*Special issue: The future of software engineering for security and privacy*

Survey Paper

# Security software engineering in wireless sensor networks

Eric PLATON[1] and Yuichi SEI[2]
[1] *National Institute of Informatics*
[2] *The University of Tokyo*

**ABSTRACT**
**The engineering of security is an essential discipline in software engineering. It requires one to embrace a holistic approach, as any weakness along the engineering process of the system may lead to future security breaches. It is in general difficult to achieve and becomes particularly acute in wireless sensor networks, owing to the stringent limitations in communication and computational power. We survey the current state of the art on this topic and set forth issues that require further study. The analysis of current work covers general security issues of wireless sensor network research and discusses the present achievements for engineering security with regards to earlier surveys in this domain. We also cover security capabilities of major implementation platforms, namely TinyOS and Sun SPOT$^{TM}$, and present available and required mechanisms that will become essential for software engineers.**

**KEYWORDS**
Security engineering, wireless sensor networks, software engineering, survey

## 1 Introduction

Security is beyond doubt one of the great challenges in software systems. Most companies rely extensively on information systems to support their daily business activities. The multiplication and diversification of attacks against these vital systems call for appropriate protections. Beyond business, the fact that our lives are becoming increasingly digitized with the rise of internet-based services further emphasizes the need for security (e.g., protecting individuals' privacy). The advent of ubiquitous computing and other pervasive plans to integrate computers in any object stress security issues even more; virtually all aspects of life can be monitored, inspected, mined, and dissected in an Orwellian manner. Advances in miniaturization of hardware and energy technology permit unthinkable scenarios. Therefore, wireless sensor networks grow into both a great technology with notable applications and a potential threat, if security concerns are not considered appropriately.

Security engineering is an essential discipline of software engineering and the increasing awareness of security issues is widely recognized [1], [2]. Embracing a holistic approach on security is both very difficult and of crucial importance, as any weakness in the engineering process and in any component of the system may lead to a security breach. Security concerns become particularly acute in applications deployed over wireless sensor networks, owing to the stringent limitations in communication and computational power [3], [4]. Perrig et al. [4] expressed in 2004 that 'we have the opportunity to architect security solutions into [wireless sensor] systems from the outset, since they are still in their early design and research stages'.

The present paper reports on the situation in 2007, covering three aspects as follows. As a first broad aspect, we briefly review the main security concerns in wireless sensor networks (WSN) in the next section. For the second aspect, sections 3 and 4 report on research advances in security engineering. Section 3 presents research on security mechanisms and section 4 emphasizes research issues that are addressed insufficiently to guarantee high levels of security engineering, compared with the security objectives in WSN. Sec-

Received September 28, 2007; Revised November 16, 2007; Accepted November 29, 2007.
1) *platon@nii.ac.jp, http://honiden-lab.ex.nii.ac.jp/˜eric/*
2) *sei@nii.ac.jp, http://honiden-lab.ex.nii.ac.jp/˜sei/*
DOI: 10.2201/NiiPi.2008.5.6

tion 5 covers the third aspect, namely concrete technology and tools to engineer security features of systems. It briefly introduces the representative technologies that serve engineering applications on WSN and focuses on the latest available platform, Sun SPOT$^{TM}$ from Sun Microsystems, Inc.$^{TM}$. We present the security facilities provided by the platform and identify the current shortcomings of this technology.

## 2 Security Challenges in Wireless Sensor Networks

Past surveys on security in wireless sensor networks report research issues and solutions that remain challenging problems despite recent achievements [3]–[6]. In this section, we review the main concerns of security in wireless sensor networks to provide updated information and introduce the background of security engineering in WSN. There are three main elements for information security that need to be ensured:[1)] confidentiality, integrity, and availability [2]. Traditional techniques for these three concerns cannot be used off the shelf, because wireless sensor networks have stringent and distinctive characteristics. First, sensor nodes are usually deployed in environments that attackers can easily access. Sensor nodes are not usually tamper-resistant, because of hardware limitation and cost issues [7], [8]. Hence, attackers can get sensor nodes, extract stored keys, and then can insert malicious code. This node compromising is one of the main issues in wireless sensor networks. Second, sensor nodes are limited in their energy and computation abilities. We briefly review confidentiality, integrity, and availability for wireless sensor networks considering these two issues.

### 2.1 Confidentiality

The protection of data assets and privacy are difficult requirements to fulfill on WSN but necessary to guarantee reasonable degrees of confidentiality. Network nodes are easy to capture and compromise, thus, fractions of assets can steadily be stolen from the system without appropriate protection against attacking nodes. A trusted network can also fail to ensure privacy if authentication and access control are not enforced consistently: queries to the network from different locations or with different credentials could deliver different access levels without adequate management. WSN are therefore vulnerable to Sybil attacks [9], distance-bounding attacks (masquerading nodes in some locations), and location leaks.

Confidentiality requires dedicated mechanisms. Traditional security solutions include encryption and protocols, which guarantee data hiding and consistent processing respectively. Encryption often relies on key infrastructures to secure the communications. Keys must be deployed on network nodes to guarantee the confidentiality of data. The physical attack of nodes can reveal some keys, so that confidentiality cannot be guaranteed locally to the attacked node [10], [11], with potential risk to epidemically break protections through the network. Encryption is also an expensive process, e.g. asymmetric cryptography, and a number of research teams work on controlling and reducing its cost impact [12], [13].

Protocols enforce systematic processing in the network, so that one can assume for example that access control is consistently applied throughout the network. WSN applications communicate using unreliable networks, where unpredictable environmental conditions can prevent protocols from being completed successfully. Protocols are therefore carefully crafted and provide degrees of resilience to network issues, but all additional functionalities incur computational costs that require time for completion. In other words, protocols cannot usually guarantee full consistency, notably in environments with fast-paced dynamics as in WSN. State-of-the-art technologies such as Network Access Control (NAC) — currently in active development — are tailored for networks with significant resources and the case of WSN is not yet considered as viable investment, thus leaving room for significant research.

Multi-level security is another mechanism related to confidentiality but challenging on WSN. It allows different users with different rights to use network data. For example, super-users can have full access with maximum resolution on the data whereas simple users can only access limited data with a low sample rate. The challenge again stems from physical issues: computation power and eavesdropping. Eavesdropping may break a multi-level security scheme by letting simple users access more data ‘by accident’. In addition, the scheme must be consistent throughout the network: for all network positions, a user must have the same level of access rights.

Most research efforts address these challenges with particular focus on cryptography and key establishment strategies.

**Cryptography**. Traditional cryptography mechanisms have been difficult to apply to WSN, owing to their limited computation and memory [14], [15]. Recent improvements in cryptographic primitives and power of network nodes allow their usage in practice (see section 5), but they still require significant resources.

**Key establishment**. Key establishment refers to the

[1)] Please refer to the discussion in 2.4 for their relation to other important topics, such as authentication and freshness.

deployment of a key infrastructure (KI) on a WSN. Much work focuses on the challenge to settle keys on ‘free’ nodes at the system initialization phase [16], but also at run-time when nodes must be added or replaced [17]–[19]. The difficulties in these challenges are to cope with compromised nodes, as they thwart KI playing its intended protective role.

### 2.2 Integrity

Integrity is a well-known issue in wired and mobile phone networks. WSN can therefore rely on a significant amount of models and techniques. WSN introduce additional challenges because of their limited resources. Despite the increase in available resources with latest hardware, physical constraints clearly prevent from comparing capacities on WSN with full-fledged notebooks or even low-end mobile phones, as WSN nodes tend to be even smaller [20]. Cryptography is among the main techniques to ensure integrity, and it remains a challenge with regards to energy management and decentralized approaches.

Locating data sources poses acute integrity issues in WSN. Forensics and network administrators will need to track data back to their sources in some circumstances (when possible, owing to destructive in-network aggregation). The location of sources requires that data is not altered when it migrates to the sink, so that integrity is a necessary condition.

The network infrastructures also differ significantly from other systems, so that existing work may not adapt to WSN models appropriately. In particular, the number of sensor nodes per base station is very large, compared to mobile phone or wireless broadband connections. These conditions stress the limitations of existing techniques, which are furthermore exasperated with the higher risk of node compromise. Routers are usually trustworthy network components, but sensor nodes play the role of routers in WSN, so that we cannot reasonably trust them because of compromise [21], [22]. Integrity relies then on the ability to detect this kind of attacks and their origins, which both remain challenging issues.

