Enlighten Me! Secure Key Assignment in Wireless Sensor Networks

Matthias Gauger
Universität Bonn, Bonn, Germany and
Fraunhofer IAIS, St. Augustin, Germany
gauger@cs.uni-bonn.de

Olga Saukh
Structural Engineering Research
Laboratory, EMPA, Switzerland
olga.saukh@empa.ch

Pedro José Marrón
Universität Bonn, Bonn, Germany and
Fraunhofer IAIS, St. Augustin, Germany
pjmarron@cs.uni-bonn.de

***Abstract*—The availability of secret keys is a precondition for the use of many security solutions and protocols. However, securely assigning such keys to nodes is a challenging task in the context of wireless sensor networks. In this paper we present a novel solution for a secure key assignment in wireless sensor networks that can be used during the initial configuration of nodes or for an ad-hoc key assignment by mobile nodes. The idea is to transmit the key information over a side channel using a controllable light source as the sender and the light sensors available on wireless sensor nodes as receivers. We demonstrate that our solution fulfills the relevant security requirements while at the same time being cost effective and easy to use.**

I. Introduction

With wireless sensor networks slowly moving from being pure research platforms to becoming used in small and medium-scale commercial deployments, the security of these systems is becoming an increasingly important factor: Once wireless sensor networks carry mission-critical or sensitive data, it is essential that their communication cannot be tampered with by malicious entities. However, as for most other aspects of system operation, the strong resource constraints of the individual sensor nodes as well as the ad-hoc nature of their deployment make achieving security goals in wireless sensor networks a challenging task.

The availability of key information is an important aspect in realizing security as most security solutions and protocols require shared secrets or some kind of secret key information to provide security services like authentication, integrity guarantees or encryption. While it is generally possible to exchange key information on-demand using a key exchange protocol (e.g., [1], [2], [3]), nodes still need to be equipped with a certain amount of initial key data that can be used by such protocols in establishing and securely exchanging dynamic session keys.

Assigning secret key information to individual nodes in wireless sensor networks is a difficult problem. On the one hand, this is due to the limited interfacing capabilities of typical sensor node hardware. On the other hand, securely assigning keys to nodes is complicated by the fact that the standard communication method in wireless sensor networks – sending RF messages over the wireless medium – is inherently insecure: Any interested party within communication range can listen to messages or inject its own spurious messages.

Preloading key information at node programming time or transmitting the key data over a wired interface provided by the sensor node (e.g, a USB port or a serial interface) is no alternative, because we expect that future sensor nodes (or devices with similar capabilities) – unlike today's sensor network research platforms – will be delivered preprogrammed from the manufacturer (who might not be willing to deal with key management and key security for its customers) and will not necessarily provide a wired interface.

In this paper we present *Enlighten Me!*, a novel approach to implement secure key assignment in wireless sensor networks. The basic idea of *Enlighten Me!* is to transmit secret key information over a separate communication channel (a side channel) which allows to effectively limit the set of listeners to legitimate receiver nodes. However, for the approach to be implemented on standard wireless sensor nodes, we cannot require additional hardware for this separate communication channel. Our solution is to communicate using light with a controllable light source used to generate the data signal and using the light sensors available on many sensor node platforms to capture and record the signal. To securely transmit key information over this light communication channel, we have developed a simple, yet efficient communication protocol that can be implemented on top of the standard operating system abstractions provided by TinyOS.

We have developed two different key sender devices for *Enlighten Me!*. The first device, the Sensor Node Lamp, uses hardware specifically developed for this purpose and sends key information using a powerful LED as its light source. The second device, the *Enlighten Me!* PDA, is based on standard PDA hardware and uses the display of the PDA to transmit key information to a sensor node by varying the light level on the display.

Our solution provides the following set of advantages. Firstly, it effectively prevents RF-based eavesdropping in the transmission of key data from the key sender to the key receiver. Secondly, the system is easy and convenient to use and provides meaningful feedback to the user. Thirdly, it allows key assignment both as part of the initial network configuration and in an ad-hoc manner during normal system operation. Finally, it is a cost effective solution that does not require additional hardware on the wireless sensor nodes.

The rest of this paper is organized as follows. In the next

978-1-4244-5113-5/09/$25.00 ©2009 IEEE

section, we discuss important related work from the area of security in wireless sensor networks in general and key assignment and key exchange in particular. Section III gives an overview of the different aspects of our key assignment method before Section IV describes the protocol in detail. Section V describes the two different key sender devices and specifics of the key receiver implementation. We then present the evaluation of our approach in Section VI before Section VII concludes this paper with a summary and an outlook of future work.

