
Luigi Pomante, Marco Pugliese, Stefano Marchesani, and Fortunato Santucci
Center of Excellence DEWS
University of L’Aquila
L’Aquila, Italy
{luigi.pomante, stefano.marchesani, fortunato.santucci}@univaq.it, marco.pugliese@ieee.org

Abstract—This paper deals with the design and implementation of security mechanisms in networked embedded systems (e.g. Wireless Sensor Networks (WSN)), without the support of external “resource unconstrained” facilities. While most literature contributions propose to put intelligence (that is usually more consuming both in computational resources and in memory as well) outside the WSN, we have already reported in some previous works that, under certain limitations, a WSN can operate as a functionally “autonomous entity” not only for sensing operations. This paper provides a global overview of the related research project and then reports on recent design upgrades to what we denote as the “Secure Platform”, i.e. the SW platform over WSN that is able to support a wide range of applications in non-standard environment such as monitoring of critical infrastructures, data acquisition in hazardous environments under standard security conditions. The work presented here is a partial achievement of the internal project WINSOME (WIreless sensor Network-based Secure system fOr structural integrity Monitoring and AlErting) at our Centre DEWS, whose target is to develop a cross-layer secure framework for advanced monitoring and alerting applications.

Keywords—security, wireless sensor network, secure platform, security components

I. INTRODUCTION

Homeland security and monitoring of critical infrastructures, such as buildings, bridges, nuclear power plants, aircrafts, etc., represent challenging application domains for modern networking technologies. In this context wireless sensor networks are gaining interest as a fundamental component of an advanced platform that embeds pervasive monitoring, networking and processing. Indeed, recent literature has addressed the perspectives of WSNs for monitoring structural and functional health of industrial plants, e.g. [2]: nevertheless, we can observe that the dominating paradigm is to exploit WSN features in terms of a “network of small sensors”, while still unexplored is the more advanced paradigm of “networked smart sensors” and the underlying opportunity to actually support autonomous (anomaly) detection processes. Adequate levels of security and trustiness are also essential to enable development and deployment of services in realistic environments. A large body of specialized literature deals with this topic and several ad-hoc solutions can be found. On our side, in [15], [16], [17] and [18] we have developed a different approach in this context: resorting to security mechanisms that are made available in traditional networks can provide a suitable and reliable framework, while smart adaptations are targeted to meet tighter resource constraints and possible performance degradation. Within our research framework, this paper reports the newest design upgrades and implementation achievements towards a middleware platform that is able to provide a WSN with secure services. We denote as “Secure Platform” that SW platform that is intended to support a wide range of applications in non-standard environment such as monitoring of critical infrastructures, data acquisition in hazardous environments, also in tactical operations under standard security conditions [15], [16], [17] and [18]. If proper design approaches [19] are adopted, the provision of fundamental security services can be pursued. In this context, providing security in a WSN system cannot be restricted to providing a robust cryptographic scheme, also because this kind of schemes are heavy demanding in terms of computational power and memory. Indeed, a smart intrusion detection service should be provided also with ciphering and authentication in order to build up a “security service” package that will enhance the typical middleware services provided by an Application Execution Environment (AEE).

This paper is organized as follows: Sec. II deals with the security services provided by the “Secure Platform”, Sec. III and Sec. IV describe fundamental algorithms and architectures supporting those security services, while Sec. V reports the design approach of the platform.

II. SECURE PLATFORM SERVICES

Fig. 1 shows the main functional blocks of the proposed Secure Platform: apart from the block providing the typical middleware services (MW Services) and shared memory, other specific services (in this case security-oriented) are implemented as customizations of specific SW component and provided to the AEE via different APIs. It is very important to
note that the “secure platform approach” offers a promising guideline to design and implement “integrated security” over WSNs through an “application-oriented” approach which is aligned to the current SW development paradigms over resource constrained devices [8], [12] and [19].