### 2.3 Availability

Availability also becomes a critical matter in WSN. The reasons are twofold and significantly constraining. First, limited energy supplies put an ultimate limit on network availability. Large systems can rely on human intervention and renewable energy solutions are under development, but it is notable that the software contribution to availability is bounded and that security engineering cannot rely exclusively on software solutions. Holistic measures involving hardware and human interventions are required in some conditions. Node compromise remains an issue, as the destruction or loss of control of nodes unilaterally reduces availability metrics.

Second, the broadcasting communication model of WSN is sensitive to denial-of-service attacks, including high-energy and jamming attacks [23]–[25]. Traditional attacks like message dropping are managed with standard approaches, such as multiple routing of the same message, but the resource limitations clearly restrict their viability.

### 2.4 Discussion

Challenges in WSN remain numerous, especially relative to the background of existing awareness for security in distributed and mobile systems. The importance of the holistic approach is salient due to the direct exposure of the system to external, environmental, and malicious threats. One difficulty in this issue is to find a balance between the software contribution, which consumes energy but is potentially robust against intrusions, and hardware or human contributions, which can better control energy, even renew it, but emphasize exposure. Security engineering has to cope with this balance.

In addition to main security properties, the literature refers to issues like quality of service, freshness, or authentication. Quality of service extends availability and is challenging, too. Data freshness refers to problems of inappropriate message duplicates (intentional or accidental) and is source of problems of integrity and availability. Authentication aims at guaranteeing the identity of principals and is therefore a precondition to confidentiality. The two issues of freshness and authentication are addressed in this survey within the corresponding main properties they relate to.

## 3 Existing security solutions

In this section, we review some representative work to address the aforementioned security challenges. We complement this review with the presentation of integrated projects that address several security issues into a single approach. The section ends with a mapping of this review on the standard waterfall-like software development process to show the present state of research on security engineering.

### 3.1 Confidentiality, integrity, availability

**Cryptography.** Cryptography constitutes the main theoretical concept, along with key infrastructures, underlying approaches that ensure integrity and confidentiality. Elliptic curve cryptography (ECC [26]) has been recognized as a viable approach for WSN, despite the absence of a formal proof for this technology as of 2007. Nevertheless, ECC is gaining momentum with a major company issuing an official technology adoption statement in 2005, and the American National Security Agency adding the approach to its official Suite B for digital signatures [27], [28]. ECC is a promising alternative to RSA-based algorithms [29], as the typical size of ECC keys is much shorter for the same level of security. Gaubatz et al. report on public-key cryptography in sensor networks in more detail [30], and Uhsadel et al. propose an efficient implementation of ECC [31].

The SPINS project shows that the security cost entailed by cryptography (among other computations) bears more on communication overhead per packet than on computational demand [7]. Despite this result, significant work aims at improving the execution of ECC algorithms with hardware support [32], or elaborate new algorithms for efficient multiplication [33], [34], as it is a very common and costly operation in public-key cryptography. Other approaches also exploit specific hardware for improving confidentiality, for example with tamper-resistant chips that prevent attacks from revealing cryptographic keys [35], to the cost of increasing the price of hardware units.

**Key infrastructures.** Key infrastructures draw particular attention from the research community for the concerns of confidentiality and integrity [36]–[41]. The management of keys is critical, at the root of the security defenses of the system. Keys in WSN have a life-cycle that emphasizes the deployment, protection, and revocation phases. The setup of key infrastructures often relies on pre-distribution schemes [40], [42]. Ghosh demonstrates that such approaches are valid if the network topology verifies structural properties like the absence of bottleneck nodes under some conditions [43]. Protection of keys is achieved either by specific hardware, as introduced above, or by security protocols that manage the life-cycle of keys. For example, regular revocation of keys is recommended to refresh the infrastructure and protect against stolen keys [44].

**Integrity and in-network processing.** Integrity concerns are also the subject of additional research relative to secure in-network aggregation technique. In-network aggregation aims at processing sensor data assets while they migrate toward the sink node, typically for computing averages or sums. It is an important technique to use node energy in more efficient ways in WSN. Several existing aggregation approaches lack security guarantees, such that subsequent research endeavors propose secure schemes [45]–[47].

**Location problems.** Locating data sources is a dilemma between verifying that a data source is a trusted party and protecting privacy. Capkun et al. introduce a secure positioning scheme using standard Manchester encoding to address the first aspect of locating nodes [48]. On the other hand, privacy has become a primary issue. Some approaches allow nodes to deliver data, while protecting location information with pseudonyms, since tracking a data source without its real identity is more difficult [49]. Other approaches achieve similar goals with physical measurements [50].

Fake location information causes security breaches, since an attacker can masquerade a node where there is none and become a data black-hole. Several research endeavors focus on the distance bounding attacks to cope with this issue, as referred in [51].

**Secured protocols.** Secured protocols protect WSN against attacks and misbehaviors that thwart the availability of nodes and services. In the representative work, Wood et al. conducted a survey on denial-of-service attacks, and they proposed an approach against radio jamming attacks [52], [53]. Karlof and Wagner expose a number of (successful) attacks against existing routing protocols for sensor networks and suggests countermeasures [54]. The rapid resource exhaustion of network nodes motivates the research against such kind of attacks, and several endeavors integrate appropriate defenses [55], [56].

Misbehaviors result from compromised nodes and insiders. The work of Lee and Choi deals for example with malicious nodes dropping packets [57], [58]. On the other hand, Ye et al. propose statistical means to recognize the insertion of false data [21], [59], while Sei and Honiden rely on a key management scheme to detect false information when a number of nodes are compromised [22]. The area of intrusion detection is also well represented in WSN, since the capture of nodes is potentially easier. Krauß et al. show how to detect compromised cluster heads (distinguished nodes equipped with additional functionalities) using attestation protocols [60]. Buttyán et al. locate wormholes using two statistical values, namely the change of the number of neighbors of a node and the decrease of the length of all shortest paths [61].

**Recovery.** Even though sensor nodes usually have a short life, it might be a turbulent one, not only in terms of the data they sense but also in terms of the software installed. There might be an inherent need to patch the code of a node for a security update — ideally without removing the sensor node from its location but via wireless communication. Such a code update has to meet certain requirements in terms of efficiency and security. SCUBA is a challenge-response protocol, which allows to repair compromised or damaged code [62], and Strasser and Vogt present algorithms for distributed node recovery [63]. It should be noted that automatic update approaches in WSN, such as network reprogramming [64], are usually not secured schemes. Deluge is an example where a secured version has been implemented afterwards [65], [66].

### 3.2 Integrated security solutions in WSN

The rich body of research achievements presented so far contribute to the security concerns in a rather 'local' way, focusing on a very specific issue. In the following, we present five approaches, namely LEAP, SPINS, TinySec and MiniSec, and TinyPEDS, to show the extent of some representative approaches on security in WSN.

#### 3.2.1 LEAP

The Localized Encryption and Authentication Protocol (LEAP) is a key management protocol designed to also support in-network processing [8]. Multiple symmetric keying mechanisms are supported, namely individual keys (shared with the base station), group keys (shared completely), cluster keys (shared with a selected group of nearby nodes), and pairwise shared keys. It aims at "restricting the security impact of a node compromise to the immediate network neighborhood of the compromised node" [8]. Using this granular approach, storage requirement per node is competitive.

#### 3.2.2 SPINS

SPINS stands for 'Security Protocols for Sensor Networks' [7]. It is an economical security scheme with low overhead. SPINS consists of two components called $\mu$TESLA and SNEP.

$\mu$TESLA is a specialization of the 'Timed, Efficient, Streaming, Loss-tolerant Authentication' protocol for WSN [67]. Its purpose is to provide authenticated broadcast, since this communication mode is the standard in WSN. The problem solved by $\mu$TESLA is that authenticated broadcast requires a costly asymmetric mechanism that sensor nodes cannot afford usually. The protocol emulates asymmetry by sending encrypted messages and key information independently. The sender first computes a key-chain with a random key $K_n$ and public one-way function. It then encrypts message packets using the key-chain in reverse order. A single key is used for all packets that can be sent in a predefined time frame. The sender sends encrypted data packets according to the time frame and delays the sending of keys. The receiver gets (securely) only the last key $K_0$ of the key-chain. It gets encrypted packets and, once the key arrives, it can authenticates the packet if the one-way function applied to the key results in $K_0$ (the function is applied as many times as the time interval frame number). Luk et al. show subsequently how to authenticate broadcast and addresses this problem in a more general way [68]. The second component SNEP (Secure Network Encryption Protocol) ensures data confidentiality, data authentication between two-parties, data freshness, and protection against replay in the protocol, with a low overhead of 8 bytes per packet (order in kilobyte for other approaches).