II. Related Work

A growing set of security solutions and protocols is available for sensor networks dealing with issues as diverse as communication security [1], security of routing protocols [4] or even secure node localization [5]. All these approaches rely on basic cryptographic mechanisms that require shared secrets or key data as it can be assigned to sensor nodes using the mechanism we present in this paper.

Among the existing sensor network security solutions, key distribution mechanisms [1], [2], [6], [7], [3] are particularly related to our approach. Their fundamental goal is to allow for a secure communication among any pair of nodes in the network by dynamically exchanging key information. Basic solutions rely on a centralized entity with whom all nodes share pairwise keys and which acts as a mediator in the key exchange process (e.g., [1]). To overcome the need for a central authority, several authors have proposed to predistribute sets of keys to the nodes in the network either randomly (e.g., [2], [6]) or in a controlled fashion (e.g., [7], [3]). These sets of keys are then used to dynamically create session keys for node-to-node communication.

Multiple authors have proposed to transmit key data or authentication data over a privileged side channel. **Talking to Strangers** [8] describes a general protocol that assumes the availability of a "location-limited channel" that allows to identify devices based on their physical context (e.g., the sender must be in the same area as the receiver). The authors do not assume that secrecy is being provided on the location-limited channel. Consequently, they exchange public keys over the normal wireless channel and use the location-limited channel for authentication and to check the key integrity.

Stajano and Anderson [9] propose to require physical contact among devices for the initial transmission of key information. They then transmit the key over an electrical contact – requiring special hardware support both on the sender and on the receiver side.

The idea of **Message-In-a-Bottle** [10] is to place key sender and key receiver together in a Faraday cage that prevents outside eavesdroppers from overhearing the messages exchanged over the radio channel. An additional node placed outside of the cage supervises the protocol and also jams the wireless channel to overshadow any signals not blocked by the Faraday cage. The major disadvantage of Message-In-a-Bottle is its need for direct physical access to the nodes as they have to be placed inside of the Faraday cage. Consequently, it is not well-suited for an on-demand assignment of keys after the nodes have been deployed.

Shake Them Up! [11] securely exchanges key information between two entities A and B by sending messages between them over an anonymous channel that hides the identity of the message sender. Both nodes send random sequences of messages claiming to be either node A or B. Due to the anonymity of the channel, only the respective communication partner can check the validity of the claim and can convert this into a single bit of key information. Shake Them Up! implements an anonymous channel by randomizing the message send times, by operating both nodes with the same transmission power and by requiring the user to shake the nodes together during the key exchange to prevent determining the sender of a message by analyzing the strength of the received signal. One limitation of the approach is that it might be difficult to provide for identical signal strengths and radio properties with different nodes in the context of wireless sensor networks. Moreover, shaking nodes is an exhausting task and might tempt the user to neglect this important part of the secure key exchange.

III. Enlighten Me! Overview

This section provides an overview of the *Enlighten Me!* key assignment mechanism including its goals and application scenarios, the procedure of using *Enlighten Me!* and a discussion of the attacker model we are building on.

A. Goals and Application Scenarios

The primary goal of *Enlighten Me!* is to provide a secure mechanism for the assignment of keys to wireless sensor nodes. In order to achieve this, *Enlighten Me!* needs to prevent the overhearing of key data transmitted from a key sender to a key receiver. Moreover, it needs to prevent attackers from covertly manipulating transmitted key information or injecting spurious keys. We will go into more details of this when we define the attacker model below.

Ease and efficiency of use are important secondary goals of *Enlighten Me!*. The system should require only little user input, provide immediate and easy-to-understand feedback and should be tolerant to user errors.

Finally, our key assignment approach should not require any special hardware on the sensor nodes as this would increase the node costs, impede the portability to other platforms and, in general, limit the number of application scenarios for *Enlighten Me!*. We mostly achieve this by only requiring a simple light sensor on the receiver node – a feature found on many sensor node platforms today.

We aim at two different application scenarios with our method for key assignment in wireless sensor networks: Our main goal is to support the initial assignment of key data as part of configuring a wireless sensor network. As a second application scenario, *Enlighten Me!* allows mobile users the dynamic assignment of keys to nodes already deployed in the environment.

