ISSN 1726-5479

SENSORS & TRANSDUCERS

3 vol. 14-2 Special /12

Physical and Chemical Sensors & Wireless Sensor Networks

International Frequency Sensor Association Publishing

IFSA

Sensors & Transducers

Volume 14-2, Special Issue
March 2012

www.sensorsportal.com

ISSN 1726-5479

Sensors & Transducers Journal (ISSN 1726-5479) is a peer review international journal published monthly online by International Frequency Sensor Association (IFSA).
Available in electronic and on CD.

Sensors & Transducers Journal

Contents

Volume 14-2
Special Issue
March 2012

www.sensorsportal.com

ISSN 1726-5479

Research Articles

Authors are encouraged to submit article in MS Word (doc) and Acrobat (pdf) formats by e-mail: editor@sensorsportal.com
Please visit journal's webpage with preparation instructions: http://www.sensorsportal.com/HTML/DIGEST/Submition.htm

International Frequency Sensor Association (IFSA).

The 6th International Conference on Sensor Technologies and Applications

SENSORCOMM 2012

19 - 24 August 2012 - Rome, Italy

Deadline for papers: 5 April 2012

Tracks: Architectures, protocols and algorithms of sensor networks - Energy, management and control of sensor networks - Resource allocation, services, QoS and fault tolerance in sensor networks - Performance, simulation and modelling of sensor networks - Security and monitoring of sensor networks - Sensor circuits and sensor devices - Radio issues in wireless sensor networks - Software, applications and programming of sensor networks - Data allocation and information in sensor networks - Deployments and implementations of sensor networks - Under water sensors and systems - Energy optimization in wireless sensor networks

http://www.iaria.org/conferences2012/SENSORCOMM12.html

The 3rd International Conference on Sensor Device Technologies and Applications

SENSORDEVICES 2012

19 - 24 August 2012 - Rome, Italy

Deadline for papers: 5 April 2012

Tracks: Sensor devices - Ultrasonic and Piezosensors - Photonics - Infrared - Geosensors - Sensor device technologies - Sensors signal conditioning and interfacing circuits - Medical devices and sensors applications - Sensors domain-oriented devices, technologies, and applications - Sensor-based localization and tracking technologies

http://www.iaria.org/conferences2012/SENSORDEVICES12.html

The 5th International Conference on Advances in Circuits, Electronics and Micro-electronics

CENICS 2012

19 - 24 August 2012 - Rome, Italy

Deadline for papers: 5 April 2012

Tracks: Semiconductors and applications - Design, models and languages - Signal processing circuits - Arithmetic computational circuits - Microelectronics - Electronics technologies - Special circuits - Consumer electronics - Application-oriented electronics

http://www.iaria.org/conferences2012/CENICS12.html

Sensors & Transducers Journal, Vol. 14-2, Special Issue, February 2012, pp. 125-149

www.sensorsportal.com

Sensors & Transducers

ISSN 1726-5479

http://www.sensorsportal.com

A Framework for Secure Data Delivery in Wireless Sensor Networks

[1] Leonidas PERLEPES, [1] Alexandros ZAHARIS, [1] George STAMOULIS and [2] Panagiotis KIKIRAS

[1] University of Thessaly, Volos, Greece

[2] AGT Germany, Darmstadt, Germany

E-mail: alzahari@inf.uth.gr, leperlep@inf.uth.gr, georges@inf.uth.gr, pkikiras@agtgermany.com

Received: 14 November 2011 /Accepted: 20 December 2011 /Published: 12 March 2012

Abstract: Typical sensor nodes are resource constrained devices containing user level applications, operating system components, and device drivers in a single address space, with no form of memory protection. A malicious user could easily capture a node and tamper the applications running on it, in order to perform different types of attacks. In this paper, we propose a 3-layer Security Framework composed by physical security schemes, cryptography of communication channels and live forensics protection techniques that allows for secure WSN deployments. Each of the abovementioned techniques maximizes the security levels leading to a tamper proof sensor node. By applying the proposed security framework, secure communication between nodes is guaranteed, identified captured nodes are silenced and their destructive effect on the rest of the network infrastructure is minimized due to the early measures applied. Our main concern is to propose a framework that balances its attributes between robustness, as long as security is concerned and cost effective implementation as far as resources (energy consumption) are concerned. *Copyright © 2012 IFSA.*

Keywords: Security framework, Sand-boxing, Live forensics, Cryptography, Wireless sensor networks.

1. Introduction

Wireless Sensor Networks (WSN) are emerging as an important tier in the IT ecosystem where active research involving hardware and system design, networking, distributed algorithms, data management and security, is blended to deal with a unique environment with distinctive characteristics and demands. The main function of a sensor network is the utilization of tiny sensing devices which are

capable of sensing various types of incidents/parameters and communicating those with other devices. Sensor networks sensing can be applied for many applications such as target tracking, surveillance, environmental monitoring, etc. [30].

Due to the unattended environment on which wireless sensors operate and the resource constrained nature of these devices in the manner of computational capabilities, memory size and available energy, it is a major challenge to employ efficient security schemes coming from the computers or ad hoc wireless networks domain [2, 30].

In this paper, the critical security issues in wireless sensor networks are addressed, various types of threats and attacks against them are explored in order an efficient multi-tier security framework to be proposed. Furthermore, an evaluation of the combination of cryptography of the communication channel is presented along with valid sand-boxing techniques for providing protection in energy constrained embedded sensor nodes. As shown in Fig. 1, the three layers of the proposed framework are:

a) Physical Sensor Protection;
b) Sand-Boxing;
c) Crypto-Communication.

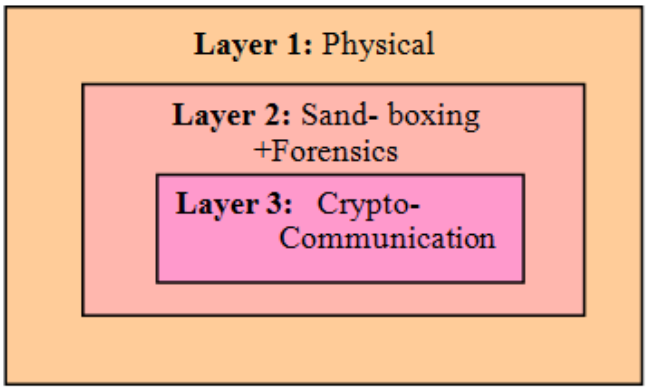

Fig. 1. Multi layer Security Framework.

Framework's primary goal is the effective blending of common security techniques such as physical security or cryptography with more modern ones like sand-boxing [1, 2, 26, 27].

The proposed protection framework is thoroughly presented along with real life use examples that prove its robustness and effectiveness against the most popular WSN security attacks. The overall concept of combining live forensics along with "sand-boxing" techniques and other commonly used security schemes as cryptography in a single framework is, to our knowledge, a unique and out of the box security attempt that can lead to an impenetrable multi-tier security framework.

The remainder of this paper is organized as follows: Section II provides a review of similar security techniques and frameworks. In Section III, we briefly explain the components on which the framework is based on. Section IV describes in details the proposed framework. In Section V, the framework's efficiency against different types on attacks in explained. Finally, Section VI some concluded remarks are presented.

2. Similar Work

Attacks on the sensor network can be classified as:
a) Physical attacks on sensor devices, e.g., destroying, analyzing, and/or reprogramming sensors.
b) Service disruption attacks on routing, localization
c) Data attacks, e.g., traffic capture, spoofing.
d) Resource-consumption and denial-of-service (DoS) attacks.

One of the serious attacks to the sensor networks deployed in an unattended environment is physical tampering with sensors. An adversary can easily capture, reverse-engineer the sensor, and deploy (multiple clones of) manipulated sensors. The compromised sensors will then be exploited by the adversary to mount actual attacks which will facilitate the subversion of the entire network.

Traditionally, the tamper-proofing of programs relies on tamper-resistant hardware [1, 2]. However, hardware-based protection will likely fail to provide acceptable security and efficiency on its own because 1) strong tamper-resistance is 'expensive' to be implemented in resource-limited sensor devices and 2) the tamper-resistant hardware itself is not always absolutely safe due to various tampering techniques [1, 3, 4].

Existing approaches to generating tamper-resistant programs without hardware support can be classified as:
a) Code obfuscation that transforms the executable code to make analysis/modification difficult [5-8].
b) Result checking that examines the validity of intermediate results produced by the program [9-11].
c) Self-decrypting programs that store the encrypted executables and decrypt them before execution [12-13].
d) Self-checking that embeds, in programs, codes for hash computation as well as correct hash values to be invoked to verify the integrity of the program under execution [12, 14, 15].
e) Software based Attestation to remotely verify the integrity of sensor software [20].

However, most of the above mentioned approaches will more likely fail on sensor networks where a program runs on slow, less-capable microcontrollers.

Software attestation is a challenge-response protocol where a verifier (e.g., base station) sends an attestation command to the attester (the node being attested) asking for certain state information as the evidence of its software integrity. Such state can be computed correctly only if the attester's system meets certain integrity requirement. After receiving the response, the verifier compares it with the known good state to check if the software at the attester has been corrupted. If a sensor node fails to give the correct answer, actions can be taken to revoke this node from the network. Several software attestation schemes have been proposed to attest the static memory regions of the software [17-20].