SPINS exemplifies how different security measures can be integrated in the case of WSN, and this typically shows that usual approaches cannot be trivially assembled for security engineering.

#### 3.2.3 TinySec & MiniSec

Karlof et al. introduced TinySec, a link layer security architecture, which introduces an overhead cost of less than 10% [12]. TinySec is thus claimed appropriate for use in WSN, and it has been adopted as a standard library in TinyOS. TinySec provides fully-implemented protocols and makes reasonable tradeoffs between performance and security. TinySec exploits the idea that, even though sensor nodes are very restricted in computational and communication power, even a powerful adversary suffers from low bandwidth in WSN. Message authenticity and integrity are guaranteed by using a MAC[2] with each packet (included in the overhead), and confidentiality is ensured by encryption with symmetric cryptography primitives.

MiniSec is a recent improvement and extension of TinySec's functionalities for node-to-node security in WSN [13]. It is more efficient than TinyOS in terms of energy consumption (5.6% more efficient in the published evaluation), and it provides an additional protection against replay attacks (malicious message duplication). Replay attacks are already managed by protocols such as ZigBee, but MiniSec is an alternative with significant performance improvements that maintain the level of security without sacrificing energy (working in the data link layer).

The mechanisms exploited in TinySec and MiniSec exemplify deployable security measures that address several issues in viable ways for WSN. It is notable that these two approaches explicitly identify data link layer security as fundamental in WSN, because of the observation that in-network computation such as aggregation is crucial in order to prolong the network's lifetime. If a standard end-to-end security scheme is applied, these computations become very intricate due to the possible aggregation schemes.

#### 3.2.4 TinyPEDS

TinyPEDS (Tiny Persistent Encrypted Data Storage) is a secured data store for WSN [69]. It aims at protecting data assets collected by the sensors, both for transient and long-term storages. TinyDB demonstrated the use of in-network aggregation for processing SQL queries, without security measures [70]. TinyPEDS elaborates on this approach by encrypting

[2] Message Authentication Code

transient storage for aggregation processing with symmetric keys, which request reasonable resources. Long-lasting storage is protected with an asymmetric key scheme however, owing to the need for stronger protection. Sensor nodes only receive public keys and the compromise of node does not reveal signing keys. A high-end node, typically a sink computer connected to the WSN, owns the signing keys and performs costly cryptographic computations. TinyPEDS combines secure storage with a replication protocol to restore data even with the loss of 40% of the nodes. TinyPEDS shows an example application that integrates advanced security measures, although several aspects remain to be covered to ensure high protection, however out of the scope of the project (e.g. attacks that eliminate nodes by exhaustion).

Some issues are not covered in the integrated approaches of this section, such as a number of attacks (e.g. resource exhaustion), node capture, and end-to-end security. It is notable however that current achievements put together integrate complementary security measures.

### 3.3 Security engineering in the software development process

Several approaches are available to secure various aspects of sensor networks, often for a reasonable cost. Research contributes significantly to isolated issues in the *engineering of security*, and part of this research has focused on integrating some aspects together. A secure system requires however consideration of security issues as a whole, i.e. in all phases of the software engineering development process. That is the reason for security engineering. Software designers and engineers are seemingly not supported sufficiently yet in the light of the review: If an efficient solution is to be created, **requirements** have to be specified very carefully. The framework of Slijepcevic et al. [71], for example, might assist the designers to adjust different security levels depending on the sensitivity of data. The unmanageable number of different threats is nevertheless hard to deal with. Much research covers **design** and **implementation** phases and describes how to protect specific aspects though. There is finally a clear shortcoming of study in the **testing** and **verification** phases, where engineers lack systematic tools or approaches tailored for sensor networks to check whether the specified security is achieved.

The aim of the following sections is to elaborate on this result: Security engineering requires further research to counter security methodological and instrumental holes in the software development process.

## 4 Issues in engineering security solutions

Existing work addresses several aspects of security concerns along the software engineering process, but the previous review shows that engineers cannot rely on present achievements to conduct a holistic approach on security. The specific concerns with WSN, namely node exposure and limited resources, remain indeed critical throughout the different stages of building the software layer of WSN applications.

The aim of this section is to present issues in security engineering in a progressive manner according to the standard waterfall software development life-cycle. The implementation stage is postponed to section 5 to focus on more technical matters.

### 4.1 Requirements engineering

The engineering of security requirements relies on specific aspects of the target software. One usually expresses constraints such as the definition of access rights, which usually leads to access control policies in the implemented system, or the segmentation of the system into secure and non-secure sub-parts. Although technological choices occur relatively late in the requirement engineering stage, the choice for WSN technologies leads to consider the security levels with care, notably relative to the system resilience to node capture and software-based attacks.

Requirement elicitation techniques are often needed to better specify target security qualities, and general ones provide a number of tools like extended use cases (which are also used in analysis and design phases). The literature refers to abuse cases to specify system usage patterns that design should prevent [72], [73], to misuse cases to relate correct and incorrect usage patterns [74], [75], and to security cases to organize security requirements [76], [77]. WSN-based applications require to focus on cases that should become ‘standard’ issues in a base set of requirements for any secure design. A typical standard case is when an attacker obtains encryption or decryption keys, because the consequences of such a scenario are the rupture of the security model. We have seen that tamper-resistant nodes are not feasible and in many applications one cannot prevent attackers from compromising nodes, so that the designers must consider the loss of keys as long as the client does not explicitly state this being superfluous. Well-known cases in mobile phone networks and other wireless technologies are likely to inspire cases in WSN, but the infrastructural differences justify further work.

Risk analysis methods provide additional tools in engineering security requirements. The advantage of risk analysis is to establish metrics over risk criteria and vulnerability, thus allowing quantitative and systematic analysis. FTA (Fault Tree Analysis) is a risk analysis method that identifies and classifies events in a tree according to their levels of harm to the system [78], [79]. The root of the tree then reveals potentially dangerous events, which the system must be prepared to deal with. Methods like FTA support the analysis of security requirements to the cost of building and exploiting a tree data structure. Although this approach applies well to a range of systems, the large number of nodes in WSN (thousands in some scenarios) tends to prevent such approaches from scaling up, so they become less usable in practice. The introduction of node compromise risks is also another issue that calls for further work to cope with WSN.

### 4.2 Analysis & design phase

The analysis of security requirements and their integration in the design is a sensitive task, which often relies on specialized programming staff in large projects. As such, specific tools that focus on security issues support engineers in these phases of the development.

Security patterns extend the design pattern approach with a set of patterns and association of patterns to guide the developers in their engineering endeavors [80]–[82]. For example, 'available system patterns' (ASP) and 'protected system patterns' (PSP) capture engineering practices that have been validated in several implemented systems. ASP aim at facilitating the construction of systems that provide predictable and uninterrupted access to services and resources they offer to users. PSP aim at protecting valuable resources against unauthorized use, disclosure or modification [83]. General security patterns apply to WSN, but several issues remain to be catalogued and validated by experience. For example, patterns to guarantee that the capture of a node and the loss of keys do not break the security level—or the breach can be confined—in the rest of the system.

In this respect, experience in developing WSN applications lead to distinguishing several situations relative to key loss, to be addressed by appropriate patterns, in addition to specific protocols [10], [22], [84]:
(1) the attacker has no key
(2) the attacker has less than $T$ keys
(3) the attacker has more than $T$ keys
$T$ is a threshold value that specifies the degree of robustness of the system against key loss, i.e., the number of key losses the system can tolerate without compromising the level of security. Taking inspiration from good practice for database security, patterns would ensure that services and assets provided in the system by each node are protected whenever $T$ keys are lost. The protection could be to, e.g., block any access to service and data but by special codes, similar to GSM and some smartcards security schemes with PUK and PUK2 (Personal Unblocking Key). This example illustrates a naive approach using security patterns, but we expect this research direction to provide significant results.

### 4.3 Testing & verification

The formal or statistical proof that a software product complies with its requirements relies on a strong discipline in the development process. Security requirements are challenging at these stages, because violating situations are difficult to simulate. WSN challenges arise for the major part from their distributed nature. Test, validation, and verification of distributed systems is intrinsically difficult due to size and complexity of the global state and the number of situations involving unknown environmental factors (commonly called the 'fallacies' of distributed systems). Beyond these intrinsic difficulties, WSN suffer from their large scales and wireless communication infrastructure, which both exacerbate the state spaces to test and verify.

