Securing a Tiered Re-Taskable Sensing System

Alessandra De Benedictis, Andrea Gaglione, Nicola Mazzocca
Department of Computer Science and Systems
University of Naples Federico II
Naples, Italy

Abstract—Sensor Networks are widely used in several application domains thanks to their data acquisition and data processing capabilities. They are well suited to a multitude of monitoring and surveillance applications and are often involved in mission-critical tasks, thus making security a primary concern. Many architectures and protocols have been proposed to address this issue, mainly based on cryptographic operations, but it still represents an open research area: such techniques in fact, to be effective, often require complex computations and a large amount of dedicated resources, which are not available on sensor platforms according to the existing technology. Nevertheless, if considering tiered sensor networks, where tiny motes coexist with more powerful nodes, it is possible to perform some complex and efficient security schemes by exploiting the different capabilities of such nodes. In this paper we present an secure architectural proposal of the Tenet system, a tiered re-taskable sensor network architecture. Specifically, we have integrated some security library into the Tenet architecture in order to implement a hybrid cryptosystem. The latter combines symmetric and asymmetric cryptographic schemes to benefit of the security provided by asymmetric protocols and the better performance of symmetric ones.

Keywords: Sensor network security, Secure communication architecture

I. INTRODUCTION

The increasing spread of sensor networks has led to the diffusion of middleware platforms as well as sensor network programming systems, which aim to bridge the gap between applications and the underlying hardware platforms. These systems provide high level programming abstractions and implement services such as routing, transport, task dissemination and execution and time synchronization, thus simplifying application development. Tenet [1], [2] is an example of such systems and the validity of its architecture has been demonstrated in several application domains [3], [4]. The Tenet architecture has been conceived for tiered sensor networks consisting of two classes of devices: motes are in the lower tier, which enables flexible deployment of dense instrumentation, while less constrained 32-bit nodes (which we call masters) are in the upper tier and implement multi-node data fusion and application logic.

One of the main open issues in such kind of systems is related to the development of general purpose security protocols. There is a large number of application scenarios where data exchanged between sensor nodes is critical (e.g. health or military applications), and providing security services for such applications is a technical challenge, due to hostile deployment environments and resource limitations. In fact, the openness of wireless channels lets anyone be able to monitor or participate in communications, undermining integrity and confidentiality requirements of the system. Furthermore, unreliable communication and unattended operations (i.e. physical attack or manumission), together with resource and computing constraints, make it difficult to directly employ the existing security approaches to the area of wireless sensor networks.

Most security protocols are based on cryptographic operations as encryption and authentication; they massively involve the adoption of keys and complex mathematical functions that require dedicated computational resources. Indeed, the adoption of such security mechanisms on so small devices can be critical from a performance and power consumption point of view. At this aim, in this paper, we discuss the design and implementation of a hybrid cryptosystem that combines symmetric and asymmetric cryptographic schemes to benefit of the security provided by asymmetric protocols and the better performance of symmetric ones. We also exploit the advantage of having different computational and energy constraints between a mote and the base station [17] which is even more effective in a tiered system like Tenet.

The reminder of the paper is structured as follows: in Section 2 a description of security requirements and open issues in sensor networks are discussed. In Section 3 we will give a brief overview of the Tenet architecture, while in Section 4 we will illustrate our proposal as well as our aims. In Section 5 we will describe our security enhanced Tenet architecture and also give some implementation details. Finally, in Section 6 some conclusions and future work will be drawn.

II. SECURITY IN SENSOR NETWORKS

Recently there has been an intense research aimed at developing security schemes for sensor networks applications.

On the basis of the Dolev-Yao threat model [5], an attacker can spoof, intercept, alter and inject any message exchanged between sensor nodes. According to that, main requirements of secure sensor network architectures are authentication, confidentiality, integrity, and freshness.

As previously said, cryptography is the basic method to implement security and both symmetric and asymmetric
schemes have been investigated for their application in sensor networks. Several implementations of Symmetric Key Cryptography (SKC) algorithms have been proposed in literature (i.e. Skipjack, DES, 3DES, AES, RC5, RC6) and nowadays some implementations of complete secure protocols based on symmetric schemes are available (TinySec [6], MiniSec [7], ZigBee [8], SNPEP [9]). Unfortunately, Tinysec, Zigbee, and SNEP, the latter being part of the SPINS protocol suite, are unable to ensure low energy consumption while simultaneously providing the above mentioned security properties. MiniSec provides a better trade off between achieved security level and energy consumption while requiring less packet overhead.