In this case at least two functional blocks are provided: the cryptography module, which can implement TAKS (Topology Authenticated Key Scheme) or, alternatively, ECTAKS (Elliptic Curve-based TAKS), that represents its extension to elliptic curve cryptography (ECC) [10], and the intrusion detection module, which implements WIDS (Weak process model-based Intrusion Detection System): the former module represents a novel contribution of the present paper that enhances both works [17] and [18] by exploiting the advanced security features of elliptic curves, while the second module integrates the developments proposed in [15] and [16].

TinySEC [7] and TinyECC [13] modules respectively represent the basic security packages in WSN as native integration with TinyOS [20] (i.e. the widely used operating system over WSN), and its extension to support ECC. Our schemes TAKS and ECTAKS, respectively, rely on TinySEC and TinyECC security services to authenticate / encrypt / decrypt messages. Sec. III and Sec. IV deal with TAKS / ECTAKS modules and WIDS module, respectively, as well as with security and cost evaluations; background material for related topics can be found in [15], [16], [17] and [18].

III. THE CRYPTOGRAPHIC SCHEME: TAKS AND ECTAKS

A. Motivations

Providing security in traditional networks often means using asymmetric cryptographic scheme, or Public Key Cryptography (PKC). The increasing of available computational, memory and power resources make it possible to ignore the main disadvantage of this strategy: the robustness of asymmetric algorithms is highly dependent on the size of the keys that obviously affects the complexity of the algorithms. Moreover, the topology changes that often occur in WSNs make PKC not practicable. As a consequence, in these cases symmetric cryptography must be revalued. In this context, the most important problem is related to key management, because encryption and decryption algorithms are lighter than those based on asymmetric key. This problem is one of the most addressed in the literature [1]: there are pair-wise key pre-distribution solutions that are based on deterministic pre-distribution of keys for each pair of nodes. The trivial solution consists in distributing a key for each pair of nodes, with eventually the same key for the entire network. Other random pair-wise key schemes are based on storing only a subset of all possible keys in each node [1]. To communicate with another one, a node needs to negotiate a key with its peer, randomly selecting one element in its subset. Whenever locations of nodes are known, it is possible to simplify the last algorithm providing to each node only the keys for the actual neighbors. Other techniques are based on cluster pre-distribution: different keys are used in each cluster, while within in a cluster keys can be build specifically for each node pair on the base of the nodes composing the cluster itself.

With respect to previous solutions, our proposed scheme does not rely on pre-distribution of keys in nodes, but it considers their dynamic generation based on partial information stored in nodes as in [18]. Through computationaly inexpensive operations, a node can compute the decrypt/encrypt key in a single phase with no steps of setup/negotiation. Furthermore, it is possible to authenticate a message (or better, the key with which it was encrypted) with respect to a planned network topology. From this assertion, the algorithms were named TAK (Topology Authenticated Key) and TAKS (Topology Authenticated Key generation Scheme). We define qualitatively planned network topology as the network topology planned by a service manager (the planner) to satisfy some service requirements. According to this definition, the planned network automatically gets the attribute of certified network topology where the certification authority is the planner itself. The path to ECTAKS is as follows: ECC [10] has been the top choice among various PKC options due to its smaller key size and compact signatures: for example, to provide equivalent security to 1024-bit RSA, an ECC scheme only needs 160 bits on various parameters, such as 160-bit finite field operations and 160-bit key size [9]. TinyECC [13], targeted at TinyOS [20], includes almost all known optimizations for ECC operations. Taking into account the above considerations, it was shown in [17] how TAKS can be enhanced using an ECC-based vector algebra (therefore the new denomination ECTAKS) and be compatible with TinyECC.

B. Basic TAKS Scheme

While a more detailed description can be found in [18], here the essential design principles are summarized: TAKS requires the offline setup of some topology-related parameters to allow their pre-distribution in the entire network:

- **Local Key Component (Loc.Key,Comp.);**
- **Transmitted Key Component (Trs.Key,Comp.);**
- **Local Planned Topology (Loc.Pld.Top.)** i.e. a set of vectors (denoted as “Topology Vectors”) in one-to-one relationship with admissible neighbor nodes.