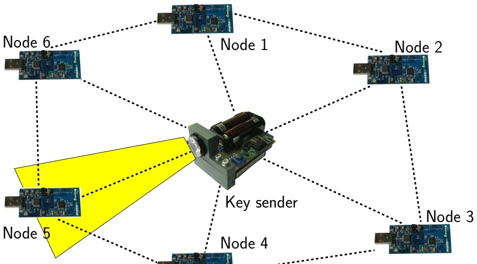


Fig. 1. Basic communication principle

B. Using Enlighten Me!

The idea of *Enlighten Me!* is to secure the transmission of key information in a wireless sensor network by using a separate second communication channel based on light whose recipients the user is able to control. While *Enlighten Me!* exchanges its protocol messages over the normal radio channel, the key data is sent exclusively over the light channel.

The *Enlighten Me!* system comprises two different types of nodes, a set of **key receiver** nodes and one or multiple **key sender** nodes. The key receiver nodes are standard wireless sensor nodes possibly already deployed in the application area. They detect and record the light signal using their integrated light sensors. The key sender is a mobile device carried by an administrator or a normal user. It integrates a controllable light source used to generate the light signal for transmitting the key information.

Fig. 1 illustrates the fundamental working principle of *Enlighten Me!*: They key sender in the middle intends to assign a key to an individual node (node 5) but a total of six nodes lie within the transmission range of its RF transceiver. While the key sender cannot prevent any of these nodes to overhear and record the radio messages transmitted, it is able to control the propagation of the light signal and make sure that only node 5 is able to receive the key information transmitted as a light signal.

C. Attacker Model

Let us now introduce our attacker model and discuss which types of attacks our key assignment scheme is supposed to withstand. We also list the types of attacks that our approach is not able to deal with and justify why this is an acceptable limitation in our scenarios.

1) Passive and Active Attacks: We assume that both passive and active attacks on the wireless communication channel are possible and must be dealt with by the *Enlighten Me!* system.

In a passive attack, an adversary quietly eavesdrops on the message exchange of the nodes in the network and records relevant data. From the viewpoint of our system, passive attacks are not critical as no secret information is transmitted over the radio communication interface. By listening to the message traffic, the attacker is able to overhear the key exchange protocol but never record the key itself.

In an active attack, an adversary actively participates in the communication, for example by injecting spurious messages, replaying messages it has received or by forwarding manipulated message content. Active attacks on our system are possible with the help of fake protocol messages. However, as will be shown in the detailed protocol description in Section IV-B, all phases of the protocol involve the use of the light communication channel so that fake protocol messages on the radio channel quickly lead to a timeout on the receiver side. Consequently, an adversary can only cause short disruptions of the system operation. Sending a very large number of fake protocol messages in a short time interval leads to a denial-of-service attack which we discuss below.

2) Denial-of-Service Attacks: In the context of key assignment, a denial-of-service attack prevents the successful transmission of key information from a sender to a receiver. One possible way for an adversary to do this is to jam the wireless medium in the area or to flood the nodes with useless messages. Usually, it is possible to detect such attacks [12].

It has been shown that dealing with denial-of-service attacks in wireless sensor networks is a very complex problem that has to be addressed for all aspects of the system at design time [13]. Consequently, we do not explicitly deal with denial-of-service attacks but rely on an underlying service that ensures communication capability among the nodes of our system.

3) Physical Access to Nodes: It is usually impossible to deal with attackers that have physical access to the nodes as they are able to read out the memory (including all key information), reprogram or even exchange the nodes. However, we are able to deal with adversaries operating in the vicinity of the sensor nodes.

An adversary might try to write his own key information to a sensor node using the *Enlighten Me!* protocol. We can prevent this by only allowing to set the keys once at the beginning of the system operation. If an entity wants to exchange this key later on, it needs to provide the old key first.

If keys are assigned dynamically on a per-client basis, it is generally acceptable for any mobile node to assign a key to a sensor node as this key is only used for securing the communication between this pair of nodes.

4) Attacks on the Light Channel: The fundamental assumption of *Enlighten Me!* is that key information can be securely transmitted over the light channel whereas attackers might have passive or active access to the radio communication channel. This assumption holds if the user has control over the reception area of the light signal and no attacker can jam the light signal or even directly block the line of sight. We will discuss the security of the light channel separately for our two key sender devices in Section V.

IV. The Enlighten Me! Protocol

This section describes the details of the *Enlighten Me!* protocol including the message encoding and decoding, the individual protocol steps and the protocol behavior in case of errors.