Main drawback of SKC architectures is that key management is a fundamental concern. There has been a substantial amount of research on key distribution schemes [10]-[15], but often they are not scalable, generate heavy traffic and require complex architectures. On the other hand, asymmetric schemes of Public Key Cryptography (PKC) allow for flexible key management but require a significant amount of computation. The use of PKC in sensor networks has been usually considered as “nearly impossible”, but at present some studies have demonstrated that with careful design, the Elliptic Curve Diffie-Hellman (ECDH) key agreement technique [17], can be deployed on even the most constrained of the current sensor network devices [16], [20], [21], [22]. ECDH is based on Elliptic Curve Cryptography (ECC) [17], which is the top choice among various PKC options due to its fast computation, small key size, and compact signatures.

An important security requirement which arises within the sensor network domain is the broadcast authentication. In the two-party communication case data authentication can be achieved through a purely symmetric mechanism making use of a Message Authentication Code (MAC). However this is insecure in broadcast communication scenarios. In fact, anyone of the receivers knows the MAC key and could impersonate the sender. Instead, asymmetric schemes are the natural way for providing broadcast authentication. Despite that, Perrig et al. propose a key-chain distribution system for their µTESLA secure broadcast protocol, part of the SPINS system [9]. The basic idea of the µTESLA system is that it constructs authenticated broadcast from symmetric primitives, but introduces asymmetry with delayed key disclosure and one-way function key chains. One of the limitations of µTESLA is that some initial information must be unicast to each sensor node before authentication of broadcast messages can begin. To face with these constraints enhancements to the µTESLA system have been proposed [24], [25]. However all of these schemes use symmetric key techniques with an elaborate design to add asymmetric properties to them and require loose time synchronization between nodes. Broadcast authentication is naturally achieved through asymmetric schemes. They do not need time synchronization and make use of digital signature which associates a message with an entity. Elliptic Curve Digital Signature Algorithm (ECDSA) [17] can be used for signature generation and verification. ECDSA is a variant of the Digital Signature Algorithm (DSA) [18] that operates on elliptic curve groups.

III. TENET OVERVIEW

The Tenet system is an architecture for tiered sensor networks which provides a high-level programming abstraction and allows applications to dynamically task and ret-task the sensor network. The Tenet architecture is motivated by the observation that future large-scale sensor network deployments will be tiered, consisting of motes in the lower tier and masters, relatively unconstrained 32-bit platform nodes, in the upper tier [27]-[29]. The Tenet project's guiding architectural principle asserts that multi-node data fusion functionality and complex application logic should be implemented only on the masters, while allowing motes to process locally-generated sensor data. This simplifies application development, allows mote-tier software to be reused and can result in significant communication energy savings.

In Tenet applications run on one or more master nodes. All communication to the mote tier consists of tasks, and all communication from the mote tier consists of task responses (such as sensor data) destined for a master and delivered to the application program. The latter can then fuse the results, re-task motes or trigger other sensing functionalities. More than one application (multiple tasks) can run concurrently on Tenet. Applications specify a task as a linear data flow program consisting of a sequence of tasklet implementing such functionality as timers, sampling, data compression, thresholding, statistical operations, and other forms of simple signal processing. For example, to construct a task that samples the temperature sensor every minute and sends the samples to its master, an application should construct the following task:

\[
\text{periodic(1 min)} \rightarrow \text{sample(TEMPERATURE)} \rightarrow \text{Send()}
\]

Tenet is equipped with a networking sub-system which provides task dissemination, routing from motes to the master and end-to-end reliable transport. Finally, it provides globally synchronized timing service by using FTSP [30].