Information is classified according to the following definitions:

- **public:** any information anyone can access (attackers included);
are currently testing in our lab includes some important upgrades with respect to [18]; therefore, we introduce TAKS2 whose new features are synthetically described below:

- Each Topology Vector in a node coincides with the Transmitted Key Component associated to each admissible neighbor. Therefore, a smaller amount of memory is needed in each node to store static information.
- The transmission protocol is 1-phase (i.e. there is no need of other party response message to proceed) as there is no need of a priori exchange of the Transmitted Key Components between nodes (as occurs in [18]): each transmission contains the ciphered text (c), the authentication tag and the ephemeral Transmitted Key Component of the transmitter. Any node which receives this message can check whether it is the right recipient and message integrity as well. The ephemeral Transmitted Key Component is defined as the Transmitted Key Component multiplied by a one-shot random value;
- The Secret Share (SS) for each admissible node pair is given by TAK multiplied by a one-shot random value (in [18] it is coincident with TAK): therefore, in TAKS also SS is a one-shot random value for each admissible node pair and security level gets enhanced (in [18] there is only a SS value for each admissible node pair, hence each node must store several SSs according to the number of admissible neighbors).

Suppose $q = 2^k$ where $k$ is an integer such that $q = 2^k >> N$ with $N$ the total number of nodes in the network. Let $U$ be a vector space over $GF(q)$ where the generic vector $u \in U$ is represented with a 3-ple $(u_1, u_2, u_3)$ of vector components elements in $GF(q)$. The introduced upgrades imply some slight refinements to requirements defined in [18]. Let $TAK(\cdot, \cdot)$ be a function satisfying the following requirements:

1) $TAK(\cdot, \cdot)$ is not invertible and $TAK(u, u') \neq 0$, $\forall u, u' \in U$;
2) $TAK(u, f(u')) = TAK(u', -f(u))$, $\forall u, u' \in U$ where $f(\cdot)$ is an arbitrary vector function over $GF(q)$;
3) $TAK(\alpha u, u') = TAK(u, \alpha u') = \alpha TAK(u, u')$, where $\alpha$ is a random value in $GF(q)$.

Let $g(\cdot, \cdot)$ be another function satisfying the following requirements:

4) $g(p, v)$ is not invertible;
5) $g(p, v) = 0$ only for a predefined set of distinct values of $p, v$.

According to Kerchoff principle, the explicit expressions for both $TAK(\cdot, \cdot)$ and $g(\cdot, \cdot)$ are public. Fig. 2 reports the conceptual representation of the proposed scheme. If $n_i$ wants to communicate with $n_j$, it has to generate a random value $\alpha$ in $GF(q)$ and build the message as concatenation of:

- The ciphered text ($c$) produced by an encryption algorithm with symmetric key equal to $\alpha TAK(Loc.Key.Comp_j, Trx.Key.Comp_j)$;
- The deciphering information ($d$) where $d \in U$ and $d = -\alpha Trx.Key.Comp_j$ as key;
- The message authentication code ($\tau$) associated to the cipher text ($c$) with key equal to $\alpha TAK(Loc.Key.Comp_j, Trx.Key.Comp_j)$.

When $n_j$ receives the message, it has to compute a pair-wise key to decrypt it. It computes its own key as $TAK(Loc.Key.Comp_j, d)$. If $Trx.Key.Comp_j$ and $Loc.Key.Comp_j$ are compliant to requirements 2 and 3, we have that:

\[
\]

The expression $TAK(Loc.Key.Comp_j, d)$ is validated as a true TAK only if the verification function $g(\cdot)$ authenticates the received message.
D. The Authentication Scheme

Let $MAC(.)$ be a cryptographic hash function. We define a verification function $g(p, v) = MAC(SS_j) − MAC(SS_j)$ where $SS_j = −αk_{i,j}·k_j, \tau = \tau$ and $SS_j = d · k_{i,j}$. Given a node pair $n_i$ and $n_j$, if $g(p, v) = 0$ then $n_i$ is network topology authenticated by $n_j$. The proof is straightforward: if $n_i$ receives a message from $n_j$, it will calculate $SS_j = k_{i,j} · d = −αk_{i,j}·k_j$. $n_i$ provides $\tau$ as a result of $MAC(SS_j)$ with $SS_j = αk_{i,j}·k_j$. If $g(SS_j, SS_j) = 0$ then $−αk_{i,j}·k_j = αk_{i,j}·k_j$, i.e., $n_i$ is network topology authenticated by $n_j$.