A. Message Encoding and Decoding

We aim to transmit key data using a light signal emitted by a light source on the key sender and recorded with the help of

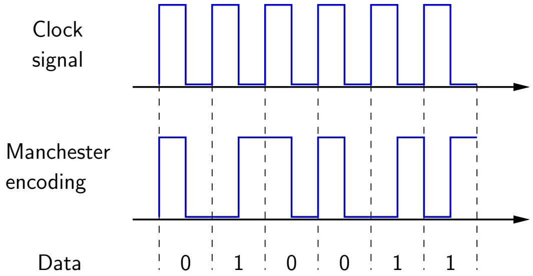


Fig. 2. Manchester encoding example

a light sensor on the receiver node. The task of the message encoding is to convert the bitstream of the key data into such a light signal. The decoding then performs the inverse operation, reconstructing the original data bitstream out of the received light signal. Two different signal states can be used in the encoding, 'light on' and 'light off'.

For the encoding of our data, we use the Manchester code, a relatively simple but robust digital baseband modulation code [14]. As illustrated by an example in Fig. 2, it transmits exactly one bit per clock cycle and encodes a one as a signal level transition from low to high at the middle of the clock cycle whereas a zero is encoded using a transition from high to low.

An important advantage of the Manchester code in our setting lies in its self-timing property: The code ensures that a signal level transition occurs at least once per clock cycle which allows the receiver to easily synchronize (and resynchronize) to the clock used by the sender for encoding the signal.

There are two parameters that we need to set for the encoding of data: the period of the clock signal Δ_{clock} and the threshold distance δ_{Thresh} used to differentiate between a 'light on' and a 'light off' signal. The smaller Δ_{clock}, the higher is the throughput of the key transmission. However, limiting factors in this selection include the switching time of the light source, the reaction time of the light sensors, interrupt handling times, times required for processing in software and the limited accuracy of the software timers.

Before the key transmission starts, the receiver captures its current ambient light level l_{Base} which represents the 'light off' state. A predetermined threshold value δ_{Thresh} is then used to differentiate between the two states as follows: Every light sensor reading l_t at time t is considered to belong to the 'light on' state iff $l_t \geq l_{Thresh} = l_{Base} + \delta_{Thresh}$ holds. Otherwise, it belongs to the 'light off' state. Note that the value for δ_{Thresh} needs to be determined separately for each type of light sensor used on different nodes.

B. Key Assignment Protocol

The key assignment protocol consists of four phases. In the first phase, the **handshake phase**, the key sender advertises the upcoming key assignment and sender and receiver find each other. The key transmission is prepared in the second phase, the **initialization phase**, during which the key sender specifies parameters of the key transmission. The actual transmission of the key information is performed in the **key distribution phase** before the correct completion of the transmission is checked in the **key verification phase**.

The time diagram of the key assignment protocol in Fig. 3 illustrates the sequence of events of a successful key transmission for the simple example key '1001'. The following explanation of the individual protocol steps refers to events numbered in the time diagram in Fig. 3.

The key sender starts the protocol with the handshake phase by broadcasting a handshake trigger message (1) which notifies the nodes in the neighborhood of the imminent key transmission. These candidate receiver nodes react by sampling their base sensor value l_{Base} (2) used to differentiate between the 'light on' and the 'light off' states and then continue sampling their light sensors every $t_{SamplePeriod}$ time units.

The key sender meanwhile waits for $t_{HSDelay}$ time units and then activates its light source (3) to provide a sensor stimulus to the key receiver. The key receiver detects this 'light on' event (4) and sends a handshake reply message back to the key sender (5). All other candidate receiver nodes not lying within the light beam time out after $t_{HSTimeoutReceiver}$ time units, stop sampling their light sensor and do not participate any further in the key assignment process.

After receiving the handshake reply message (6), the key sender starts the initialization phase by sending a key announcement message to the key receiver (7) which contains information on the size of the key. Receiving this message (9) triggers the key receiver to sampling its light sensor again while the key sender proceeds by sending a 1 on the light channel (8). This 1 in Manchester encoding consists of a low signal in the first half of the clock cycle and a high signal in the second half. The change from low to high allows the receiver to synchronize itself to the clock of the incoming light signal.

The initialization phase is directly followed by the key distribution phase when the key sender starts transmitting the encoded key data over the light channel after it has completed sending out the 1 (10). The key distribution phase then ends when the key sender has sent out all bits of the key data and stops the transmission turning off its light source.