IV. PROPOSAL

As previously seen, security issues are a central concern for sensor networks, as they are often adopted in critical applications despite having many characteristics that make them very vulnerable to malicious attacks. Because of their resource constraints, it is very difficult to implement strong security algorithms on sensor platforms and there’s still much work to do to address this matter. However, if we consider a tiered system such as Tenet, whose master layer nodes are supposed to have relatively more plentiful resources, we can assume that the most complex and power consuming operations are placed on such nodes: this way it is possible for example to perform some complex cryptographic algorithms exploiting the different capabilities of network components.
Hence, our proposal is the enhancement of the Tenet architecture by means of the introduction of a cryptosystem, in order to achieve some security requirements in a tiered network. As for now we will not cover aspects such as data fusion, secure localization, secure time synchronization, secure routing and transport but we will keep such points as future works. Instead our proposal aims to ensure the following security properties:

- achieve end-to-end encryption, integrity and freshness of response packets sent by motes to the master
- implement a mechanism for key exchanging (and storing) between the master and motes in such a way that different pairs of keys are kept between each motes and the master
- achieve broadcast authentication of messages sent by a master to the motes

As for the first point (SKE operations) we have integrated the MiniSec architecture with the Tenet system, while PKC protocols (key exchanging and broadcast authentication) have been implemented by exploiting the TinyECC library [20]. TinyECC is a publicly available software package for ECC operations and includes some optimization features which can be enabled/disabled through opportune software switches. The key exchanging protocol is naturally achieved itself via Tenet tasking system, while as for broadcast authentication the only constraint is that each mote has to be preloaded with the public key of the base station. This is slightly acceptable since a Public Key Infrastructure for sensor networks still does not exist at the moment.

Current implementation of the cryptosystem has been realized by taking into account a single master Tenet architecture, we made no assumptions on master-to-master communication but we have kept this point as a future work. In the following sections, we firstly give a brief overview of the Tenet system and describe the adopted software packages (MiniSec and TinyECC) and then illustrate the design principles of our security schemes. The latter has been implemented and tested on TelosB motes and PC-class devices with Tenet 2.0 running on top of TinyOS 1.x [26]. However a more complete evaluation of security features of our cryptosystem will be addressed in future works as well as extensions of existing ones with new components.

V. SECURITY ENHANCED TENET ARCHITECTURE

The design of the cryptosystem for the Tenet architecture focuses on exploiting low level security primitives provided by publicly available software packages. In this section, we first give an overview of the adopted tools, and finally illustrate the design as well as some implementation details of our cryptosystem and its integration with the Tenet architecture.

A. Adopted Technologies

MiniSec is a secure network layer that provides a high security level while keeping low energy consumption. It provides data confidentiality, integrity and freshness. Minisec’s source code is publicly available for Telos motes, but can be easily ported to other platform. It has two operating modes, one tailored for single-source communication (unicast communication), and another tailored for multi-source broadcast communication. Both schemes employ OCB [31], or Offset CodeBlock as encryption mode ensuring authenticated encryption (confidentiality and integrity) and a counter as a nonce (freshness). Authors rewrote part of the TinyOS network stack, specifically the Active Message layer (GenericComm and AMStandard) in such a way all outgoing messages are encrypted, while all received packets are decrypted. MiniSec uses 80-bit symmetric keys, considered to be secure until 2012. When 80-bit keys become insecure, it will be possible to use 128-bit AES keys, secure for the next 20 years. Minisec’s packet format is based on the current TinyOS packet header for Telos mote’s CC2420 radio. The net overhead of a MiniSec packet is 3-byte increase over a standard TinyOS packet.

TinyECC is a configurable library for ECC operations in wireless sensor networks. Its primary objective is to provide a ready-to-use, publicly available software package for ECC-based PKC operations that can be flexibly configured and integrated into sensor network applications. TinyECC includes all the well-known ECC schemes, such as ECDH key agreement scheme and ECDSA digital signature scheme. It also includes a public key encryption scheme (ECIES) and some optimization features for ECC operations, which can be enabled/disabled by developers by means of apposite software switches. TinyECC has been tested on MICAz, TelosB, Tmote Sky, and Imote2 platforms running TinyOS. By default, TinyECC includes all 128-bit, 160-bit and 192-bit ECC parameters recommended by SECG (Standards for Efficient Cryptography Group) [17], with the 160-bit ECC having the same security level as 1024-bit RSA.

B. Design and Implementation Details

Figure 1 shows the security enhanced Tenet architecture, having been realized form the current Tenet prototype. Red dashed lines indicate new modules added to the system as well as extensions of existing ones with new components.