E. TAKS2 extension to ECC (ECTAKS)

The main drawback in TAKS2 is the ephemeral Transmitted Key Component to be transmitted each time (in [18] only at the setup phase). This increases energy consumption per transmission and it can turn to a problem in case of large data transmission rates and large key size. In monitoring applications, transmission rates are related to sampling rates on sensor boards, which depend on the dynamics of the monitored system: if large transmission rates are needed, key size (hence vector size) should be reduced without degrading security; in this occurrence ECC facilities should be included into TAKS2 (according to [9]). ECC schemes can provide the same security level of conventional schemes even with key size about 7 times smaller) at a cost of higher computation time (see next Sec. III F). TAKS2 extension to ECC is denoted as ECTAKS and, in case of high transmission rates, this option should be preferred to TAKS or TAKS2. Details about ECTAKS can be found in [17]. Here just some the main design principles are summarized. Extension to ECC leads to enhance the vector algebra over $GF(q)$ in TAKS to a vector algebra over EC in ECTAKS: hence expression (1) is replaced with the following one, where $P$ is a (public) EC point.

$$αECTAKS(\text{Loc.Key.Comp}_{i,j}, \text{Trx.Key.Comp}_{i,j} · P) = (2)$$

$$αECTAKS(\text{Loc.Key.Comp}_{i,j}, −\text{Trx.Key.Comp}_{i,j} · P) =$$

$$ECTAKS(\text{Loc.Key.Comp}_{i,j}, α\text{Trx.Key.Comp}_{i,j} · P) =$$

$$ECTAKS(\text{Loc.Key.Comp}_{i,j}, d)$$

where $d = −α\text{Trx.Key.Comp}_{i,j} · P$.

Moreover, some new operators which couple scalars, scalar vectors and point vector have been introduced in [17].

F. Security and Cost Analysis

We will show the security level provided by TAKS and ECTAKS. The relevant questions and related answers are as follows:

1. Which is the entropy per binit associated to TAK and ECTAK? For both schemes, the entropy per binit is $≈ 1$. The proof for TAK is given in [18]; the proof for ECTAK is straightforward, when we take into account that, as the point $P$ is public (and its contribution to entropy is zero), the uncertainty associated to ECTAK must be the same as TAK.

2. How much complex is the inverse problem to break TAK and ECTAK (security level in a single node)? The complexity of the inverse problem for TAK is at least comparable to DLP [18]. The complexity for ECTAK is at least comparable to EDLP, that seems to be intractable if EC is appropriate (it must be a non-supersingular curve, and $q$ such that the number of points in EC is not divisible by a large prime and that $q$ is itself a large prime [14]).

The cost is measured in terms of computational time for both TAKS and ECTAKS. Let us consider the case $q = 2^{128}$ (a 128 bit key size): it can be shown that TAK and ECTAK can be computed through $\sim$300 and $\sim$60000 16-bit operations (additions and products). If MicaZ motes are employed (8-bit processor MPR240 @ 7.4 MHz), and assuming 10 clock cycles / operation, the computation time for generation of 128-bit TAK is estimated to be about $\sim$400 $\mu$s [18] and a 128-bit ECTAK about $\sim$80 ms [17].