Physical hardening of the sensor is the first obstacle an attacker must overcome in order to tamper a wireless sensor. In order the system to be protected against attackers, passive tamper protection mechanisms, such us protective coatings and tamper seals, must used in each node. We promote the use of passive tamper protection mechanisms because they do not need any additional circuitry for their operation so that they do not consume any energy. Main goal of those mechanisms will be to make uneconomical, in manner of time and effort, to the adversaries to try to alter the behavior of captured nodes [44]. The effectiveness of the physical security on sensors is usually low and a WSN based only on physical security cannot be considered as secure. In our approach physically securing a sensor is the layer of defense mostly used to prevent less determined attackers.

Our second defense scheme strips the major functions of a live forensics check on an average system in order to match with the limited resources of a sensor, leading to the important conclusion of whether a sensor is compromised.

The live forensics security layer proposed in this paper verifies the integrity of the program residing in each sensor through a process that has been specifically designed to:

a) Prevent altering / manipulation / reprogramming of the sensor;
b) Be purely software-based;
c) Work on sensor devices with severe resource limitations;
d) The verification of the different parameters tested does not add large overhead to the communication;
e) Prevent eavesdropping attacks on the communication channel.

3. Live Forensics Framework

As the need for decentralized security emerges in large public wireless sensor networks; new application level security mechanisms aim at providing application developers with appropriate abstractions for designing the security aspects of the target software. In computer security, a sandbox is a security mechanism for separating running programs. It is often used to execute untested code, or untrusted programs from unverified third-parties, suppliers and/or untrusted users. It typically provides a tightly-controlled set of resources for guest programs to run in, such as scratch space on disk and memory. In this sense, sandboxes are a specific example of virtualization.

Zaharis et al. [26] proposed a protocol based on sandboxing technique. According to this approach, they divide isolates in two categories:

- The Security-Dedicated Isolates ("SDI");
- The Work-Dedicated Isolates ("WDI").

An Isolate Verification Server plays a key role on verifying the genuine WDIs from the malicious ones while performing all the computational and energy consuming and needy tasks. The verification of genuine WDIs is based on 1) RAM dumbing and 2) Hashing techniques. This framework uses a secure RAM dumping technique specially designed for sensors. This technique provides the framework with safer intrusion recognition while complying with the classic digital forensic techniques. The Hashing technique that is used by the framework is based on the Randomized Hashing Function [16]. This technique is used on the Work-Dedicated isolates in order to acquire highly secure tamper-proofing on sensor-resident programs. The hashing function plays a key role in the effectiveness of the proposed architecture as it is robust technique, used frequently in computer security and digital forensics due to its precision in detecting altered code.

Our goal is to improve this mechanism by enriching it with cryptographic procedures, in order to provide a secure end-to-end data delivery framework.

3.1. Cryptography

The Tiny Encryption Algorithm (TEA) is a cryptographic algorithm designed to minimize memory footprint and maximize speed. It is a Feistel type cipher that uses operations from mixed (orthogonal) algebraic groups [31]. Despite its robustness minor extensions have been published in order to present safer encryption results. In this research, we determine the weaknesses and identify the robustness of TEA, XTEA and XXTEA algorithms in wireless sensor networks and implement them in secure framework to harden security during communication [27-29].

The conditions must be met in order the algorithm to bet truly "inseparable" are:

- The distribution of keys must have been to all nodes in a secure manner;
- Each message uses a secure, unique key;
- The key generation has become with a truly random cryptographic way.

In order to generate a set of unique, truly random keys, we use the Random Number Generator service designed and operated by the University of Trinity [25]. RANDOM.ORG's source of entropy is atmospheric noise. This noise is obtained by tuning a radio to a radio frequency that no one is using. It is then played into a workstation where a program converts it to an 8-bit mono signal at a frequency of 8 KHz. Then the first seven bits are discarded and the remaining bits are gathered together. This stream of bits has very high entropy.

4. Network Architecture

4.1. Centralized Scheme

Our sensor network consists of an Isolate Verification Server (IVS) an Isolate Verification Database (IVDB) and numerous sensors which consist of an SDI and one or more WDIs (see Fig. 2). The Security-Dedicated Isolate (SDI) is the one executed on start up and conducts the forensics check of the second isolate. The 'SDI" is the one responsible for the communication with the Isolate Verification Server (IVS).

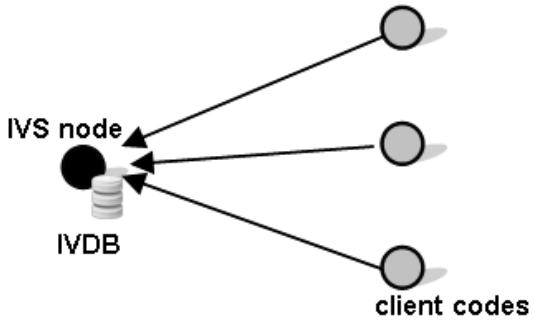

Fig. 2. The centralized scheme with an Isolate Verification Server and multiple client nodes.

The Role of the Isolate Verification Server is:

- To communicate with the SDI of every sensor in its vicinity;
- To update/manage its local IVDB;
- To act as a trusted authentication third party.

This centralized scheme is distinguished for its simplicity and is proposed for non mission critical applications with a small number of client nodes. But on the other hand, it is undesirable to equip only one IVS in a network as it becomes a single point of failure and a performance bottleneck and so the use of multiple IVSs can be facilitated over the entire network. This approach is required for performance issues but also for security reasons, e.g. in the case that an IVS gets captured by an intruder, all the critical information concerning the nodes' authentication will be available to unauthenticated users.

4.2. Decentralized Scheme – Clustering

Hierarchical schemes have shown to be more scalable and energy-aware in the context of WSN. In hierarchical architectures, nodes play different roles in the network and typically are organized into clusters. Clustering is the method by which sensor nodes in a network organized into groups according to specific requirements or metrics. Each group or cluster has a leader referred to as cluster-head and other ordinary member nodes (MNs) [43].

In our approach, we let cluster-heads in a cluster-based hierarchical architecture serve as IVSs. This allows each IVS to maintain a local IVDB that stores information of the sensors. The proposed architecture leads to a decentralized model of sensor protection where its cluster - head / IVS is responsible for its sensors.

4.2.1. Cluster scheme – Full IVDB

The Cluster – Full IVDB scheme is described in Fig. 3. Using this scheme, each local IVS is equipped with a full copy of the IVDB. The roll of each local IVS is to authenticate the sensors belonging to its own cluster. In the case of a wrong authentication, the local IVS is responsible to inform all the IVSs about the possible enemy node.

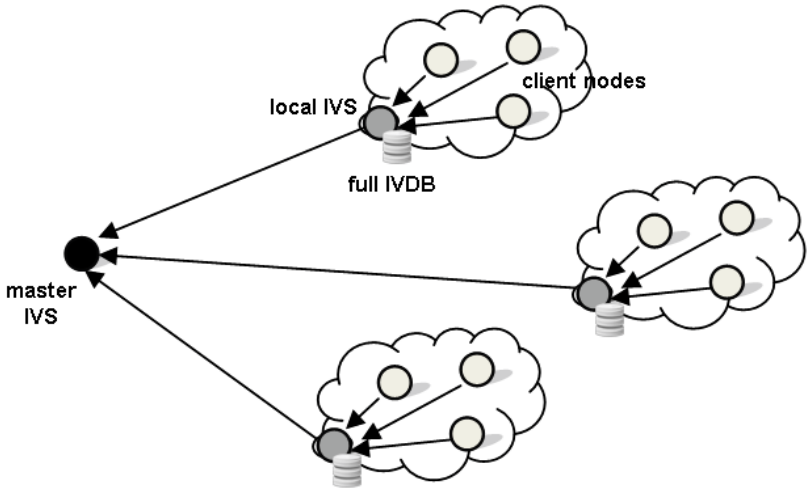

Fig. 3. A cluster scheme with local IVS as cluster-heads. Each local IVS is equipped with a full copy of the IVDB.

This scheme increases network scalability and lifetime as it solves the performance bottleneck and reduces the amount of data transmitted in the network. The authentication procedure requires data transmission only between the nodes of each cluster (local IVS – client node). The only case that a communication between different local IVS happens is when a possible enemy node is identified.

On the other hand, each local IVS must be equipped with a powerful microcontroller and enough memory in order to manage the full IVDB effectively. Also if a local IVS captured by an intruder, all the critical information concerning the nodes' authentication will be available to unauthenticated users.

4.2.2. Cluster Scheme – Distributed IVDB

Using the scheme described in Fig. 4, each local IVS is equipped with a part of the IVDB. During the network's initialization, the IVDB is divided in **N** parts, where **N** is the amount of the network's clusters. These **N** parts are transmitted to each one of the network's local IVS.

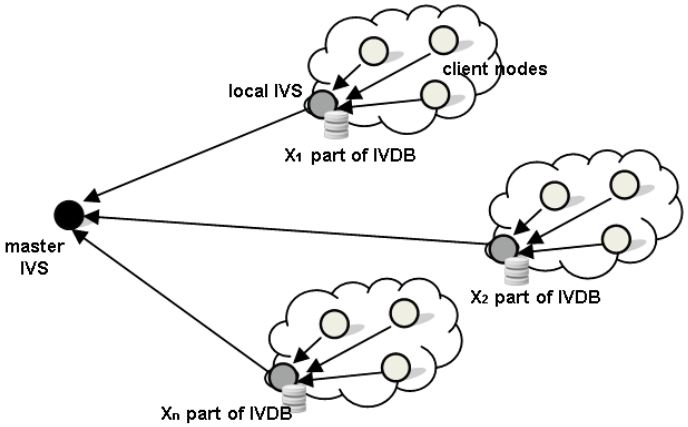

Fig. 4. A cluster scheme with local IVS as clusterheads. Each local IVS is equipped with a part of the IVDB, where 1,2,…,n is the cluster-id, X_1, X_2,…,X_n is the IVDB's part of each local IVS.