Test methodologies based on unit tests and regressions are the primary approaches to expect in WSN, due to the large body of work and experience with test suites. Test units are appropriate to check the compliance of the software with specifications; security use cases from requirement to design phases can be checked as well. Disciplined structure of the units into suites should provide significant results with efforts concentrated on the application of existing techniques to security issues. Some additional techniques may be required to generate various 'tiny test cases' that arise from the range of situations faced by nodes in their environment. The software layer of these nodes must consistently react to inputs despite fast changing, and possibly unexpected, environmental conditions. Common techniques such as fuzz testing may provide grounds for such generation phase, but appropriate studies are necessary.

Simulations are alternative means to check security properties. Attacks can be reproduced to observe the reactions of the systems and provide some insightful information from simulation logs for further adjustment. Existing work from distributed system research can be applied to WSN, but current achievements require significant investment in simulation design, and specifics of WSN are usually not taken into account [85]. Such investment may not be possible, and WSN particularities can invalidate simulation attempts. Network simulators like *ns-2* [86] or the WSN-specific *TOSSIM* [87] show that WSN require specific care, but the results obtained from such tools should serve only for coarse testing, as they usually assume stringent restrictions on the environment or computation models and may invalidate their own results.

Verification of security properties can also be conducted formally with appropriate models and tools. Existing work like SPIN [88], SMV [89], BLAST [90], and SLAM [91] for C code conduct model-checking for verifying general safety or liveness properties, but there is little work referring to such formal verification in WSN research and practice to date. These tools can be applied to check security properties, provided adequate formulation of the target property. The case of SLAM can be particularly interesting, as it verifies whether programming interfaces are correctly used, which can then show when security measures (usually part of the API) are properly used by the software designer. Other languages like Java would rely on model-checkers like JPF [92], or Bandera/Bogol [93] (see section 5 for Java on WSN). Language-independent verification (thus exploitable early in the development process) are also available, with notable work for WSN with the real-time Maude language (not applied to security concerns to date) [94]. Shortcomings of these formal model tools come from two main issues. First, model-checking does not scale when state spaces are large. The large number of nodes and the unpredictable nature of WSN execution environments imply very large state spaces, despite the relatively small amount of code they run, so that model-checking approaches must be applied to restricted cases and results may not provide full property verification (e.g. liveness). The second issue originates in the limited achievements for checking distributed applications. Concurrency entails state-space explosion in addition to the effects of environment dynamics. Some approaches exist in general network applications [95], but they do not focus yet on WSN-like systems or on security issues.

### 4.4 Maintenance

Maintenance of software in WSN is an active research topic, but the security aspect is limited in present work. A notable example of maintenance mechanism is software update over the network, with proposals for reprogramming (progressive update) and use of mobile agent technologies [96], [97]. Deluge performs network reprogramming and was not initially integrating security measures [65]. A later version did however revise the system to add security [66]. Deluge is one of the few secured approaches for maintenance (see end of section 3.1 on recovery). The security concerns are however significant. Attackers who control some nodes can spread malicious code and gradually take control of all the network using update-based contamination. Appropriate counter-measures in most approaches remain open research issues.

Maintenance also refers to hardware management, including upgrade to new hardware, new firmware, or the necessary battery replacement which usually require turning off nodes temporarily. All these operations should be completed in secure ways to guarantee that no attack occurs at these sensitive times. Software protections are useful in such cases, as exemplified by security systems for, e.g., building access control, and specific protocols for WSN applications should be defined, with likely standard definitions.

### 4.5 Discussion

Research issues in security engineering for WSN are numerous and impact each stage of the software development process. Existing work is often applicable, but necessary adaptations or new approaches are also necessary for the major part. The main issue on this topic is indeed the difficulty to really achieve a holistic approach on security, as it should be accomplished to avoid the ‘forward-propagation’ of security problems, as can be observed in other software systems. As a result of the above analysis, we will present the consequences and needs in the development phase for which we have postponed the study, i.e. the implementation phase.

## 5 Engineering technology review and concrete needs of the security engineer

Engineering the security aspect of an application mainly relies on the availability and quality of tools. Although all tools along the software development process contribute to supporting the engineer, programming languages and related API (Application Programming Interface) for the implementation are among the most critical, because they serve to write the final product and should represent all specifications expressed in other development stages. In fact, high-level languages stemmed from the need to remove error-prone programming tasks from the hand of the programmers (e.g., goto statements, brittle assembly design patterns for control structures), so that they can focus on the core tasks of the target business logic.

The role of programming languages is essential in security engineering and the aim of this section is twofold. The first aim is to provide a review of the available technologies for the programmer, so as to demonstrate the current capabilities of existing work to facilitate and guarantee security requirements. The second aim is to set forth a number of supports that are not available, but that could lead to serious security issues with regards to the results of our survey in the previous sections.

To this end, we review the major platforms available, namely TinyOS on motes and Sun SPOT$^{TM}$ with respect to security engineering characteristics. The final part of this section presents aspects of security engineering that should arguably be integrated in current frameworks due to their impact in the context of WSN.

## 5.1 Major technologies

Wireless sensor network are supported by a limited number of technologies. We briefly review the representative TinyOS and Sun SPOT, with special focus on their security engineering facilities.

### 5.1.1 TinyOS on motes

TinyOS is an open source operating system tailored for low-power sensor network applications [98], [99]. The system was developed along WSN hardware called 'motes', with various commercial versions. The motes gather micro-controller, a number of sensors and LEDs, and batteries into a single package. The model of TinyOS is independent from the developed hardware and can be adapted to various platforms. In the short history of WSN technology, TinyOS has been the enabling technology[3], which fosters the deployment of applications and prototypes in various academic and industrial projects.

The technical aspects of TinyOS are concentrated on the optimization of the size and consumption of the system. The main features are a component-based architecture to foster code reuse and reduce the system footprint (less than 400 bytes in its earlier versions), an event-driven execution model, and integrated resource and power management in the latest version 2.0. The system and its applications are written in the nesC language [100], a variant of C providing a component-based framework and an appropriate subset of C concepts sufficient for low-power hardware (e.g., there is no function pointers in nesC). The system does not support preemption requests (only hardware-based interruptions), real-time scheduling, dynamic loading (applications are statically linked to the TinyOS core modules), but it does support data-link security with TinySec. Most notably, the system does not provide low-level protections yet, such as memory protection in its mainstream version [102] or on-the-fly verification of code downloaded over the air interface. Several extensions of TinyOS, such as SPINS presented in section 3, may however be integrated in the new releases of the system [7], [12].

[3] The Real-time Os Nucleus (TRON) was historically available from 1984, but its application to wireless sensor networks and broad availability are difficult to forecast today, despite the potential of the technology [101].

### 5.1.2 Sun SPOT

Sun SPOT is a product of Sun Microsystems, Inc. encompassing both hardware and software [103]. The project started in 2003 on the experience of the company with the technologies related to Java ME, and the first release occurred in April 2007. The recent release of the platform entails that the hardware provides among the most powerful sensor nodes, with similar size and scale factors to motes. The software part is independent from the hardware and consists of the Sun Squawk Java virtual machine [104]. Squawk is a closed-source JVM that encompasses necessary operating system functionalities, so that it can run directly on hardware [105].

The technical aspects of Squawk also aim at reducing the size and consumption of the system, similar to TinyOS, but it also seeks to bring a significant part of the Java model over WSN. Squawk is hence a full-fledged virtual machine that complies with the CLDC 1.1 specification. It can dynamically deploy code and run several applications concurrently. Although there is no reference on real-time issues yet, the platform provides security features borrowed from the standard Java model. We review the facilities provided by Sun SPOT at the lower and application levels. This review focuses on the software and excludes hardware-based security, as it was shown in section 3 that major issues lie in packet management and above the data link layer of the OSI model.

**Low-Level Security Facilities.** Low-level security mechanisms aim at protecting the system against erroneous or malicious code that exploits weaknesses in the software layers to harm or subvert the system. Squawk contains standard low-level security mechanisms from the Java model and specific extensions for WSN. The use of Java first provides built-in protections against code-based attacks that would exploit dangling pointers, pointer arithmetics, pointer forging, unchecked cast, or array boundaries. The virtual machine also provides a code verifier, which checks incoming code and asserts compliance with the Java standards before execution. These two mechanisms reflect the compatibility of Squawk with the CLDC security specification of Java ME [106]. They prevent a number of low-level security attacks, so that the system designers and programmers can focus on other security aspects.

In addition, Sun SPOT contains implementations of RSA and ECC (Elliptic Curve Cryptography) algorithms for security key management. The implementations have been optimized for sensor network nodes [107], providing programmers with essential cryptography mechanisms. In practice, the addition of these algorithms is essential on WSN as basic blocks for end-to-end security (see discussion in this section).