IV. WEAK PROCESS-BASED INTRUSION DETECTION SYSTEM (WIDS)

A. Motivations

The further security service component in our Secure Platform is the intrusion detection logic (IDS). Its main function is to identify abnormal network activity that differs from the expected behavior. In [15] and [17] we have shown how a light state-based anomaly-based detection logic can be suited to be implemented over WSNs [11]. Smart nodes are typically provided with mechanisms to identify changes in system parameters or anomalous exchanges of information: such data can be used as relevant observations to predict the hidden state of the system and infer whether it is under attack. A Hidden Markov Model (HMM) [4] is a doubly stochastic finite state machine with an underlying stochastic process that represents the real state of the system: the real state of the system is hidden but indirectly observable through another stochastic process that produces a sequence of observable events. The relationships between hidden states and observable data are stochastic as well as the transitions among states. While detailed description of models and proofs can be found in [15] and [17], we would like to emphasize here that our consolidated line of research is oriented towards a network-layer anomaly detection logic which exploits Weak Process Models (WPM) [15] over WSN, denoted here as WIDS (WPM-based Intrusion Detection System): WPMs are non-parametric version of HMM, wherein state transition probabilities are reduced to rules of reachability in a graph representing the abnormal behaviours. The estimation of a threat in the case of weak processes is greatly simplified and less demanding for resources. The most probable state sequence generated by the Viterbi algorithm [6] for HMM becomes the possible state sequence generated by simplified estimation algorithms for WPMs. The intensity of the attack is
evaluated by introducing a threat score, a likelihood criterion based on weighting states and transitions.

B. The Logic

In [15] we introduced two classes: LPA (Low Potential Attack) and HPA (High Potential Attack). An attack is defined in a “low potentially dangerous” state (or in a LPA state) if the threat is estimated to be in an early stage, otherwise it is defined in a “high potentially dangerous” state (or in a HPA state) if the threat is estimated to be close to its completion. WIDS module identifies any observable event correlated to a threat by applying a set of anomaly rules to the incoming traffic. Attacks are classified into LPA or HPA according to the WPM-based model for abnormal system behaviour and the WPM-based model for abnormal system behaviour and the Anomaly Detection Logic (ADL) block implements detection and alarm generation functions. The Intrusion Reaction Logic (IRL) schedules the intervention priority towards the compromised nodes according to specific criteria (defence strategy); IRLA applies the countermeasures against attacks to compromised nodes, including node isolations (quarantine), key revocations, link release or inclusions in black lists / grey lists.

The Logic

In [15] we introduced two classes: LPA (Low Potential Attack) and HPA (High Potential Attack). An attack is defined in a “low potentially dangerous” state (or in a LPA state) if the threat is estimated to be in an early stage, otherwise it is defined in a “high potentially dangerous” state (or in a HPA state) if the threat is estimated to be close to its completion. WIDS module identifies any observable event correlated to a threat by applying a set of anomaly rules to the incoming traffic. Attacks are classified into LPA or HPA according to specific states in the corresponding WPM-based threat model (Fig. 3). Alarms are issued as soon as one or more high potential attacks are detected. WIDS is tailored to detect attacks coming from insider intruders, i.e. attacks based on well-formed messages generated by authenticated nodes where control information is chosen to generate malicious network topologies.

V. SECURE PLATFORM DESIGN

The adopted architectural design [16] is cross-layer [12] and platform-based [19]. Cross-layer (CL) results in the interplay between network layer (topology management and routing protocol) and presentation layer (mobile agent based execution environment for distributed monitoring applications): when applied to security, an important benefit of CL mechanism is the exploitation of the interplay between different security mechanisms in different layers to provide an enhanced security service to applications. Platform-based design (PBD) results in the availability of a software platform where the internal structure is composed by “interconnected” SW components, which represent abstractions of the wired hardware components. Achievements of research goals are sought by taking care of the following major topics: selection of the right layers in the architectural design (a middleware layer is an essential component), application of the platform-oriented concepts for service mappings between layers, enhancement of the middleware layer with security services offered by lower layers entities and, on top, the creation of a flexible AEE by means of agents. Fig. 4 depicts WIDS functional blocks: the Threat Model (TM) block implements the WPM-based model for abnormal system behaviour and the Anomaly Detection Logic (ADL) block implements detection and alarm generation functions. The Intrusion Reaction Logic (IRL) schedules the intervention priority towards the compromised nodes according to specific criteria (defence strategy); IRLA applies the countermeasures against attacks to compromised nodes, including node isolations (quarantine), key revocations, link release or inclusions in black lists / grey lists.