The roll of each local IVS is to authenticate the sensors belonging, each time, to its own cluster. In order to complete this procedure, it does the following steps:

1. It receives the client's authentication request.
2. The local IVS checks the IVBD's local part in order to find the requested information.
3. In case that this information is located in the local IVBD, the authentication's procedure is continued by the local IVS
4. In case that this information isn't located in the local IVDB, a request is sent to the other IVSs, in order to find the appropriate information.
5. If the requested information is located in a local IVS, it deletes its local copy of information and transmits it to the local IVS that request it.
6. Next the authentication's procedure is continued by the local IVS. In the case that an enemy node is identified, no communication between the local IVS is happened. The enemy's identification is saved in the IVDB's local part.

This scheme increases network scalability as it solves the performance bottleneck. In case of a local IVS captured by an intruder, only the local part of critical information will be available to unauthenticated users.

During the network's initialization, the IVDB's information is randomly divided between the local IVS. But after some runs of the authentication procedure, the information is distributed between the local IVSs in accordance with the sensors belonging to their own cluster. The worst case scenario of this information's spatial distribution is when a node is repeatedly moving between different local IVS. In this case, the appropriate information must transmit continuously between the respectively local IVS.

Using this scheme, the network communication between the local IVS is increased, especially during the first runs of the authentication procedure. This network communication contains critical information concerning the client's nodes authentication. In order to block an intruder to gather this

information by sniffing the communication between the local IVSs, a cryptographic communication scheme must be used. The XTEA cryptographic algorithm with a session key meets the requirements of the specific communication.

4.3. Sandboxing in Action

In order to achieve the maximum tampering protection of our sensors, sand-boxing is applied to achieve safety against malicious code execution. While more than one Security-Dedicated Isolates can run on a sensor in our proposed Framework we will use one per sensor.

On the other hand, more than one Work-Dedicated Isolates can run on a sensor performing different tasks (see Fig. 5). The verification process performed on a single WDI applies for more than one instance with the same results. Failure to verify one of the WDIs leads to locking and blacklisting of the sensors.

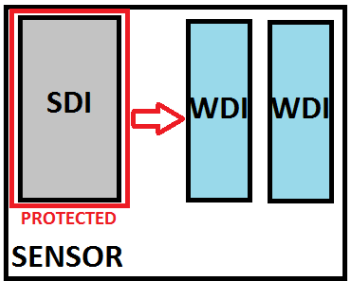

Fig. 5. The isolates on a sensor.

4.3.1. Security Dedicated Isolate

The Security-Dedicated Isolate is actually a mini forensics tool case specially created to perform live forensics in a sensor, on demand or periodically in order to specify if the sensor is compromised and react depending on the result. The Security-Dedicated Isolate has a unique id/key for every sensor, the "SDI_ID" that is used in order to communicate with the Isolate Verification Server. On the first supervised boot the SDI is the first to execute and perform a mini mapping and state validation of the sensor. These results are stored on the Isolate Verification Database ('IVDB') which resides on the Isolate Verification Server ('IVS'). As in every Digital Forensics case these data are going to be used as a proof of the sensors authenticity on the field.

From now on all data transmitted by the SDI are going to be compared to those stored on the 'IVDB' depending on the "SDI_ID". The tasks the SDI is responsible for are:

a) Communicating safely with IVS;
b) Checking the Work-Dedicated Isolates;
c) Applying countermeasures upon intrusion detection.

4.3.2. Work Dedicated Isolate

The Work-Dedicated Isolate performs the everyday tasks of a typical sensor. Due to the sand-boxing technology, more than one WDI can be executed simultaneously on a sensor, performing different tasks (see Fig. 6). Execution of non verifiable WDIs will lead to the activation of countermeasures by the SDI.

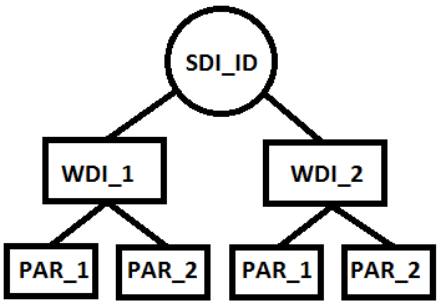

Fig. 6. More than one WDI and their check parameters.

The fingerprinting of the performance of every isolate on different parameters is stored on Isolate Verification Database along with the SDI_ID of the sensor on which the WDIs belong. The fingerprinting parameters of a WDI can depend on:

a) The hash value of the isolate;
b) The RAM dump of the isolate.

4.4. State-Transition Diagram

Each sensor device is associated with one of four states:

a) "LOCKED"
b) "VERIFYING"
c) "ACTIVATED"
d) "COUNTERMEASSURES"

When a sensor starts its execution, it is in the LOCKED state. Upon deployment a sensor device will remain in LOCKED state until it securely authenticates with IVS. No other tasks can be performed until it is authenticated.

After a valid authentication, it makes a transition to the VERIFYING state by executing the SDI verification checks. The stripped results are transmitted back to the IVS where: If the verification fails, it returns to the LOCKED state, causing the network to deny this sensor's access to the network. Otherwise, it transitions to the ACTIVATED state, in which the WDIs code is normally executed. Periodic re-verification by the SDI during ACTIVATED state can lead to LOCKED state or COUNTERMEASSURES state. COUNTERMEASSURES is the state in which a sensor is already accepted on the network and then compromised. In order to avoid denial of service attacks on which the attacker can lead all sensors to LOCK state, the COUNTERMEASSURES state can be used. In this state the compromised sensor tries to identify the type of attack on which it has been subjected through a different type of live forensics process. All other nodes ignore the compromised node through an alarm message send by the IVS. Finally it returns to the LOCKED state. The transitions between the four states are described in Fig. 7.

4.5. Authentication Protocol

The proposed protocol is consisted by three phases where certain actions must take place. These phases are divided into actions prior to deployment, during the "initialization" phase and, while in regular operation.

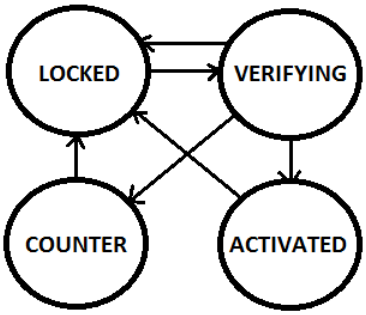

Fig. 7. State Transition Diagram.

4.5.1. Pre-deployment Phase

During the phase prior to deployment a set of random keys is generated by the base station. This set is stored to the tamper resistant storage area and it is the same for each of the network's nodes. This set will act as the key repository from where the nodes and the base station will choose their encryption keys during the operational life of the network.

The generation of the keys prior to deployment allows for significant gains in the energy consumed by the nodes, due to the fact that in order to compute a strong cryptographically key, a number of complex mathematical operations and a sequence of iterations are required, which are energy consuming and computational demanding operations.

4.5.2. Initialization Phase

After the deployment of the network the following actions are taking place:

a) Initiation: This step starts the authentication protocol between the IVS and the sensor by transmitting the SDI_ID. The sensor, after receiving the IVS_ID, asks for authentication.
   If the authentication fails the protocol is terminated.
b) If authentication succeeds, SDI is executed.
c) The result of SDI is transmitted back to IVS. IVS checks the IVDB and validates the results. The received hash value and Ram dump are checked. If it passes the test, the IVS registers the sensor in the IVDB. Then, the IVS notifies the sensors SDI of the verification result.
d) Based on the verification result, the sensor is either activated or locked. The sensor state will be changed to either ACTIVATED or LOCKED, accordingly.

Step 1 ensures sensor security, i.e., a malicious device can neither passes the authentication procedure nor has its own code executed on the sensor as far as the IVS's authentication key is kept secret from the attacker.

4.5.3. Regular Operation

After the initialization phase, the activated sensors can perform the data's collection, encryption and transmit ion to the base station. All message transactions, described to the above phases, are encrypted using the XTEA cryptographic algorithm. Each message is encrypted with a key belonged to the set of random keys deployed during the Pre-Deployment Phase of the protocol.

4.6. Verification Protocol

The verification of a sensor is based on two widely used digital forensics techniques 1) Hashing (RHF) and 2) RAM dumping per WDI (see Fig. 8).

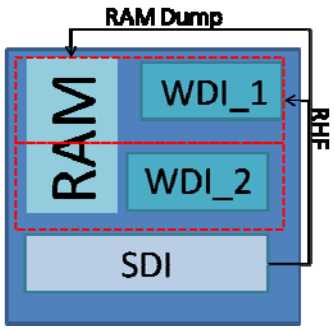

Fig. 8. Fingerprinting a WDI.

4.6.1. Hashing

Every Work-Dedicated Isolates has a unique Randomized Hashing Function (RHF) [16] which can be easily and with a minimum cost be calculated. Once calculated for every user it is stored on ISDB along with the SDI of every sensor creating the first fingerprint of the sensor. Also thanks to the fact that sensors of the same network usually perform the same task can lead to a smaller number of different hash patterns stored on ISDB per WDI.