After the key receiver has received all bits of the key data (the size was specified in the key announcement message), it stops sampling its light sensor and starts the key verification phase by sending the key confirmation message (11). The key sender uses the content of this message to verify the correct transmission of the key (12) and confirms this in a key acknowledgment message (13). The key receiver finally assigns the new key when this acknowledgment has been received (14).

The details of the key verification procedure and the structure and content of the last two messages depend on the type of keys and the security protocol used. If symmetric encryption is used, we propose the following procedure based on the challenge-response principle: The key sender sends a randomly generated number I_{chall} encrypted with the new key K as part

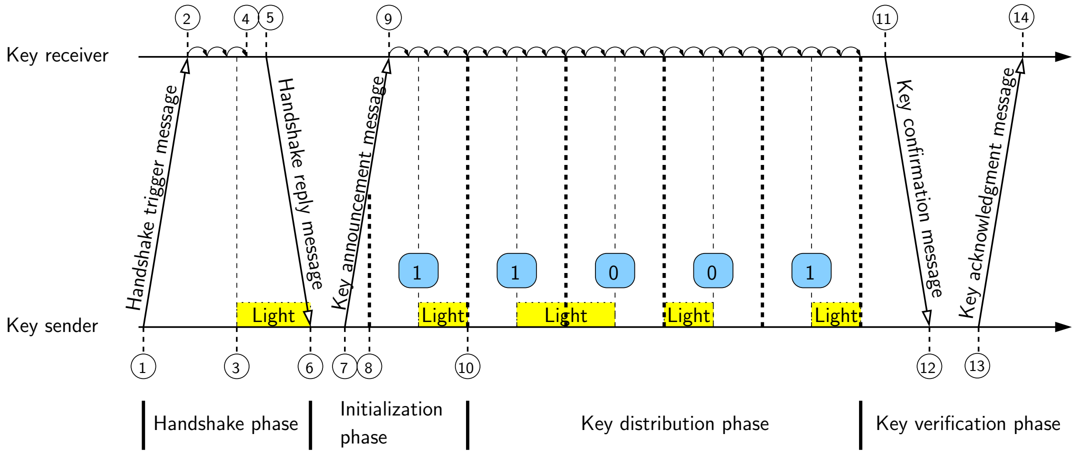


Fig. 3. Key assignment protocol time diagram

of the key announcement message to the key receiver.

$$Key\ announcement: \quad S \rightarrow R : Keysize, \{I_{chall}\}^K$$

Once the key receiver has successfully received the key over the light channel, it is able to decrypt the message, decrement I_{chall} by one, encrypt it again using key K and send it back to the key sender in the key confirmation message.

$$Key\ confirmation: \quad R \rightarrow S : \{I_{chall} - 1\}^K$$

This proves to the key sender that the key receiver has correctly received K. The key sender can now confirm this using again I_{chall} and sends back $I_{chall} - 2$ encrypted by K.

$$Key\ acknowledgment: \quad S \rightarrow R : \{I_{chall} - 2\}^K$$

C. Behavior in case of errors

So far, the protocol description assumed that no errors occur. However, the protocol behavior is also clearly defined in case of errors.

The most common error occurs during the key transmission over the light channel. Internal or external interferences can prevent the signal from being correctly received by the key receiver node. Due to the properties of the Manchester code, such errors hardly cause bit errors but rather a code violation that disrupts the signal detection. The key receiver reports such an abort back to the key sender which repeats the key transmission up to two times starting again with the key announcement message. As the protocol can detect erroneous keys (caused by bit errors) during the key verification phase, we neither included a CRC mechanism nor forward error correction into the key transmission phase.

Another type of error occurs if the light source is not pointed in the direction of the key receiver node or the light signal is too weak. In this case, the protocol fails during the handshake phase as no handshake reply message is received within $t_{HSTimeoutSender}$. The key sender device then provides feedback to the user and returns to its initial state waiting for further key assignment requests.

V. Enlighten Me! Systems

In this section, we describe two different systems that implement the key sender functionality of the *Enlighten Me!* protocol: The Sensor Node Lamp and the *Enlighten Me!* PDA. At the end of this section, we also introduce the key receiver implementation and describe how it relates to the two key sender approaches.