A. Mobile Agent-based Middleware

A key characteristic of mobile agent-based middleware is that any host in the network is allowed a high degree of flexibility to possess any mixture of code, resources, and processors. Its processing capabilities can be combined with local resources. Code (in the form of mobile agents) is not tied to a single host, but it is rather available throughout the network. Moreover, the mobile agent paradigm supports data-centric applications because the implementation code can migrate towards data, no matter about node addressing. Therefore, in a mobile-agent application execution environment, each agent implements a sub-set of application components which can be proactively aggregated through agent mobility (code mobility across the network). Among the agent-based middleware solutions available from literature, we will refer to AGILLA [5]. According to the function decomposition shown in Fig. 4, the mapping between WIDS functions and SW / agent components is as follows: ADL and TM blocks are mapped into SW components, while IRL and IRLA blocks are mapped into a mobile agent, which is denoted by Intrusion Reaction Agent (IRA). This design enables optimal allocation and code distribution for those functions that do not need to be implemented anywhere.

B. Enhancements to AGILLA middleware

Although AGILLA middleware fits important requirements of our Secure Platform, considerable adjustments were needed in our case and are still under development. Enhancements have been designed as NesC stubs embedded into AGILLA code, as reported in [16]. Fig 5 schematically represents this added interfaces as bold arrows: the design philosophy is that each added SW component should be able to directly interface the AGILLA native building blocks (the Service Manager, the Neighbor List, the shared Tuple Space) as well as to directly connect to AAEE. In other words, each new functional
component is fully integrated into the new AGILLA platform as a peer building block.

![AGILLA Application Execution Environment (AAEE)](image)

**Fig. 5 Enhanced AGILLA Mobile Agent-based Secure Platform Architecture**

## VI. CONCLUSIONS AND PERSPECTIVES

In this paper we have presented our contributions to the design and development of a “Secure Platform” as a cross-layered framework providing security over WSNs. Major efforts in current activities are focused on completing a prototype implementation through mobile agents supported by a MicaZ wireless sensor network. This work is a partial achievement of the internal project WINSOME (Wireless Sensor Network-based Secure System for structural integrity Monitoring and AllErging) at Centre of Excellence DEWS, whose target is to develop a cross-layer secure framework for advanced monitoring and alerting applications. Several developments are also planned for the near future. One objective is to extend WIDS to detect anomalies in data message content and signalling as well: in this frame bayesian analysis and decision techniques (e.g. the Dempster-Shafer theory) have been successfully applied in traditional networks where resource availability is not a problem, but in WSNs it might be a big issue. Another important issue is to consider monitoring as a component in a control process where correlated actuations on the environment can be performed. This vision implies the integration of Hybrid System Control [3] items into the service platform. Finally, from a signal processing and communication viewpoint, some efforts have been already devoted to optimize the information flow on WSNs: the existing correlation among measurement information taken from “contiguous” sensing units should be exploited to increase coding efficiency without losses (the Slepian-Wolf coding theory).

## ACKNOWLEDGMENT

We would like to thank Dr. Annarita Giani (UC Berkeley) for the joint work and the long discussions, so stimulating and profitable, on our common research items. Furthermore, the research leading to these results has received funding from the European Union Seventh Framework Programme [FP7/2007-2013] under grant agreements n° 257462 HYCON2 Network of excellence and n° 240555 ERC SG VISION. Moreover, it has been motivated and supported by the ESF-COST Action IntelliCIS (Prof. Fortunato Santucci is participating to this Action). The development of the middleware platform also fits in the frame of the Projects “Ricostruire” and “SMILING” supported by the Ministry of Economic Development to enhance technology transfer in the RIDITT framework.

## REFERENCES