Each WDI can be classified as being 1) common to all sensors in the network, 2) common to a group of sensors with the same missions, or 3) unique to a specific sensor.

4.6.2. RAM Dump

When using this technique, our SDI reads arbitrary RAM contents from the different WDIs running on the sensor. Every process running on a system leaves specific, well distinguished footprint on the RAM. Our goal is to create hash like footprint of the memory and store it on ISDB along with the SDI of every sensor creating the second fingerprint of the sensor. In order to keep our framework in energy efficient levels specific parts of the RAM dump are checked concerning the execution of the WDIs.

These WDI –specific fingerprints are also hashed using the Randomized Hashing Function providing an extra protection parameter.

4.7. Protocol Implementation

In order to evaluate our protocol we have implemented it on Mica2 sensor nodes [23]. The implemented protocol is based on a centralized architecture as described in section IV. The MICA2 is a third generation mote module used for enabling low-power, wireless sensor networks. It consists of an ATMega128L CPU, 4kb of Ram, 128kb of program memory and 512kb of serial flash memory and a ChipCon CC1000 radio. The Crossbow MTS310 sensor board was used which provides temperature, and other sensor types.

The protocol is implemented in two parts; the first part corresponds to IVS code and the second to sensor code (Fig. 9).

IVS Code

on receive UserHashMsg:

```
receive( idAddr , UserHashMsg );
decrypt(UserHashMsg);
if( check (UserHashMsg) == valid() ){
        VerifyMsg = Valid;
        encrypt(VerifyMsg);
        send(idAddr , VerifyMsg );
}
else{
        VerifyMsg = Invalid;
        encrypt(VerifyMsg);
        send(idAddr , VerifyMsg );
}
```

on receive HashMsg:

```
receive( idAddr , HashMsg );
decrypt(HashMsg);
if( IVDBcheck (HashMsg) == valid() ){
        VerifyMsg = Valid;
        encrypt(VerifyMsg);
        send(idAddr , VerifyMsg );
}
else{
        VerifyMsg = Invalid;
        encrypt(VerifyMsg);
        send(idAddr , VerifyMsg );
}
```

Sensor Code

on boot:

```
state = LOCKED;
encrypt(UserHashMsg);
send( broadcastAddr , UserHashMsg );
```

on receive VerifyMsg:

```
receive( idAddr , VerifyMsg );
decrypt(VerifyMsg);
if( VerifyMsg==Valid  && state==LOCKED){
        state = VERYFING;
        HashMsg= computeHashValue();
        encrypt(HashMsg);
        send( broadcastAddr , HashMsg );
}
else if( VerifyMsg==Valid  && state==VERYFING){
        state = ACTIVATED;
        start  data  process();
}
else{
        state = LOCKED;
}
```

Fig. 9. IVS & Sensor Code.

The messages sequence diagram of the aforementioned code implementation can be seen in Fig. 10.

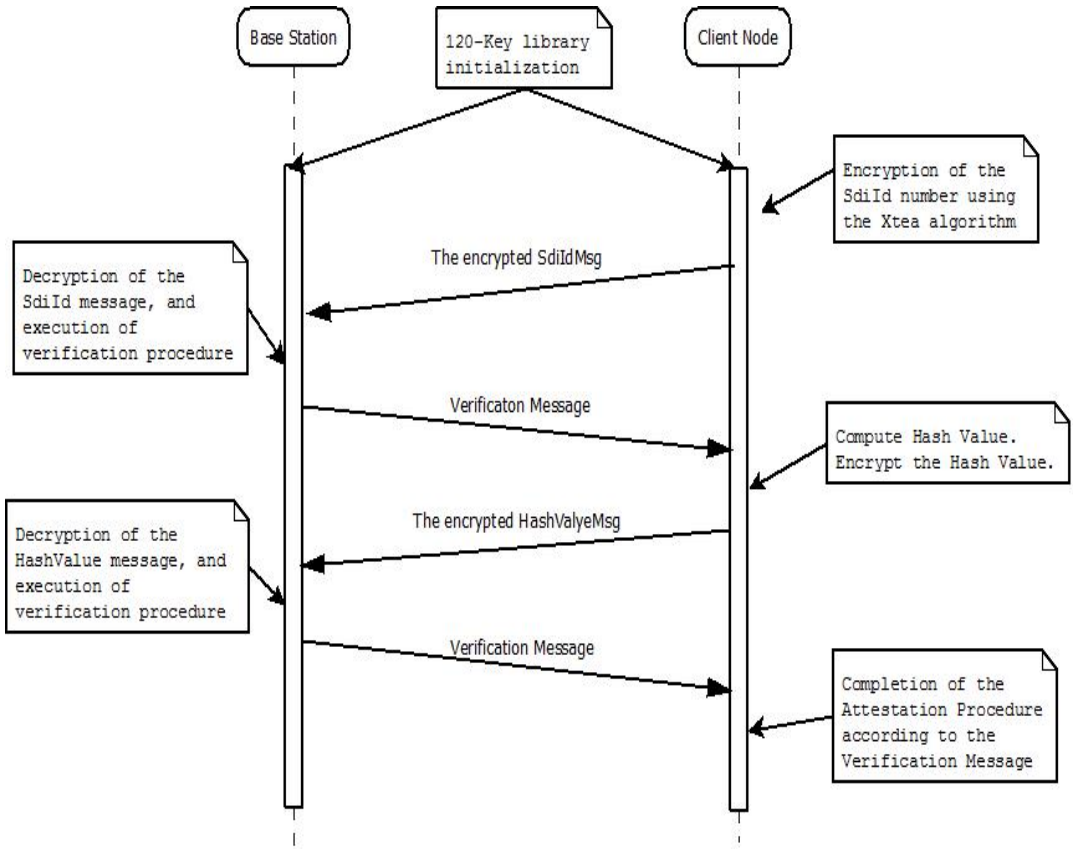

Fig. 10. Protocol Messages.

a) During the Pre-deployment Phase, each node equipped with a set of secure keys. (120 keys.)
b) The client node encrypts the SDI_ID using the XTea algorithm.
c) The client node sends the 80-bit encrypted message to IVS
d) The IVS decrypts the message and checks the authenticity of the SDI_ID (this step was simulated with 1 sec delay during simulation).
e) IVS sends to the sensor the appropriate response (valid/invalid).
f) On valid response sensors turns from locked to verifying status, initiates the hashing and RAM-dumbing procedure (during the simulation we have used the SHA-1 algorithm (~13ms/hash) [24]. For the calculation of the hash – value the algorithm utilizes 512 bytes from the memory and produces a 160 bit hash.
g) The sensor encrypts the 160-bit hash values, producing a 208-bit message. The sensor transmits the 208bit message to the IVS for validation.
h) IVS verifies the validity of the sensor's hash value. (This step was simulated with 1 sec delay during simulation).
i) IVS validates or the sensor.
j) Sensor turns status into ACTIVATED or LOCKED in accordance with IVS message.

4.8. Energy Analysis

In order to measure protocol's energy consumption we have implemented it and simulate its performance in Avrora Simulator. Avrora [22] is a set of simulation and analysis tools for programs written for the AVR microcontroller produced by Atmel and the Mica2 sensor nodes. Avrora contains a flexible framework for simulating and analyzing assembly programs, providing a clean Java API and infrastructure for experimentation, profiling, and analysis. Avrora uses the AOEN (Accurate Prediction

of Power Consumption)[21] energy consumer model. AOEN uses empirical current consumption measurements (of hardware such as the radio transceiver, microcontroller and sensors) to calculate the overall power consumption. AOEN is based on the execution of real application and OS code and measurements of node current draw, this model enables accurate prediction of the actual energy consumption of nodes. Thus, it prevents erroneous assumptions on device and network lifetime. Such a detailed prediction allows the comparison of different low power and energy aware approaches in terms of energy efficiency and the estimation of the overall lifetime of a sensor network.

4.8.1. Energy Cost of Cryptography Operations

Table 1 compares the energy consumed by the different versions of TEA cryptography algorithm. The values represent the energy consumed by a node in order to execute the following procedure:

- Encryption and sending of a 64-bit packet
- Receiving and decryption of the 64-bit packet.

Table 1. Energy cost of TEA cryptography algorithms in order to encrypt-send/decrypt-receive 64bit data (μJoule).

Algorithm	Energy Cost	
	Encryption – Send	**Receive – Decryption**
SIMPLE	6041.81μJoule	6002.94 μJoule
TEA	6067.82 μJoule	6030.77 μJoule
XTEA	6071.98 μJoule	6033.19 μJoule
XXTEA	6087.23 μJoule	6042.30 μJoule

The SIMPLE algorithm represents the procedure of sending and receiving the raw packet, without the execution of any cryptographic command.

We do not present the cost of key generation. We assume that the key is created during the pre-deployment phase, as described on section 4. As it can be seen in Fig. 11 the energy cost of using the TEA family of algorithms is relative low and comparable to no encrypted communication at all.