As for key establishment, we have implemented the ECDH key agreement protocol by exploiting the Tenet tasking system and TinyECC primitives. In a key establishment scenario the master sends to each mote the following task:

![Figure 1. Modified Tenet stack on (a) master side and (b) mote side](image-url)
pubKeyExchange(PPx, PPy) → Send()

where pubKeyExchange is a new tasklet added to the Mote Tasking Library that aims to perform ECC security operations according to the ECDH key agreement technique. On master side, we ported TinyECC code from nesC to C, by constructing the ECC Library exploited by the master in order to let a master node perform ECC security operations. Also, we added the pubKeyExchange element to the Tasking Library on master side to let the Tenet system correctly interpret a task containing the pubKeyExchange tasklet. Figure 2 illustrates the steps according to whom we have implemented the ECDH protocol:

1. the master runs the ECDH application and initialize the Elliptic Curve;
2. the master calculates its Public Point on that curve and
3. sends the previously mentioned task to the mote with the two coordinates (PPx, PPy) of its Public Point;
4. the mote initializes the Elliptic Curve and
5. calculates its public point on that curve;
6. the mote calculates the shared secret, that is its own private key shared with the master, and stores it in the MiniSec keyfile;
7. the mote sends the task response with the two coordinates of its Public Point
8. Finally, the master calculates the shared secret and stores it in the MiniSec keyfile. The master keeps a list of as many keyfiles as the number of motes.

Then, the procedure is iterated for all motes in such a way that each of them shares a different private key with the master. The calculated shared secret is a 128-bit key, however only the first 80 bit will be stored in the MiniSec keyfiles.

As for broadcast authentication, we assumed that broadcast tasking messages from master to motes must be authenticated in such a way each mote can verify the identity of the master node. Hence, we have implemented the ECDSA scheme by using again the primitives provided by TinyECC. The only constraint is that during the initialization phase of the system the master should generate a key pair (private key – public key) and store its private key in the ECC Library. On the other side, each mote should be preloaded with the public key of the master, opportunistically stored in the TinyECC Library. That assumption can be accepted since there is not yet a Public Key Infrastructure for public key distribution in sensor networks. On master side, tasking messages are signed with the master private key in the TenetAPI module and sent to motes together with the signature. On mote side the signature is verified in the TaskInstaller component with the master public key. The high modularity of the Tenet system allowed us to easily add security operations into the above mentioned opportunite elements.

Finally, as for confidentiality, integrity and freshness of task response messages from motes to the master, we have opportunistically integrated the MiniSec security layer into the Tenet system. As previously mentioned, MiniSec’s authors simply rewrote the ActiveMessage layer of the TinyOS network stack for encrypting all outgoing messages and decrypting all received ones. Since we are just interested in securing task response messages, on mote side we integrated the MiniSec AMStandard module and modified it in such a way it only does encryption of outgoing task response messages which are identified with a specific tag; on master side we added MiniSec decrypting operation into the AMFiltered component running on the base station in order that it just decrypts incoming task response messages identified with the above mentioned specific tag. Obviously, those operations are performed by using previous exchanged private keys between the master and each mote.

VI. CONCLUSION AND FUTURE WORKS.

In this paper, we have proposed the design of a hybrid cryptosystem aimed to secure the Tenet architecture. We have combined symmetric and asymmetric cryptographic schemes in order to achieve key exchange mechanisms (through the definition of a specific tasklet), end-to-end encryption, integrity and freshness of response packets sent from motes to the master, and broadcast authentication of tasking messages coming from the master to motes. These goals have been reached by opportunistically integrating the TinyEcc library and the Minisecc security layer with the Tenet architecture. We have implemented and tested our schemes for Telos motes running Telent-t1 on top of TinyOS 1.1.x. Future works will be devoted to port our code to TinyOS 2.x in order to be compliant with Tenet-t2 release as well as to port it to other sensor platforms. Finally, we plan to set up a more complete testbed for the evaluation of our schemes in terms of achieved security level, energy consumption and performances.

ACKNOWLEDGEMENTS

We thank Ramesh Govindan for having inspired and followed out this work. We also thank all members of the Embedded Networks Laboratory (University of Southern California), above all Omprakash Gnawali, Jeongyeup Paek and Marcos Vieira for their advice and collaboration.
REFERENCES