A. Sensor Node Lamp

The idea of the Sensor Node Lamp approach is to provide a special, dedicated hardware device, the Sensor Node Lamp (SNL), as the key sender for the *Enlighten Me!* protocol. The SNL looks a little like a flashlight and is also used in a very similar manner: It is equipped with a strong LED as its light source and allows to support key assignment both over small and also larger distances by simply pointing the light source in direction of the key receiver node.

1) Hardware: We have built a prototype of the Sensor Node Lamp as an extension board that can be mounted on top of a TelosB sensor node (see Fig. 4). The main component of the SNL is a powerful 1 watt LED fixed behind a lens that focuses the light in a beam with a specified cone angle of 8.7 degrees. The LED we use emits red light at a dominant wavelength of 625 nanometers which lies within the optimal reception range of the light sensors on the TelosB sensor nodes. The SNL also provides a button for user input to the application, a separate power supply in the form of two AA batteries and a constant current transformer used to provide for a constant light level of the LED over the lifetime of the batteries.

2) Flashlight Analogy: With its LED light source, the lens focusing the light into a light beam and the button used to control the lamp, the SNL does not only look like a flashlight, it is also used in a very similar manner: Assigning a key to a node is as simple as illuminating the node with the SNL for a few seconds. This, as we call it, **flashlight analogy** is an important aspect for the usability of the SNL as a flashlight is a

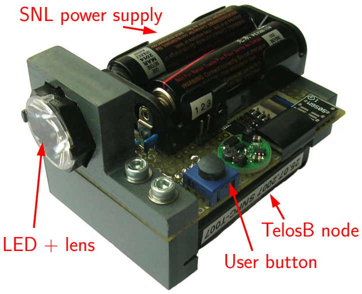


Fig. 4. Sensor Node Lamp

well-known device most users are familiar with. Being able to explain the key assignment process starting from the flashlight analogy greatly simplifies the explanation of the device and its working principle to new users.

Another important aspect is the visual feedback provided by the light beam of the SNL. It not only helps in pointing the device in the right direction, it also gives hints on the size of the reception area and allows the user to adjust this based on the node density or nearby adversaries.

3) Implementation: We have implemented the key sender functionality on the SNL in TinyOS 2.1. We also developed a simple auxiliary application that allows to generate keys on a PC or a PDA and then upload them to the SNL over a USB connection.

For the SNL implementation, we extended the *Enlighten Me!* protocol with an additional phase in the beginning, the **aiming phase**. The goal of this phase is to support the user in aiming the light beam in direction of the key receiver node. For this purpose, pressing the user button on the SNL activates the LED and allows the user to direct the light beam in the right direction. When he releases the user button again, the aiming phase ends, the LED is deactivated again and the actual *Enlighten Me!* protocol starts.

To illustrate the key transmission process with the SNL, Fig. 5 shows the light levels of an example key transmission of the 16 bit key value 54613 with $\Delta_{clock} = 150ms$. One can clearly identify the aiming phase at the beginning of the recording as well as both long and short 'light on' and 'light off' time intervals.

Fig. 5 also shows an interesting issue that complicates the signal decoding on the receiver side: Under seemingly stable external conditions, the light levels recorded by a receiver node oscillate significantly both during the 'light on' and the 'light off' states of the SNL. We found the reason for this behavior in the fluorescent tubes used instead of standard light bulbs for the room lighting in many public buildings. Instead of providing a constant light level, their intensity oscillates with a frequency of 100 Hz (or 120 Hz depending on the frequency of the electrical system). While this effect is usually not noticeable for the human eye, it strongly influences the values recorded by the light sensors resulting in oscillations like shown in Fig. 5.

Continuous variations of the recorded light values can interfere with the detection of the light signal if they cause untimely 'light on' or 'light off' events. This is particularly critical if we select a small value for δ_{Thresh} which might be necessary to provide for the operation of the Sensor Node Lamp over larger distances or in scenarios with a high ambient light level. This is illustrated in Fig. 6 (a) where the base value l_{Base} lies near the minimum value of the current light oscillation l_{min}. Samples taken in the shaded area are now detected as belonging to the 'light on' state without the SNL being turned on.