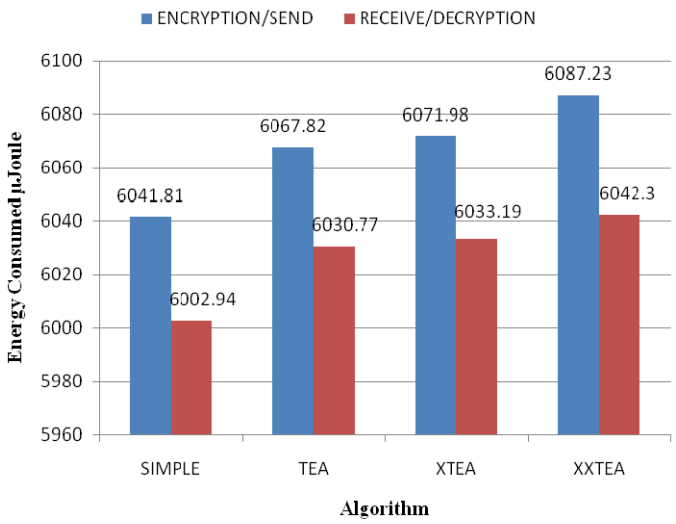

Fig. 11. Energy cost of TEA cryptography algorithms (μJoule).

4.8.2. Energy Cost of Attestation Protocol

We analyze the energy usage of the Attestation's protocol handshake procedure. Table 2 compares the energy consumed by the 2 different version of the protocol. The main difference between these versions is the encrypted message transactions that are implemented in the second protocol. As described above, in the Encrypted Attestation Algorithm we the TEA cryptographic algorithm in order to provide an end-to-end secure data delivery protocol. For our analysis we chose to focus on XTEA version of TEA's family cryptographic algorithms (see Table 2 and Fig 12 the energy cost of the attestation protocol.).

Table 2. Attestation's protocol energy cost (Joule).

Protocol	Energy Cost		
	CPU Energy	**Radio Energy**	**Total Energy**
Attestation	0.02286 Joule	0.06034 Joule	0.08321 Joule
Encrypted Attestation	0.02311 Joule	0.06094 Joule	0.08406 Joule

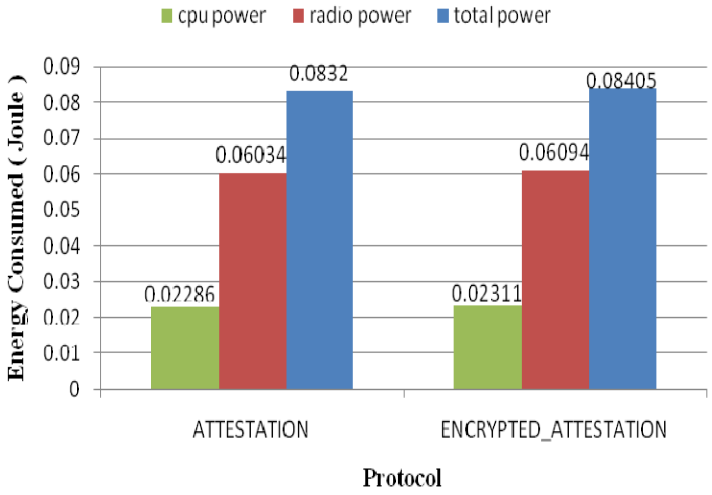

Fig. 12. Energy cost of Attestation's protocol handshake procedure (Joule).

5. Multihop Protocol

In practical WSN applications, the network radius is greater that one hop. Therefore the Attestation protocol provides multi-hop support. As described in Fig. 10, the protocol needs a two-way communication. The communication going from the base station to the nodes is called downlink. The downlink contains the verification messages. On the other hand, the communication going from the nodes to the base station if called uplink. The uplink, during the Initialization phase, contains the encrypted SdiIdMsg and the encrypted HashValueMsg. Additional, during the regular phase, the uplink contains the encrypted data messages.

In order, the attestation protocol, to support multihop communication, an appropriate routing protocol must be selected. The selected routing protocol must support the above two-way communication. The flooding algorithm stands out as the simplest solution. In this algorithm, the transmitter broadcasts the data which are consecutively retransmitted in order to make them arrive at the intended destination.

However, its simplicity brings about significant drawbacks. Firstly, an implosion is detected because nodes redundantly receive multiple copies of the same data message. Moreover, the nodes do not take into account their resources to limit their functionalities. [32] In our experiments, the flooding algorithm can be is used, as the simplest but worst case routing protocol. The only assumption the protocol makes is that every node in the network has a unique ID.

5.1. Experimental Data

In order to measure protocol's energy consumption we have implemented it and simulate its performance in PowerTossim Simulator. PowerTOSSIM is a scalable simulation environment for wireless sensor networks that provides an accurate, per-node estimate of power consumption. PowerTOSSIM is an extension to TOSSIM, an event-driven simulation environment for TinyOS applications. PowerTOSSIM includes a detailed model of hardware energy consumption based on the Mica2 sensor node platform. It provides accurate estimation of power consumption for a range of applications and scales to support very large simulations. [33]

5.1.1. Scenario 1

The experiment scenario involves 25 Mica2 deployed in a 5x5 grid in such way that each mote can communication with its neighbours only. Furthermore, the node with the smallest id (ID0) is located in the upper edge of the network as shown in Fig. 13. The node with the ID0 is the Base Station.

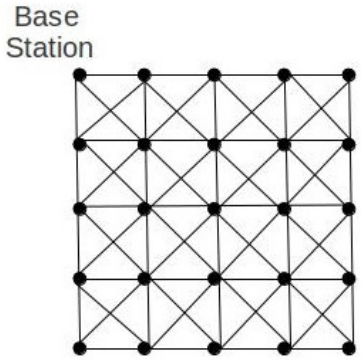

Fig. 13. The layout (5x5 grid) and links of the experimental setup. Each node can communicate with its (at most 8) neighbours.

During the experiment scenario 1, the nodes, except the base station, are in Regular Operation mode. These nodes transmit data to the base station every 5 seconds.

Table 3 compares % packet loss and the total energy consumed by the network. The values represent % packet loss and the energy consumed by the total network using:

- Different size of data packets;
- Different type of data packets (encrypted / unencrypted).

Fig. 14 represents the energy cost and the % packet loss of Attestation's protocol for different data packet size, in a graphic way.

Table 3. Attestation's protocol energy cost & % Packet Loss.

Packet Size	Energy Cost		% Packet Loss	
	Unencrypted	**Encrypted**	**Unencrypted**	**Encrypted**
16 bit	6904 μJoule	7920 μJoule	35.15	43
32 bit	7052 μJoule	7920 μJoule	40.09	42.6
64 bit	7277 μJoule	7920 μJoule	42.8	42.7
96 bit	7491 μJoule	8545 μJoule	46	49
128 bit	7717 μJoule	8545 μJoule	48	49.1

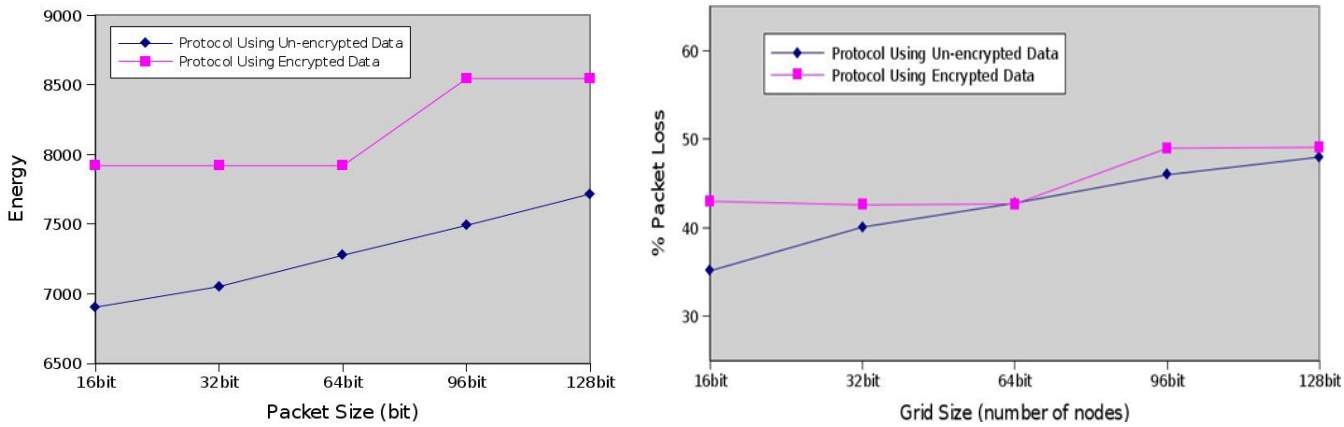

Fig. 14. Energy cost & % Packet Loss of Attestation's protocol for different data packet size.

We do not present the cost of key generation. We assume that the key is created during the pre-deployment phase, as described on section 4.

At Table 3, it is observed that at the encrypted protocol the values are the same for packet size 16, 32, and 64 bit, and another value for packet size 96 and 128 bit. This happens because of the TEA encryption function. As described on section 6.6 the TEA algorithm operates on 64-bit blocks. Therefore, even in the case that is fed with a 16 bit packet, the algorithm produces a 64 bit encrypted packet.

5.1.2. Scenario 2

The experiment scenario involves Mica2 nodes deployed in grid topology. These grid topologies differ as their size. These topologies are square grids and their size varies from 1 × 2 to 6 × 6. Furthermore, the node with the smallest id (ID0) is located in the upper edge of the network as shown in Fig. 13. The node with the ID0 is the Base Station. During the experiment scenario 2, the nodes, except the base station, are in Regular Operation mode. These nodes transmit data to the base station every 5 seconds. The size of these data packets are 64 bit.

Table 4 compares the % packet loss and total energy consumed by the network. The values represent the % packet loss and energy consumed by the total network while varies its size.