**Application-Level Security Facilities.** Application-level security mechanisms aim at providing application developers with appropriate abstractions for designing the security aspects of the target software. Java specifies the sandbox model to control how applications can access resources and additional mechanisms to define custom security access control policies. The compatibility of Squawk with the CLDC specification guarantees the availability of these mechanisms to developers.

The sandbox security model is complemented in Squawk by the architecture of the virtual machine, which handles applications as Java objects named 'isolates' [108]. Application isolation reinforces the low-level verification process of the virtual machine and extends security policies with isolate management. Each isolate can then be engineered with different restrictions on the creation, control, and communication with other isolates.

## 5.2 Discussion

Although Sun SPOT provides a security profile comparable to standard Java, we argue that this model will prove insufficient with regards to security engineering. WSN are inherently fragile systems due to eavesdropping and physical threats. We believe that the application engineering facilities provided by the platform should therefore go further than the current models, despite the difficult challenges of stringent resources and large-scale distribution. Extended models foster the engineering of strong security measures and avoid leaving the programmer with error-prone tasks.

Ongoing work in the WSN and ubiquitous computing research community, notably over the TinyOS platform, has proved the need for further endeavors [2], [4]. In particular, end-to-end mechanisms are among the most sensitive topics, and their absence from the core CLDC specification may lack for a Java-based approach if specialized profiles are not integrated in the specifications for WSN [109]. The addition by the Sun SPOT team of the cryptography algorithms provides the basic bricks for end-to-end security, but there remains a number of engineering issues.

### 5.2.1 Required engineering facilities

**Secure key establishment facilities.** How does the engineer deploy a key infrastructure on a WSN? This question is challenging and advanced mechanisms are available [110], [111], but it is possible to preset keys on nodes before deploying them. This 'imprinting' is necessary in any case to claim ownership of blank hardware [2], so one cannot expect simplifying or improving this security-critical task to the engineer.

The key establishment scenario becomes more interesting when introducing new nodes in an existing network. A node can be added to the network in an ad-hoc fashion, without direct human support, e.g., when new nodes are jettisoned from an aircraft to an existing land network. The software control layer must then provide facilities for dynamic authentication with peer nodes. Such introduction scenario is seemingly frequent enough and critical in the case of WSN to justify the integration of ready-to-use mechanisms for the engineer, even though application-specific requirements may require refinement of the basic solution. This would help increasing the engineer awareness and code security. Cryptographic bricks exemplify again the basic element for such a solution, but this remains an open research issue before considering efficient implementations in an API.

**Secure code mobility.** Code mobility platforms have permitted attractive demonstrations of sensor network technologies, with Agilla on top of TinyOS [97], and Sun SPOT. Squawk allows the migration of applications, which is a direct benefit of using the isolation concept and additional mechanisms such as transitive closures to form 'suites' of classes [104].

The security aspect is usually not discussed in these approaches, as it is seen out of scope. In fact, additional cryptography facilities such as the algorithms provided with Squawk on Sun SPOT provide the essential elements to enable secure code mobility. For the security engineer, it is however safer and more efficient to relate mobility and security directly in the programming interfaces. A secure version of the Sun SPOT API would then provide serialization methods over the air interface that explicitly states the desired security level. The Sun SPOT development team is indeed extending the radio API with a secured version of the radio stream class facility to serve this aim (information from their forum in June 2007 [103]).

**Decentralized security policy.** The original Java security policy model was designed with applet-like deployment in mind. In this model, an applet is downloaded on-demand by a client and run in the sandbox with appropriate access rights. This two-tier model is appropriate for certain applications, but it restricts the possible deployments, notably if WSN evolve to more ad-hoc infrastructures. In addition, changing the policy configuration of thousands of deployed nodes is a compelling example of functionality that calls for an evolution of the security model towards decentralized policy management.

Decentralization of the policy management implies the possibility for engineers to remotely log in to sensors with appropriate rights and modify their policies. We pointed out the requirements for multi-level security rights in section 2. The remote policy configuration is a typical example of such mechanisms. This work is not a trivial task and it is clearly a required functionality for standard deployments, thus justifying an integration in standard APIs.

Consistency of the policy throughout the network is one of the challenges of the decentralization. The aim is to guarantee that the access control rights remain the same for any principal, in any context. That is, the injection of a query to the network should return the same result, depending on the authenticated principal and the policy, but independent from the node and physical position where the query was sent.

**Measures against XSS-like attacks on sensor assets.** A compelling demonstration of WSN was for environment monitoring with the TinyDB application [70]. TinyDB is a database management system distributed over a WSN that collects information and aggregates results in-network. The security features were out of scope in this enabling technology, but the risks are significant.

It is common sense today that accepting unstructured string inputs are extreme security breaches that can lead to loss of data assets or even jeopardize a complete system. Unstructured input strings are famous issues in databases and popularized further with cross-scripting scripts over internet. Recommendations against this type of attack are to protect the application from unstructured strings with API facilities such as the `java.sql.PreparedStatement` methods in the case of SQL in Java, or the use of filters for websites against XSS. Such mechanisms should arguably be available in standard as applications that required database facilities are likely frequent with WSN.

**Engineering despite node capture.** The subversion of nodes is entirely part of security engineering, but quite challenging as it is mostly out of the hands of the software engineer. Existing work proposes mechanisms that rely on public key infrastructures, filtering false reports [83], [112], [113], or self-protection of the network [114]. Such mechanisms are not yet part of existing infrastructure or APIs to support the engineer, but concrete endeavors are underway [22]. Sensitive research issues remain, such as the fact that capturing *sufficient* nodes with signing keys opens gateways to the network.

The Sun SPOT platform already prevents compromised nodes from harming the system by deploying a signing key on the desktop connected to a base station only, and verification keys only on nodes. This asymmetric infrastructure avoids the risk to open gateways from any node, but this approach would not be applicable to more ad hoc network infrastructures, where signing keys would be necessary on vulnerable nodes.

Further research will therefore be necessary on this topic, which should also belong to the base toolkit for the security engineer. Tolerance to a number of corrupted nodes is already achieved by the schemes referred in this section, but engineering facilities are not available yet.

### 5.2.2 Limits to the integration

The limited resources of sensor nodes remain the major obstacle to deploy state-of-the-art security measures from standard research over WSN. Sensor nodes become more powerful, as exemplified by the Sun SPOT platform, but we have seen that some problems require specific security mechanisms in the standard engineering toolkit, while they may not be serious issues in other kinds of systems, e.g. careful communication management and decentralized policy infrastructures. On the other hand, some mechanisms can rightly be considered optional or application-dependent facilities.

Traffic analysis tools and security schemes based on trust models are for example two classes of mechanisms that may not be required in many applications. Traffic analysis tools could be part of testing and debugging suites, or administration packages, and trust models serve as alternative approaches in some authentication protocols [115]. More general trust models are also notably promising in applications that involve humans, owning to their flexibility and ease of use. A typical usage is when humans carry a mobile communication device [116], which converge increasingly with WSN by integrating sensors.

## 6 Conclusion

Security engineering in wireless sensor networks is a real issue that will expand as the technology gets momentum and spreads in industrial applications. Existing work is significant and tackles most of the challenging problems, but we have shown in this paper that a desired holistic approach to engineer security aspects is not possible yet. Some research areas on this topic remain in basic stages (relatively to WSN), e.g. specialized implementation frameworks and testing, and full-fledged integration of security measures are only partially addressing potential risks. Recent advances in both hardware and software let expect significant advances soon, but research endeavors remain necessary, notably in dealing with the challenging specificities of WSN as their large scales, the absence of global and reliable information on the network state, and the vulnerable nature of the nodes.

## References

[1] R. Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems*, John Wiley & Sons Inc., 2001.

[2] F. Stajano, *Security for ubiquitous computing*, Wiley, 2002.

[3] H. Chan and A. Perrig, "Security and privacy in sensor networks.," *IEEE Computer* vol.36, no. 10, pp. 103–105, 2003.

[4] A. Perrig, J. A. Stankovic, and D. Wagner, "Security in wireless sensor networks.," *Commun. ACM*, vol.47, no. 6, pp. 53–57, 2004.

[5] S. Avancha, J. Undercoffer, A. Joshi, and J. Pinkston, *Security for wireless sensor networks*, Norwell, MA, USA, Kluwer Academic Publishers, pp. 253–275, 2004.