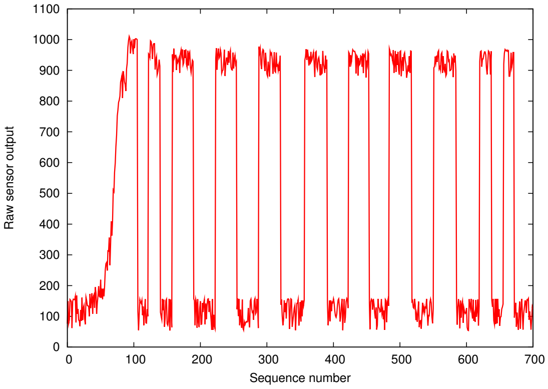
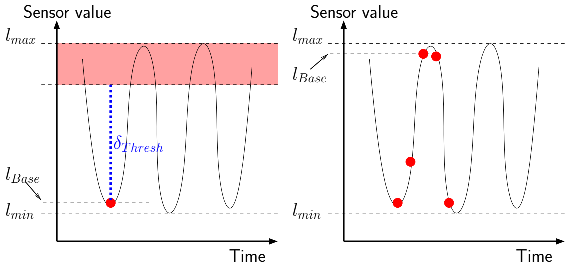


Fig. 5. Light level example SNL



(a)

(b)

Fig. 6. Effects of light oscillations on base value sampling

We deal with this problem using an extended procedure for recording l_{Base} that is illustrated in Fig. 6 (b): The key receiver samples five times (with 2 ms wait time in between) instead of only once and uses the maximum of the five values as l_{Base}. With a high probability, this gives us a base value which lies near l_{Max} and prevents oscillations to cause unexpected 'light on' events.

4) Security of the Light Channel: The primary reason for the security of the light signal sent by the SNL lies in the fact that the light signal does not penetrate walls, doors and similar obstacles. Unlike for key data sent within radio messages which can be received by any adversary node within the (difficult to control) transmission range of the sender, the propagation of the light signal is limited to the physical area the signal is sent in. While eavesdropping is usually easy, observing is not.

To be able to record the key information, the adversary needs to observe the light signal either with the help of a light sensor placed within the area of influence of the SNL or with a video camera that records images of the destination area. To do this, he needs access to the area where the key assignment is performed. This can be ruled out in many scenarios. In others, it might be possible to see if an adversary has placed a device for recording the light signal in the critical area.

An interesting side note on recording the signal with a camera is that special equipment would be required to record a light signal sequence generated by the SNL: With a minimum clock period of 10 ms as used in our experiments, the state of the light signal can change up to 200 times per second whereas standard video equipment only records between 24 and 30 images per second.

In application scenarios where the user of *Enlighten Me!* needs to deal with adversaries that might have access to the areas where the key assignment is performed, we need to make sure that they cannot record the light signal transmitted by the SNL. One important factor here is that the user is able to regulate the size of the reception area by himself. By reducing his distance to the destination node he can reduce the size of the area affected by the light signal of the SNL. We will discuss this in more detail in the evaluation.

In particularly hostile scenarios, we use a black plastic cup with a hole in the bottom to cover the receiver node and prevent any observation of the light signal from the outside.

B. *Enlighten Me! PDA*

Unlike the SNL which requires a dedicated hardware device for the transmission of key information, the motivation of the *Enlighten Me!* PDA approach is to use standard PDA hardware for this task. We place the key receiver node on the display of the PDA with the light sensor oriented towards the display and transmit the key information by varying the light levels shown on the display.

The *Enlighten Me!* PDA displays a rectangle switching its color between black and white representing the ‘light off’ and ‘light on’ states of the Manchester encoding. The underlying assumption is that the difference in the luminance level of the display showing these two colors is large enough to reliably differentiate the two states.

Based on the way it is used, the *Enlighten Me!* PDA approach is mainly applicable during the deployment of nodes and not for dynamic key assignments as it will usually not be possible to place a node on the display of the PDA after it has been deployed in the environment.

1) Hardware: We have implemented the *Enlighten Me!* PDA approach for Linux PDAs from Sharp (Sharp Zaurus SL-3200). Fig. 7 shows the setup in operation. The key receiver node is placed upside down on a predefined position on the display of the PDA so that the light sensor of the node faces the display.

In our current prototype implementation, a second TelosB sensor node attached to the PDA over USB acts as a communication bridge to the sensor network. We expect that this will not be required in the future when mobile devices are able to directly communicate with wireless sensor nodes, for example using a technology like 802.15.4.