Fig. 15 represents the energy cost and the % packet loss of Attestation's protocol for different grid size, in a graphic way.

Table 4. Attestation's protocol energy cost and & % Packet Loss for different grid size.

Packet Size	Energy Cost		% Packet Loss	
	Unencrypted	Encrypted	Unencrypted	Encrypted
1×2	3330 µJoule	3330 µJoule	0	0
2×2	3464 µJoule	3561 µJoule	1.7	1.7
3×3	3988 µJoule	4338 µJoule	8.18	9.63
4×4	5354 µJoule	5833 µJoule	26.1	27
5×5	7277 µJoule	7922 µJoule	42.8	42.7
6×6	10050 µJoule	11395 µJoule	55.9	56.9

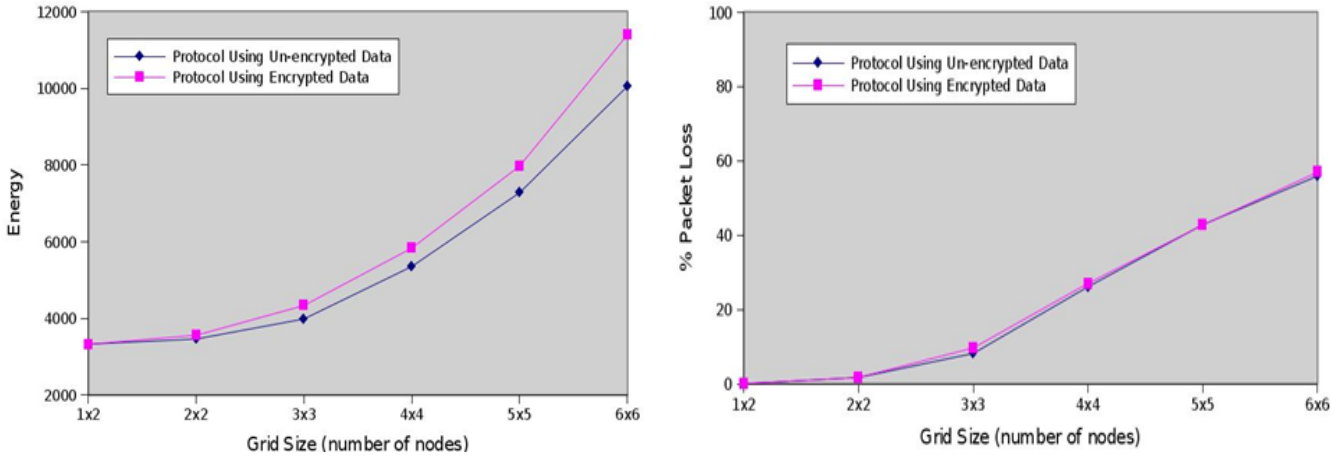

Fig. 15. Energy cost & % Packet Loss of Attestation's protocol for different grid size.

5.2. Improved Multihop Protocol

A Denial-of-service attack (DoS attack) is an attempt to make a resource unavailable to its intended users. Although the means to carry out, motives for, and targets of a DoS attack may vary, it generally consists of the concerted efforts of a person or persons to prevent a node from functioning efficiently or at all, temporarily or indefinitely [34-35].

A DoS flooding attack overwhelms a victim's limited resources, whether memory, processing cycles, or bandwidth. For example, if an adversary can cause a node to repeatedly broadcast messages, he can successfully drain power from that node. Worse, these messages will be rebroadcast by other intermediate nodes, draining their power. We need a broadcast protocol with a limited number of relay nodes to reduce the effect of DoS attacks [36-37].

In order to reduce the effect of DoS attacks, the base station informs all the authenticated nodes with the enemy's id. Using this id, the authenticated nodes block the enemy's packets. The enemy's id is transmitted to authenticated nodes, embedded on the Verification Messages, by the Base Station.

Using this improvement, the enemy's messages are received only by the enemy's neighbor nodes. This causes the successfully draining of enemy's neighbors nodes power. On the other hand, the rest network stays intact in enemy's attack.

5.2.1. Energy Analysis

The experiment scenario involves 16 Mica2 deployed in a 4x4 grid in such way that each mote can communication with its neighbours only. Furthermore, the node with the smallest id (ID0) is located in the upper edge of the network as shown in Fig. 16. The node with the ID0 is the Base Station. The enemy node is the node with the higher id. It is located in the edge of the network.

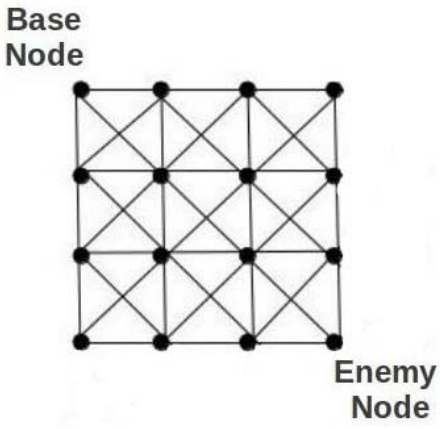

Fig. 16. The layout (4x4 grid) and links of the experimental setup. Each node can communicate with its (at most 8) neighbours.

During the experiment scenario, the nodes, except the base station, are in Regular Operation mode. These nodes transmit data to the base station every 5 seconds. The size of these data packets are 64bit. The enemy node transmits a 64bit packet every 0.5 second.

Table 5 compares the % packet loss and the total energy consumed by the network. The values represent the % packet loss and the energy consumed by the total network using:

- The default multihop protocol
- The improved multihop protocol

Fig. 17 represents the energy cost and the % packet loss of Attestation's protocol for different multihop protocol, in a graphic way.

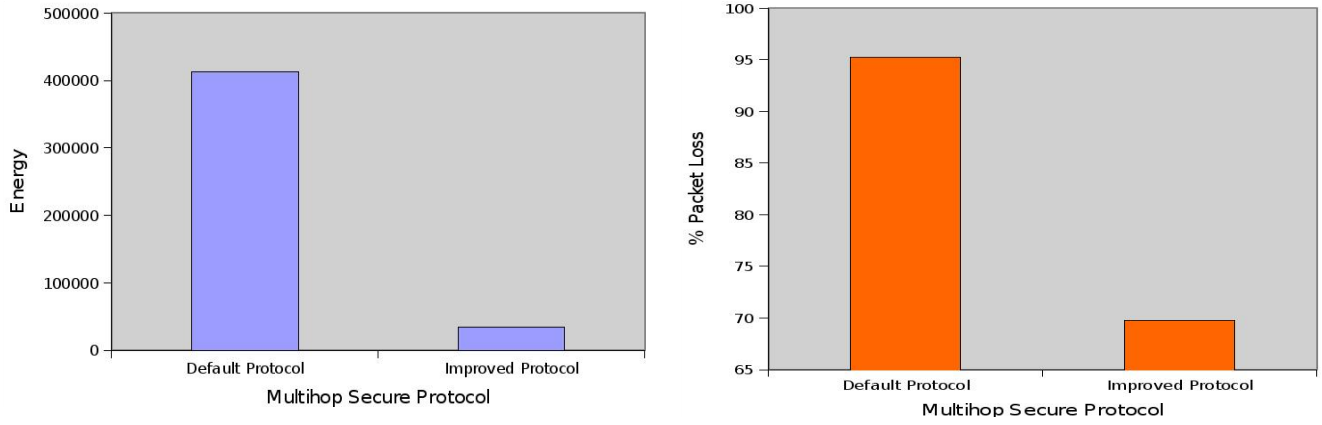

Fig. 17. Energy cost & % Packet Loss of Attestation's protocol for different multihop protocol.

Table 5. Attestation's protocol energy cost for default and improved multihop protocol.

Protocol Type	Energy Cost	% Packet Loss
Default	412908 μJoule	95.28
Improved	34348 μJoule	69.8

6. Security Analysis

Examples of our proposed framework efficiency against different types on attacks will be displayed in this paragraph along with real life scenarios that prove its robustness.

6.1. Physical Attacks

Physical attacks that can impact the coverage of the WSN and in many cases make the WSN inoperable. Because of the widespread placement of the individual nodes in an often non-secure and unmonitored area, individual nodes are subject to capture. Physical hardening of the sensors against reverse-engineering on chips, microprobing, glitch and power analysis, and cipher instruction search attacks on the first layer of security of the proposed framework can lead to the needed results.

6.2. Attacks on Random Key Generator

A possible attack on the generator is to broadcast on the frequencies that the RANDOM.ORG radios use in order to affect the generator. However, radio frequency attacks of this type would be difficult for a variety of reasons. First, the frequencies that the radios use are not published, so an attacker would have to broadcast across all frequencies of all bands used for FM and AM broadcasting. Second, this is not an attack that can be launched from anywhere in the world, only reasonably close to the generator. RANDOM.ORG currently has radio receivers in several different countries, which would make it difficult to coordinate this type of attack. Third, if an attacker actually did succeed at broadcasting highly regular signals (e.g., perfect sine waves) at exactly the right frequencies from the right locations, then the RANDOM.ORG real-time statistics would pick up the drop in quality very rapidly , which would raise an alert [25].