[6] S. Hadim and N. Mohamed, Middleware: "Middleware challenges and approaches for wireless sensor networks.," *IEEE Distributed Systems Online*, vol.7, no. 3, 2006.

[7] A. Perrig, R. Szewczyk, V. Wen, D. E. Culler, and J. D. Tygar, *Spins: security protocols for sensor netowrks.*, MOBICOM, pp. 189–199, 2001.

[8] S. Zhu, S. Setia, and S. Jajodia, "Leap: efficient security mechanisms for large-scale distributed sensor networks," *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, New York, NY, USA, ACM Press, pp. 62–72, 2003.

[9] J. R. Douceur, "The sybil attack," IPTPS (Peter Druschel, M. Frans Kaashoek, and Antony I. T. Rowstron, eds.), Lecture Notes in Computer Science, vol. 2429, Springer, pp. 251–260, 2002.

[10] N. Subramanian, C. Yang, and W. Zhang, "Securing distributed data storage and retrieval in sensor networks," *PERCOM '07: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications*, Washington, DC, USA, IEEE Computer Society, pp. 191–200, 2007.

[11] M. Shao, S. Zhu, W. Zhang, and G. Cao, "pdcs: Security and privacy support for data-centric sensor networks," in *INFOCOM* [117], pp. 1298–1306.

[12] C. Karlof, N. Sastry, and D. Wagner, "Tinysec: a link layer security architecture for wireless sensor networks," *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, New York, NY, USA, ACM Press, pp. 162–175, 2004.

[13] M. Luk, G. Mezzour, A. Perrig, and V. D. Gligor, "Minisec: a secure sensor network communication architecture," in Abdelzaher et al. [118], pp. 479–488.

[14] D. Malan, M. Welsh, and M. Smith, "A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography," *Proceedings of IEEE International Conference on Sensor and Ad Hoc Communications and Network*, 2004, Available from: citeseer.ist.psu.edu/malan04publickey.html.

[15] A. S. Wander, N. Gura, H. Eberle, V. Gupta, and S. C. Shantz, "Energy analysis of public-key cryptography for wireless sensor networks," *PERCOM '05: Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications*, Washington, DC, USA, IEEE Computer Society, pp. 324–328, 2005.

[16] C. Kuo, M. Luk, R. Negi, and A. Perrig, *Message-in-a-bottle: User-friendly and secure key deployment for sensor nodes*, SenSys, 2007, pp. 233–246.

[17] W. Du, R. Wang, and P. Ning, "An efficient scheme for authenticating public keys in sensor networks," *MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, New York, NY, USA, ACM Press, pp. 58–67, 2005.

[18] D. Liu, P. Ning, and W. Du, "Group-based key predistribution in wireless sensor networks," *WiSe '05: Proceedings of the 4th ACM workshop on Wireless security*, New York, NY, USA, ACM Press, pp. 11–20, 2005.

[19] P. Traynor, R. Kumar, H. Bin Saad, G. Cao, and T. La Porta, "Liger: implementing efficient hybrid security mechanisms for heterogeneous sensor networks," *MobiSys '06: Proceedings of the 4th international conference on Mobile systems, applications and services*, New York, NY, USA, ACM Press, pp. 15–27, 2006.

[20] S. Yamashita, T. Shimura, K. Aiki, K. Ara, Y. Ogata, I. Shimokawa, T. Tanaka, H. Kuriyama, K. Shimada, and K. Yano, "A 15–15 mm, 1 $\mu$a, reliable sensor-net module: enabling application-specific nodes.," in Stankovic et al. [119], pp. 383–390.

[21] F. Ye, H. Yang, and Z. Liu, "Catching "moles" in sensor networks," *ICDCS*, IEEE Computer Society, p. 69, 2007.

[22] Y. Sei and S. Honiden, "Resilient security for false event detection without loss of legitimate events in wireless sensor networks," *The 9th International Symposium on Distributed Objects, Middleware, and Applications* (*DOA*), pp. 454–470, 2007.

[23] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," *MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, New York, NY, USA, ACM Press, pp. 46–57, 2005.

[24] M. Li, I. Koutsopoulos, and R. Poovendran, "Optimal jamming attacks and network defense policies in wireless sensor networks," in *INFOCOM* [117], pp. 1307–1315.

[25] W. Xu, K. Ma, W. Trappe, and Y. Zhang, "Jamming sensor networks: attack and defense strategies," *IEEE Network*, vol.20, no. 3, 41–47, 2006.

[26] V. S. Miller, "Use of elliptic curves in cryptography," *CRYPTO* (Hugh C. Williams, ed.), Lecture Notes in Computer Science, vol. 218, Springer, pp. 417–426, 1985.

[27] Sun Microsystems, Inc., “Sun Microsystems Announces Support for Elliptic Curve Cryptography” [online], 2005, Available from: http://www.sun.com/smi/Press/sunflash/2006-02/sunflash.20060214.2.xml [cited July 2007].

[28] National Security Agency (U.S.), “Fact Sheet NSA Suite B Cryptography” [online], 2005, Available from: http://www.nsa.gov/ia/industry/crypto_suite_b.cfm [cited September 2007].

[29] R. L. Rivest, A. Shamir, and L. M. Adleman, “A method for obtaining digital signatures and public-key cryptosystems.,” *Commun. ACM*, vol.21, no. 2, pp. 120–126, 1978.

[30] G. Gaubatz, J.-P. Kaps, and B. Sunar, “Public key cryptography in sensor networks - revisited,” in Castelluccia et al. [120], pp. 2–18.

[31] L. Uhsadel, A. Poschmann, and C. Paar, “An Efficient General Purpose Elliptic Curve Cryptography Module for Ubiquitous Sensor Networks,” *Software Performance Enhancement for Encryption and Decryption* (*SPEED 2007*), 2007.

[32] L. Batina, N. Mentens, K. Sakiyama, B. Preneel, and I. Verbauwhede, “Low-cost elliptic curve cryptography for wireless sensor networks,” in Buttyán et al. [121], pp. 6–17.

[33] E. Dahmen, K. Okeya, and T. Takagi, “An advanced method for joint scalar multiplications on memory constraint devices,” in Molva et al. [122], pp. 189–204.

[34] L. Uhsadel, A. Poschmann, and C. Paar, “Enabling full-size public-key algorithms on 8-bit sensor nodes,” in Stajano et al. [123], pp. 73–86.

[35] P. Tuyls, G. Jan Schrijen, B. Skoric, J. van Geloven, N. Verhaegh, and R. Wolters, “Read-proof hardware from protective coatings,” *CHES* (L. Goubin and M. Matsui, eds.), Lecture Notes in Computer Science, vol. 4249, Springer, pp. 369–383, 2006.

[36] D. Liu and P. Ning, “Location-based pairwise key establishments for static sensor networks,” *SASN ’03: Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, New York, NY, USA, ACM Press, pp. 72–82, 2003.

[37] Y.-H. Lee, V. Phadke, A. Deshmukh, and J. Wook Lee, “Key management in wireless sensor networks,” in Castelluccia et al. [120], pp. 190–204.

[38] D. Huang, M. Mehta, D. Medhi, and L. Harn, “Location-aware key management scheme for wireless sensor networks,” *SASN ’04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, New York, NY, USA, ACM Press, pp. 29–42, 2004.

[39] R. Di Pietro, L. V. Mancini, and A. Mei, “Energy efficient node-to-node authentication and communication confidentiality in wireless sensor networks,” *Wirel. Netw.*, vol.12, no. 6, pp. 709–721, 2006.

[40] J. Hwang and Y. Kim, “Revisiting random key pre-distribution schemes for wireless sensor networks,” *SASN ’04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, New York, NY, USA, ACM Press, pp. 43–52, 2004.

[41] S. Ahmet Çamtepe, B. Yener, and M. Yung, “Expander graph based key distribution mechanisms in wireless sensor networks,” *IEEE International Conference on Communications*, ICC ’06., pp. 2262–2267, 2006.

[42] L. Eschenauer and V. D. Gligor, “A key-management scheme for distributed sensor networks,” *CCS ’02: Proceedings of the 9th ACM conference on Computer and communications security*, New York, NY, USA, ACM Press, pp. 41–47, 2002.

[43] S. K. Ghosh, “On optimality of key pre-distribution schemes for distributed sensor networks,” in Buttyán et al. [121], pp. 121–135.

[44] T. Moore, J. Clulow, S. Nagaraja, and R. Anderson, “New strategies for revocation in ad-hoc networks,” in Stajano et al. [123], pp. 232–246.

[45] D. Wagner, “Resilient aggregation in sensor networks,” *SASN ’04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, New York, NY, USA, ACM Press, pp. 78–87, 2004.

