sensor Web Enablement: Overview And High Level Architecture*, In Simulation Modelling Practice and Theory, 2010, (in Press) Elsevier.OGC OpenGIS White Paper,2007

[10] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler,*Spins: security protocols for sensor networks*, In ACM Wireless Networking, VIII (5), pp. 521-534, 2002

[11] R.L. Rivest, A. Shamir, and L.A. Adleman, *A method for obtaining digital signatures and public-key cryptosystems*, In Communications of the ACM XXI (2), pp. 120-126, 1998.

[12] L. Schwiebert, S. K. S. Gupta, and J. Weinmann, *Research challenges in wireless networks of biomedical sensors*, In Proceedings of ACM/IEEE MOBICOM 01, pp. 151 -165, 2001.

[13] D. Steere, A. Baptista, D. McNamee, C. Pu, J. Walpole,*Research challenges in environmental observation and forecasting systems*, In Proceedings of ACM/IEEE MOBICOM 00, Boston, August 2000.

[14] J. Tavares, F. J. Velez, J. M. Ferro, *Application of Wireless Sensor Networks to Automobiles*, In Pervasive Computing Magazine, pages 3845. IEEE, March 2004.

[15] *TinyOS Project*, URL: http://www.tinyos.net.

[16] H. Wang, B. Sheng, C.C. Tan and Qun Li, *WM-ECC: an Elliptic Curve Cryptography Suite on Sensor Motes*, Technical report, Oct. 30, 2007

[17] H. Wang and Q. Lin, *Efficient Implementation of Public Key Cryptosystems on Mote Sensors*, In Proceedings of the 8th International Conference on Information and Communications Security (ICICS 2006), Raleigh, North Carolina, USA. December 2006, pp. 519-528.

[18] G. Wiederhold, *Mediators in the Architecture of Future Information Systems*, In IEEE Computer XXV(3), pp. 38-49, 1992.

[19] ZigBee Alliance, *Zigbee specification*,Technical Report Document 053474r06, Version 1.0, ZigBee Alliance, June 2005.