6.3. Replay Attacks

Replay attacks (i.e., intercepting a message and replacing it with an old message) cannot succeed as the proposed hash computation and verification are keyed operations that can be defeated as following: First, reporting a different SDI_ID will be detected by IVS when its uniqueness is checked and, moreover, the malicious sensor will not be able to pass the hash of RAM dump test unless it has the matching program which must be free of malicious codes and created an exact fingerprint. Second, modifying the Hash algorithm will cause inconsistency between two hash outputs and, hence, the verification will fail. Encryption of the communication channel makes it more difficult for an attacker to forge messages, as the messages have to be encrypted with the appropriate secret key.

6.4. Forgery Attacks

We will now show that it is impossible for the adversary to forge the hash value without the knowledge of all the specific parameters previously described, for each WDI. Consider the situation

where the adversary reprograms the sensor with a malicious program and attempts to fake the verification process by nullifying the effect of the output of the Hash algorithm. This is impossible because the Hash algorithm is inherently a nonlinear function of program blocks. Thus it is impossible to create a malicious WDI that has the same RHF as the original. Encryption of the communication channel makes it impossible for an attacker to forge the communication between two nodes as unique keys are distributed to every node.

So, what can prevent an attacker from capturing and reverse-engineering a sensor, and using the same sand-boxing technique to keep a good copy of the sensor running in order to feed good answers to the challenge-response protocol initiated by the verifier?

The attacker will not be able to manipulate the sending data of the captured sensor by internal means as changes will be detected by the present SDI. Any attempt to copy only the WDIs will fail as no authenticated messages are going to be sent to the authentication authority leading to rendering the sensor useless. Any attempt to copy both images SDI and WDI in a different sensor or a more resource efficient device will lead in creating different Ram dumps, both in size and structure leading to non verification of the WDI and locking the sensor out of the network. The only way you can copy both SDI and WDI in a different device and get valid results is by copying them in the exact sensor model (hardware and software) thus leading to forensically sound proof results.

6.5. Hardware Tampering Attacks

If the malicious sensor has enough memory to maintain the original program blocks some of the previously stated attacks can succeed. However, as it has been previously defined upon initiation of our Framework a specific fingerprint of both the hash value of the WDIs and RAM dump has been stored.

Therefore, there is no room left in the sensor for the adversary to save and execute arbitrary code. The adversary may attach more memory to each sensor, but it will incur a considerable amount of hardware modification while the Ram dump check will identify the attack. Moving/Copying the isolates in different sensors, as far as hardware specs is concerned, or a personal computer will lead to rendering the sensor useless because of the RAM dump check inconsistencies

6.6. Encryption Algorithms

TEA operates on 64-bit blocks and uses a 128-bit key. It has a Feistel structure with a suggested 64 rounds, typically implemented in pairs termed *cycles*. It has an extremely simple key schedule, mixing all of the key material in exactly the same way for each cycle. Different multiples of a magic constant are used to prevent simple attacks based on the symmetry of the rounds. The magic constant, 2654435769 or 9E3779B916 is chosen to be $232/\phi$, where ϕ is the golden ratio.

TEA has a few weaknesses. Most notably, it suffers from equivalent keys—each key is equivalent to three others, which means that the effective key size is only 126 bits. As a result, TEA is especially bad as a cryptographic hash function. TEA is also susceptible to a related-key attack which requires 223 chosen plaintexts under a related-key pair, with 232 time complexity.

Because of these weaknesses, the XTEA cipher was designed. **XTEA** (**eXtended TEA**) is a block cipher. The cipher's designers were David Wheeler and Roger Needham of the Cambridge Computer Laboratory, and the algorithm was presented in an unpublished technical report in 1997 [40].

Like TEA, XTEA is a 64-bit block Feistel network with a 128-bit key and a suggested 64 rounds. Several differences from TEA are apparent, including a somewhat more complex key-schedule and a rearrangement of the shifts, XORs, and additions.

Presented along with XTEA was a variable-width block cipher termed **Block TEA**, which uses the XTEA round function but applies it cyclically across an entire message for several iterations. Because it operates on the entire message, Block TEA has the property that it does not need a mode of operation. An attack on the full Block TEA was described in (Saarinen, 1998), which also details a weakness in Block TEA's successor, XXTEA.

As of 2004, the best attack reported on XTEA is a related-key differential attack on 27 out of 64 rounds of XTEA, requiring 220.5 chosen plaintexts and a time complexity of 2115.15 [41].

Because of this security issues a third version **Corrected Block TEA** (often referred to as **XXTEA**) was designed to correct weaknesses in the original Block TEA.

The cipher's designers were Roger Needham and David Wheeler of the Cambridge Computer Laboratory, and the algorithm was presented in an unpublished technical report in October 1998 [42].

Formally speaking, XXTEA is a consistent incomplete source-heavy heterogeneous UFN (unbalanced Feistel network) block cipher. XXTEA operates on variable-length blocks that are some arbitrary multiple of 32 bits in size (minimum 64 bits). The number of full cycles depends on the block size, but there are at least six (rising to 32 for small block sizes). The original Block TEA applies the XTEA round functions to each word in the block and combines it additively with its leftmost neighbour. Slow diffusion rate of the decryption process was immediately exploited to break the cipher. Corrected Block TEA uses a more involved round function which makes use of both immediate neighbours in processing each word in the block.

XXTEA is likely to be more efficient than XTEA for longer messages. Needham & Wheeler make the following comments on the use of Block TEA:

For ease of use and general security the large block version is to be preferred when applicable for the following reasons.

1 A single bit change will change about one half of the bits of the entire block, leaving no place where the changes start.
2 There is no choice of mode involved.
3 Even if the correct usage of always changing the data sent (possibly by a message number) is employed, only identical messages give the same result and the information leakage is minimal.
4 The message number should always be checked as this redundancy is the check against a random message being accepted.
5 Cut and join attacks do not appear to be possible.
6 If it is not acceptable to have very long messages, they can be broken into chunks say of 60 words and chained analogously to the methods used for DES.

However, due to the incomplete nature of the round function, two large ciphertexts of 53 or more 32-bit words identical in all but 12 words can be found by a simple brute-force collision search requiring 296−N memory, 2N time and 2N+296−N chosen plaintexts, in other words with a total time*memory complexity of 296, which is actually 2wordsize*fullcycles/2 for any such cipher. It is currently unknown if such partial collisions pose any threat to the security of the cipher. Eight full cycles would raise the bar for such collision search above complexity of parallel brute-force attacks.

The unusually small size of the XXTEA algorithm would make it a viable option in situations where there are extreme constraints e.g. legacy hardware systems (perhaps embedded) where the amount of available RAM is minimal.

An attack published in 2010 by E. Yarrkov presents a chosen-plaintext attack against full-round XXTEA, requiring 259 queries and negligible work. It is based on differential cryptanalysis. A chosen plaintext attack is an attack where the cryptanalyst is able to define his own plaintext, feed it into the cipher, and analyze the resulting ciphertext. Mounting a chosen plaintext attack requires the cryptanalyst to be able to send data of his choice into the device which is doing the encryption, and it requires the cryptanalyst to be able to view the output from the device. Because of these requirements, a chosen plaintext attack is in some cases impossible to attempt.

7. Conclusion

In this paper, we have proposed a complete tamper-proofing framework based on physical security schemes, encryption, digital forensics and sand-boxing techniques which offer 1) prevention of manipulation, reverse-engineering, and reprogramming of sensors; 2) purely software based protection with/without tamper-resistant hardware; and 3) infrequent triggering of the verification.

Through securely executed isolates a verification of the Integrity of the program of each sensor device is performed successfully. For verification, it remotely calculates, 1) hash value of every WDI being executed, 2) RAM dumps and checks if the values match with those stored on IVDB depending on the SDI_ID. All communication is through encrypted channels.

Our security analysis has proven that the proposed framework effectively defeats different types of attacks while improving the state of the art in software based protection mechanisms, furthermore from the simulations conducted the protocol has proven to be low

References

[1]. R. Anderson and M. Kuhn, Tamper Resistance—A Cautionary Note, *Proc. Second USENIX Workshop Electronic Commerce*, 1996, pp. 1-11.
[2]. D. W. Carman, P. S. Kruus, and B. J. Matt, Constraints and Approaches for Distributed Sensor Network Security, *NAI Labs Technical Report*, Vol. 00, No. 010, Sept. 2000.
[3]. R. Anderson, Why Cryptosystems Fail, *Comm. ACM*, Vol. 37, No. 11, Nov. 1994.
[4]. S. Blythe, B. Fraboni, S. Lall, H. Ahmed, and U. Riu, Layout Reconstruction of Complex Silicon Chips, *IEEE J. Solid-State Circuits*, Vol. 28, No. 2, Feb. 1993, pp. 138-145.
[5]. C. Collberg, C. Thomborson, and D. Low, Breaking Abstractions and Unstructuring Data Structures, *Proc. IEEE Int'l Conf. Computer Languages (ICCL '98)* , May 1998, pp. 28-38.
[6]. C. Wang, J. Hill, J. Knight, and J. Davidson, Software Tamper Resistance: Obstructing Static Analysis of Programs, technical report, *Dept. of Computer Science, Univ. of Virginia,* 2000.
[7]. C. Wang, J. Hill, J. Knight, and J. Davidson, Protection of Software-Based Survivability Mechanisms, *Proc. Int'l Conf. Dependable Systems and Networks*, July 2001, pp. 193-202.
[8]. G. Wroblewski, General Method of Program Code Obfuscation, in *Proc. of the Int'l Conf. Software Eng. Research and Practice (SERP),* June 2002.
[9]. M. Blum and S. Kannan, Designing Programs that Check Their Work, *J. ACM*, Vol. 42, No. 1, 1995, pp. 269-291.
[10].H. Wasserman and M. Blum, Software Reliability via Run-Time Result-Checking, *J. ACM*, Vol. 44, No. 6, 1997, pp. 826-849.
[11].F. Ergun, S. Kannan, S. R. Kumar, R. Rubinfeld, and M. Vishwanathan, Spot-Checkers, in *Proc.of the ACM Symp. Theory of Computing (STOC '98)*, May 1998, pp. 717-751.