[46] S. Roy, S. Setia, and S. Jajodia, “Attack-resilient hierarchical data aggregation in sensor networks,” *SASN ’06: Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks*, New York, NY, USA, ACM Press, pp. 71–82, 2006.

[47] H. Chan, A. Perrig, B. Przydatek, and D. Song, “Sia: Secure information aggregation in sensor networks,” *Journal of Computer Security*, vol.15, no. 1, 69–102, 2007.

[48] S. Capkun and J.-P. Hubaux, “Secure positioning of wireless devices with application to sensor networks,” *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, 13–17 March 2005, Miami, FL, USA, 2005, pp. 1917–1928.

[49] E. Schoch, F. Kargl, T. Leinmüller, S. Schlott, and P. Papadimitratos, “Impact of pseudonym changes on geographic routing in vanets,” in Buttyán et al. [121], pp. 43–57.

[50] M. Roughan and J. Arnold, “Multiple target localisation in sensor networks with location privacy,” in Stajano et al. [123], pp. 116–128.

[51] J. Clulow, G. P. Hancke, M. G. Kuhn, and T. Moore, “So near and yet so far: Distance-bounding attacks in wireless networks,” in Buttyán et al. [121], pp. 83–97.

[52] A. D. Wood and J. A. Stankovic, “Denial of service in sensor networks,” *Computer*, vol.35, no. 10, pp. 54–62, 2002.

[53] A. D. Wood, J. A. Stankovic, and S. H. Son, “Jam: A jammed-area mapping service for sensor networks,”

*RTSS '03: Proceedings of the 24th IEEE International Real-Time Systems Symposium*, Washington, DC, USA, IEEE Computer Society, p. 286, 2003.

[54] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: attacks and countermeasures.," *Ad Hoc Networks*, vol.1, no. 2–3, pp. 293–315, 2003.

[55] F. Armknecht, J. Girão, M. Stoecklin, and D. Westhoff, "Re-visited: Denial of service resilient access control for wireless sensor networks," in Buttyán et al. [120], pp. 18–31.

[56] A. Agah, M. Asadi, and S. K. Das, "Prevention of dos attack in sensor networks using repeated game theory," *ICWN* (Hamid R. Arabnia, Ed.), CSREA Press, pp. 29–36, 2006.

[57] S.-B. Lee and Y.-H. Choi, "A secure alternate path routing in sensor networks," *Comput. Commun.*, vol.30, no. 1, pp. 153–165, 2006.

[58] S.-B. Lee and Y.-H. Choi, "A resilient packet-forwarding scheme against maliciously packet-dropping nodes in sensor networks," *SASN '06: Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks*, New York, NY, USA, ACM Press, pp. 59–70, 2006.

[59] F. Ye, H. Luo, S. Lu, and L. Zhang, "Statistical en-route filtering of injected false data in sensor networks," *INFOCOM*, 2004.

[60] C. Krauß, F. Stumpf, and C. M. Eckert, "Detecting node compromise in hybrid wireless sensor networks using attestation techniques," in Stajano et al. [123], pp. 203–217.

[61] L. Buttyán, L. Dóra, and I. Vajda, "Statistical wormhole detection in sensor networks," in Molva et al. [122], pp. 128–141.

[62] A. Seshadri, M. Luk, A. Perrig, L. van Doorn, and P. Khosla, "Scuba: Secure code update by attestation in sensor networks," *WiSe '06: Proceedings of the 5th ACM workshop on Wireless security*, New York, NY, USA, ACM Press, pp. 85–94, 2006.

[63] M. Strasser and H. Vogt, "Autonomous and distributed node recovery in wireless sensor networks," *SASN '06: Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks*, New York, NY, USA, ACM Press, pp. 113–122, 2006.

[64] S. S. Kulkarni and L. Wang, "Mnp: Multihop network reprogramming service for sensor networks," in *ICDCS* [124], pp. 7–16.

[65] J. W. Hui and D. E. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," *SenSys* (J. A. Stankovic, A. Arora, and R. Govindan, Eds.), ACM, pp. 81–94, 2004.

[66] P. K. Dutta, J. W. Hui, D. C. Chu, and D. E. Culler, "Securing the deluge network programming system," in Stankovic et al. [119], pp. 326–333.

[67] A. Perrig, R. Canetti, D. Tygar, and D. Song, *The tesla broadcast authentication protocol*, 2002.

[68] M. Luk, A. Perrig, and B. Whillock, "Seven cardinal properties of sensor network broadcast authentication," *SASN '06: Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks*, New York, NY, USA, ACM Press, pp. 147–156, 2006.

[69] J. Girão, D. Westhoff, E. Mykletun, and T. Araki, "Tinypeds: Tiny persistent encrypted data storage in asynchronous wireless sensor networks," Ad Hoc Networks, vol.5, no. 7, 1073–1089, 2007.

[70] University of California, Berkeley, "TinyDB Homepage" [online], 2005, Available from: http://telegraph.cs.berkeley.edu/tinydb/ [cited August 2007].

[71] S. Slijepcevic, M. Potkonjak, V. Tsiatsis, S. Zimbeck, and M. B. Srivastava, "On communication security in wireless ad-hoc sensor networks," *WETICE '02: Proceedings of the 11th IEEE International Workshops on Enabling Technologies*, Washington, DC, USA, IEEE Computer Society, pp. 139–144, 2002.

[72] J. McDermott and C. Fox, "Using abuse case models for security requirements analysis," *ACSAC '99: Proceedings of the 15th Annual Computer Security Applications Conference*, Washington, DC, USA, IEEE Computer Society, 1999, p. 55.

[73] J. McDermott, "Abuse-case-based assurance arguments," *ACSAC '01: Proceedings of the 17th Annual Computer Security Applications Conference*, Washington, DC, USA, IEEE Computer Society, p. 366, 2001.

[74] G. Sindre and A. L. Opdahl, "Eliciting security requirements by misuse cases.," *TOOLS* (37), IEEE Computer Society, 2000, pp. 120–131, Available from: http://dblp.uni-trier.de/db/conf/tools/tools37-2000.html#SindreO00.

[75] G. Sindre and L. Opdahl, "Eliciting security requirements with misuse cases," *Requir. Eng.*, vol.10, no. 1, pp. 34–44, 2005.

[76] D. Firesmith, "Security user cases," *Journal of Object Technology*, vol.2, no. 3, pp. 53–64, 2003.

[77] D. Firesmith, "Specifying reusable security requirements," *Journal of Object Technology*, vol.3, (2004), no. 1, 61–75, Available from: http://www.jot.fm/jot/issues/issue_2004_01/column6.

[78] W.G. Schneeweiss, *The Fault Tree Method*, 1999.

[79] A. Hussey, "Hazop analysis of formal models of safety-critical interactive systems," *SAFECOMP '00: Proceedings of the 19th International Conference on Computer Safety, Reliability and Security*, London, UK, Springer-Verlag, pp. 371–381, 2000.

[80] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design patterns. elements of reusable object-oriented software*, Addison-Wesley, 1995.

[81] M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad, *Security patterns: Integrating security and systems engineering*, Wiley, 2006.

[82] C. Heath B. Blakley, *Security design patterns*, the Open Group, 2004.

[83] S. T. Halkidis, A. Chatzigeorgiou, and G. Stephanides, "A qualitative evaluation of security patterns.," *ICICS* (J. Lopez, S. Qing, and E. Okamoto, Eds.), Lecture Notes in Computer Science, vol. 3269, Springer, pp. 132–144, 2004.

[84] H. Yang, F. Ye, Y. Yuan, S. Lu, and W. Arbaugh, "Toward resilient security in wireless sensor networks," MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing, New York, NY, USA, ACM Press, pp. 34–45, 2005.

[85] M. J. Rutherford, A. Carzaniga, and A. L. Wolf, "Simulation-based test adequacy criteria for distributed systems.," *SIGSOFT FSE*, Michal Young and Premkumar T. Devanbu, eds., ACM, pp. 231–241, 2006.

[86] VINT Project Team, "The Network Simulator" [online], 2007, Available from: http://www.isi.edu/nsnam/ns/[cited September 2007].

[87] P. Levis, N. Lee, M. Welsh, and D. E. Culler, "Tossim: accurate and scalable simulation of entire tinyos applications.," *SenSys*, (Ian F. Akyildiz, Deborah Estrin, David E. Culler, and Mani B. Srivastava, eds.), ACM, pp. 126–137, 2003.