[12].D. Aucsmith, Tamper Resistant Software: An Implementation, Information Hiding, *Springer-Verlag*, 1996, pp. 317-333.
[13].C. S. Collberg and C. Thomborson, Watermarking, Tamper-Proofing, and Obfuscation—Tools for Software Protection*, IEEE, Trans. Software Eng.,* Vol. 28, No. 8, Aug. 2002, pp. 735-746.
[14].B. Horne, L. Matheson, C. Sheehan, and R. E. Tarjan, Dynamic Self-Checking Techniques for Improved Tamper Resistance, in *Proc.of the 1st ACM Workshop Digital Rights Management (DRM)*, London, UK, 2002, pp. 141-159.
[15].H. Chang and M. J. Atallah, Protecting Software Code by Guards, in *Proc. of the 2nd ACM Workshop Digital Rights Management (DRM)* , 2002*,* pp. 160-175.
[16].Taejoon Park, Kang G. Shin, Soft Tamper-Proofing via Program Integrity Verification in Wireless Sensor Networks, *IEEE Transactions on Mobile Computing*, Vol. 4, No. 3, May/June 2005, pp. 297-309.
[17].Squawk Project, http://labs.oracle.com/projects/squawk/ (Accessed 4 April 2011).
[18].A. Seshadri, A. Perrig, L. van Doorn, and P. Khosla. SWATT: Software-based ATTestation for Embedded Devices, in *Proceedings of the IEEE Symposium on Security and Privacy*, May 2004, pp. 272-282.
[19].M. Shaneck, K. Mahadevan, V. Kher, and Y. Kim, Remote software-based attestation for wireless sensors. in *Proceedings of the 2nd European Workshop on Security and Privacy in Ad Hoc and Sensor Networks*, 2005, pp. 27-41.
[20].Y. Yang, X. Wang, S. Zhu, and G. Cao, Distributed softwarebased attestation for node compromise detection in sensor networks, in *Proceedings of the 26th IEEE International Symposium on Reliable Distributed Systems*, 2007*,* pp. 219-230.
[21].O. Landsiedel, K. Wehrle, and S. Gotz, Accurate prediction of power consumption in sensor networks, *in Proc. of the 2nd IEEE Workshop on Embedded Networked Sensors (EmNetS-II). IEEE Computer Society*, 2005, pp. 37–44.
[22].B. L. Titzer, D. K. Lee, and J. Palsberg, Avrora: scalable sensor network simulation with precise timing, *in Proc. of the 4th Int'l Conf. Information Processing Sensor Networks (IPSN '05),* 2005, p. 67.
[23].http://www.xbow.com (Accessed 10 April 2011).
[24].NIST, Digital hash standard, *Federal Information Processing Standards Publication,* 180-1, April 1995.
[25].L. Foley, S. Wilson, Analysis of an On-line Random Number Generator, *Trinity College Dublin*, http://www.random.org (Accessed 8 April 2011).
[26].A. Zaharis, A. I. Martini, L. Perlepes, G. Stamoulis, and P. Kikiras, Live forensics framework for wireless sensor nodes using sandboxing, in *Proceedings of the 6th ACM Workshop on QoS and Security for Wireless and Mobile Networks (Q2SWinet '10),* 2010, pp. 70-77.
[27].D. Wheeler and R. Needham., TEA, a Tiny Encryption Algorithm*, Springer-Verlag*, 1995.
[28].S. Hong, D. Hong, Y. Ko, D. Chang, W. Lee, and S. Lee., Differential cryptanalysis of TEA and XTEA., in *Proceedings of the ICISC 2003*, 2003b, pp. 402-417.
[29].E. Yarrkov, Cryptanalysis of xxtea. Cryptology ePrint Archive, Report 2010/254, 2010, http://eprint.iacr.org/ (Accessed 27 May 2011).
[30].Ovidiu Vermesan, Peter Friess, Patrick Guillemin, Sergio Gusmeroli, Harald Sundmaeker, Alessandro Bassi, Ignacio Soler Jubert, Margaretha Mazura, Mark Harrison, Markus Eisenhauer, Pat Doody, Internet of Things Strategic Research Roadmap, European Research Cluster on the Internet of Things, *Cluster Strategic Research Agenda,* 2011.
[31].Feistel H., Cryptography and Computer Privacy, *Scientific American,* Vol. 228, 1973, No. 5, pp. 15-23.
[32].García Villalba, Luis J., Sandoval Orozco, Ana L., Triviño Cabrera, Alicia, Barenco Abbas, Cláudia J., Routing Protocols in Wireless Sensor Networks., *Sensors,* 9, No. 11, 2009, pp. 8399-8421.
[33].Victor Shnayder, Mark Hempstead, Bor-rong Chen, Geoff Werner Allen, and Matt Welsh, Simulating the Power Consumption of Large-Scale Sensor Network Applications, in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems* (SenSys '04), New York, USA, 2004, pp. 188-200.
[34].Ray Hunt, Network Security: The Principles of Threats, Attacks and Intrusions, part1 and part 2, *APRICOT*, 2004.
[35].Ehab Al-Shaer, Network Security Attacks I: DDOS, *DePaul University*, 2007.
[36].A. D. Wood and J. A. Stankovic, Denial of service in sensor networks, *Computer*, Vol. 35, No. 10, 2002, pp. 54–62.
[37].Chien-Chun Ni, Tien-Ruey Hsiang J. D. Tygar, A Power-Preserving Broadcast Protocol for WSNs With DoS Resistance, in *Proceedings of the ICCCN' 200*8, pp. 777-782.
[38].Dunkels Adam, Vasseur JP, IP for Smart Objects, Internet Protocol for Smart Objects (IPSO) Alliance, Sepember 2008.

[39]. I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, Wireless sensor networks: a survey. *Comput. Netw*., 38, 4, March 2002, pp. 393-422.
[40]. D. Wheeler and R. Needham, *TEA Extensions*, October 1997.
[41]. Ko Y., Hong S., Lee W., Lee S., Kang J., Roy B., Meier W., Related Key Differential Attacks on 27 Rounds of XTEA and Full-Round GOST, *11th Fast Software Encryption Workshop, Lecture Notes in Computer Science,* No. 3017, 2004, pp. 299-316.
[42]. Wheeler, D. J., & Needham, R. J. Correction of xtea, Unpublished manuscript, Computer Laboratory, *Cambridge University*, 1998.
[43]. M. Maimour, H. Zeghilet, F. Lepage, Cluster-based Routing Protocols for Energy Efficiency in Wireless Sensor Networks, CRAN laboratory, *Nancy University*, CNRS, France.
[44]. P. K. Kikiras, J. N. Avaritsiotis, Unattended Ground Sensor Network for Force Protection, *Journal of Battlefield Technology,* Vol. 7, No. 3, November 2004.

Sensors & Transducers Journal

Guide for Contributors

Aims and Scope

Sensors & Transducers Journal (ISSN 1726-5479) provides an advanced forum for the science and technology of physical, chemical sensors and biosensors. It publishes state-of-the-art reviews, regular research and application specific papers, short notes, letters to Editor and sensors related books reviews as well as academic, practical and commercial information of interest to its readership. Because of it is a peer reviewed international journal, papers rapidly published in *Sensors & Transducers Journal* will receive a very high publicity. The journal is published monthly as twelve issues per year by International Frequency Sensor Association (IFSA). In additional, some special sponsored and conference issues published annually. *Sensors & Transducers Journal* is indexed and abstracted very quickly by Chemical Abstracts, IndexCopernicus Journals Master List, Open J-Gate, Google Scholar, etc. Since 2011 the journal is covered and indexed (including a Scopus, Embase, Engineering Village and Reaxys) in Elsevier products.

Topics Covered

Contributions are invited on all aspects of research, development and application of the science and technology of sensors, transducers and sensor instrumentations. Topics include, but are not restricted to:

- Physical, chemical and biosensors;
- Digital, frequency, period, duty-cycle, time interval, PWM, pulse number output sensors and transducers;
- Theory, principles, effects, design, standardization and modeling;
- Smart sensors and systems;
- Sensor instrumentation;
- Virtual instruments;
- Sensors interfaces, buses and networks;
- Signal processing;
- Frequency (period, duty-cycle)-to-digital converters, ADC;
- Technologies and materials;
- Nanosensors;
- Microsystems;
- Applications.

Submission of papers

Articles should be written in English. Authors are invited to submit by e-mail editor@sensorsportal.com 8-14 pages article (including abstract, illustrations (color or grayscale), photos and references) in both: MS Word (doc) and Acrobat (pdf) formats. Detailed preparation instructions, paper example and template of manuscript are available from the journal's webpage: http://www.sensorsportal.com/HTML/DIGEST/Submition.htm Authors must follow the instructions strictly when submitting their manuscripts.

Advertising Information

Advertising orders and enquires may be sent to sales@sensorsportal.com Please download also our media kit: http://www.sensorsportal.com/DOWNLOADS/Media_Kit_2012.pdf