[88] G.J. Holzmann, "The model checker SPIN," *IEEE Trans. Softw. Eng.*, vol.23, no. 5, pp. 279–295, 1997.

[89] K. L. McMillan, *Symbolic model checking*, Kluwer Academic Publishers, 1993.

[90] T.A. Henzinger, R. Jhala, R. Majumdar, and G. Sutre, "Software verification with Blast," *Proceedings of the 10th SPIN Workshop on Model Checking Software*, pp. 235–239, 2003.

[91] T. Ball, B. Cook, V. Levin, and S.K. Rajamani, *SLAM and Static Driver Verifier: Technology transfer of formal methods inside Microsoft*, Integrated Formal Methods **2999** (2004), pp. 1–20.

[92] K. Havelund, "Using runtime analysis to guide model checking of java programs," *Proceedings of the 7th International SPIN Workshop on SPIN Model Checking and Software Verification*, London, UK, Springer-Verlag, pp. 245–264, 2000.

[93] J. Hatcliff and M. B. Dwyer, "Using the bandera tool set to model-check properties of concurrent java software," *CONCUR* (K. Guldstrand L. and M. Nielsen, Eds.), Lecture Notes in Computer Science, vol. 2154, Springer, pp. 39–58, 2001.

[94] P. C. Ölveczky and S. Thorvaldsen, "Formal modeling and analysis of wireless sensor network algorithms in real-time maude," *IPDPS*, IEEE, p. 157, 2006.

[95] C. Artho and P.-L. Garoche, "Accurate centralization for applying model checking on networked applications," *ASE*, IEEE Computer Society, pp. 177–188, 2006.

[96] A. Boulis, "SensorWare Homepage" [online], 2004, Available from: http://sensorware.sourceforge.net/ [cited September 2007].

[97] C.-L. Fok, G.-C. Roman, and C. Lu, "Rapid development and flexible deployment of adaptive wireless sensor network applications.," in *ICDCS* [124], pp. 653–662.

[98] J. L. Hill, *System architecture for wireless sensor networks*, Ph.D. thesis, University of California, Berkeley, 2003.

[99] University of California, Berkeley, "TinyOS Homepage" [online], Since 2004, Available from: http://www.tinyos.net/ [cited August 2007].

[100] D. Gay, P. Levis, J. R. von Behren, M. Welsh, E. A. Brewer, and D. E. Culler, "The nesC language: A holistic approach to networked embedded systems.," *PLDI*, ACM, pp. 1–11, 2003.

[101] K. Sakamura, "TRON Project Homepage" [online], Since 1984, Available from: http://www.tron.org/index-e.html [cited August 2007].

[102] R. Kumar, E. Kohler, and M. B. Srivastava, "Harbor: software-based memory protection for sensor nodes.," in Abdelzaher et al. [118], pp. 340–349.

[103] Sun Microsystems, Inc., "Project Sun SPOT Homepage" [online], 2007, Available from: http://www.sunspotworld.com/ [cited August 2007].

[104] Sun Microsystems, Inc., "Squawk Virtual Machine Homepage" [online], 2003, Available from: http://research.sun.com/projects/squawk/squawk-rjvm.html [cited August 2007].

[105] N. Shaylor, D. N. Simon, and W. R. Bush, "A java virtual machine architecture for very small devices.," *LCTES*, ACM, pp. 34–41, 2003.

[106] Q. Mahmoud, "Wireless Java Security" [online], 2002, Available from: http://developers.sun.com/mobility/midp/articles/security/ [cited August 2007].

[107] N. Gura, A. Patel, A. Wander, H. Eberle, and S. Chang Shantz, "Comparing elliptic curve cryptography and rsa on 8-bit cpus.," *CHES* (M. Joye and J.-J. Quisquater, Eds.), Lecture Notes in Computer Science, vol. 3156, Springer, pp. 119–132, 2004.

[108] G. Czajkowski, "Application isolation in the java$^{tm}$ virtual machine," *OOPSLA*, pp. 354–366, 2000.

[109] Sun Microsystems, Inc., "Connected limited device configuration, specification version 1.1," 2003.

[110] H. Chan and A. Perrig, "Pike: peer intermediaries for key establishment in sensor networks.," *INFOCOM*, IEEE, pp. 524–535, 2005.

[111] F. Delgosha and F. Fekri, "Threshold key-establishment in distributed sensor networks using a multivariate scheme.," in *INFOCOM* [125].

[112] Z. Yu and Y. Guan, "A dynamic en-route scheme for filtering false data injection in wireless sensor networks," SenSys '05: Proceedings of the 3rd interna-

tional conference on Embedded networked sensor systems (New York, NY, USA), ACM Press, pp. 294–295, 2005.

[113] K. Ren, W. Lou, and Y. Zhang, "Leds: Providing location-aware end-to-end data security in wireless sensor networks," in *INFOCOM* [125].

[114] D. Wang, Q. Zhang, and J. Liu, "Self-protection for wireless sensor networks.," *ICDCS*, IEEE Computer Society, p. 67, 2006.

[115] S. Fouladgar, B. Mainaud, K. Masmoudi, and H. Afifi, "Tiny 3-tls: A trust delegation protocol for wireless sensor networks," in Buttyán et al. [121], pp. 32–42.

[116] D. Quercia, S. Hailes, and L. Capra, "B-trust: Bayesian trust framework for pervasive computing.," *iTrust* (K. Stølen, W. H. Winsborough, F. Martinelli, and F. Massacci, eds.), Lecture Notes in Computer Science, vol. 3986, Springer, pp. 298–312, 2006.

[117] *Infocom 2007. 26th ieee international conference on computer communications, joint conference of the ieee computer and communications societies, 6-12 May 2007, Anchorage, Alaska, USA*, IEEE, 2007.

[118] T. F. Abdelzaher, L. J. Guibas, and M. Welsh (eds.), *Proceedings of the 6th international conference on information processing in sensor networks, ipsn 2007, Cambridge, Massachusetts, USA, april 25–27, 2007*, ACM, 2007.

[119] J. A. Stankovic, P. B. Gibbons, S. B. Wicker, and J. A. Paradiso (eds.), *Proceedings of the fifth international conference on information processing in sensor networks, ipsn 2006, Nashville, Tennessee, USA, april 19-21, 2006*, ACM, 2006.

[120] C. Castelluccia, H. Hartenstein, C. Paar, and D. Westhoff (eds.), *Security in ad-hoc and sensor networks, first european workshop, esas 2004, heidelberg, germany, august 6, 2004, revised selected papers*, Lecture Notes in Computer Science, vol. 3313, Springer, 2005.

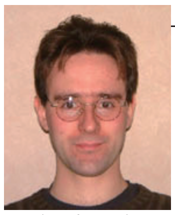
[121] L. Buttyán, V. D. Gligor, and D. Westhoff (eds.), *Security and privacy in ad-hoc and sensor networks, third european workshop, esas 2006, hamburg, germany, september 20-21, 2006, revised selected papers*, Lecture Notes in Computer Science, vol. 4357, Springer, 2006.

[122] R. Molva, G. Tsudik, and D. Westhoff (eds.), *Security and privacy in ad-hoc and sensor networks, second european workshop, esas 2005, visegrad, hungary, july 13-14, 2005, revised selected papers*, Lecture Notes in Computer Science, vol. 3813, Springer, 2005.

[123] F. Stajano, C. Meadows, S. Capkun, and T. Moore (eds.), *Security and privacy in ad-hoc and sensor networks, 4th european workshop, esas 2007, cambridge, uk, july 2–3, 2007, proceedings*, Lecture Notes in Computer Science, vol. 4572, Springer, 2007.

[124] *25th international conference on distributed computing systems (icdcs 2005), 6-10 June 2005, Columbus, OH, USA*, IEEE Computer Society, 2005.

[125] *Infocom 2006. 25th ieee international conference on computer communications, joint conference of the ieee computer and communications societies, 23–29 April 2006, Barcelona, Catalunya, Spain*, IEEE, 2006.



## Eric PLATON

Eric PLATON is a visiting researcher at the National Institute of Informatics. He holds two M.Sc. degrees in Automatic Control and Distributed Artificial Intelligence respectively, and a joint Ph.D. degree from Paris 6 University and Sokendai. His research interests pertain to distributed systems, with particular focus on security, self-adaptive programs, and exception handling in mobile systems, notably wireless ad hoc networks of sensors.

## Yuichi SEI

Yuichi SEI is a PhD student of The University of Tokyo. He is also a research assistant at the National Institute of Informatics. He received his MS degree in 2006. His research interests are in wireless networking, distributed algorithms, and security in wireless sensor networks.