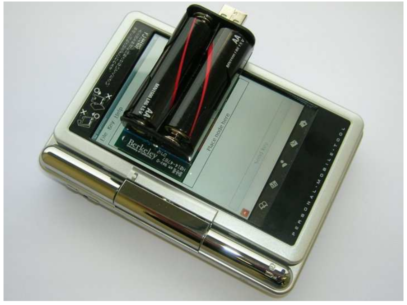


Fig. 7. Enlighten Me! PDA

2) Implementation: The software running on the PDA has been developed in C++ using the Qt toolkit. In addition to generating the light signal, the graphical user interface also allows the user to set the key which is then transmitted as part of the protocol. Before the protocol is started, a mirrored image of a sensor node helps the user in correctly placing the key receiver node on the PDA.

Despite being a much more powerful device, we cannot necessarily expect the *Enlighten Me!* implementation to work better on the PDA than on the Sensor Node Lamp. An important limitation of the PDA implementation is the switching speed of the LCD display used for the key transmission. Moreover, the low-level, monolithic TinyOS implementation on the SNL allows for a more precise control of the timings of activating and deactivating the LED as part of the encoding. Consequently, as is confirmed later in the evaluation, we do not achieve the same level of precision in switching between the ‘light on’ and the ‘light off’ state as on the SNL.

3) Security of the Light Channel: For the security of the light channel, it is again critical to guarantee that no adversary is able to observe, record and interpret the light signal transmitted between the key sender and the key receiver. However, for the *Enlighten Me!* PDA solution, this requirement is easier to fulfill than for the Sensor Node Lamp approach: The light signal used to transmit the key information is only shown in the area of the display where the light sensor of the key receiver node is placed. This area is completely covered by the sensor node. Additionally, we show bright colors on the rest of the display not covered by the sensor node during key transmission to outshine any light level differences from under the sensor node. This approach makes an external observation of the signal practically impossible.

If the user places the key receiver node incorrectly leaving the send area of the light signal exposed, then the protocol will already fail and abort during the handshake phase – before any key information is transmitted over the light channel.

C. *Key Receiver Implementation*

We have implemented the key receiver part of *Enlighten Me!* based on TinyOS 2.1 for TelosB sensor nodes. Due to the modular nature of TinyOS, it is straightforward to

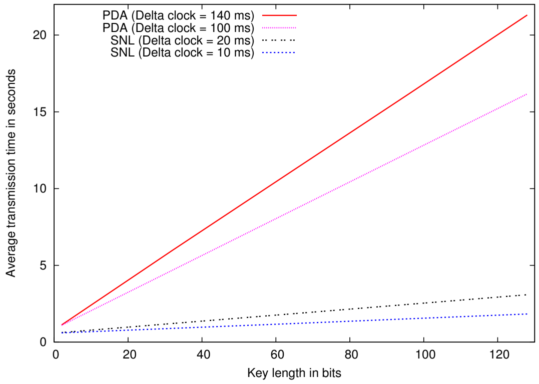


Fig. 8. Average key transmission times

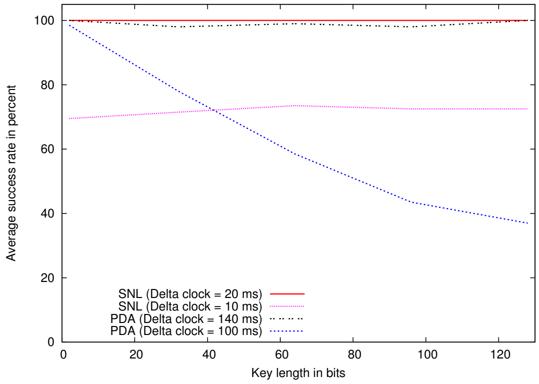


Fig. 9. Average success rates

combine the *Enlighten Me!* key receiver implementation with the application code that realizes the actual node functionality.

The basic key receiving mechanism is independent of whether the Sensor Node Lamp or the *Enlighten Me!* PDA is used for sending the key information. Consequently, we use the same receiver implementation and nodes are able to receive key data from both types of sender devices. The only difference lies in the length of the clock period Δ_{clock} used for the Manchester encoding on the SNL and the PDA. However, the sender node advertises its type as part of the handshake trigger message and the receiver is able to adjust its decoding behavior accordingly.

VI. Evaluation

In this section we show and discuss our results from the evaluation of *Enlighten Me!* with both the Sensor Node Lamp and the *Enlighten Me!* PDA.

A. *Key Assignment Performance and Reliability*

To evaluate the speed of assigning keys with *Enlighten Me!*, we measured the execution time of the key assignment process using different key lengths. We worked with key sizes of up to 128 bits which a current report [15] describes as sufficient for long-term protection of data when symmetric encryption is used.

The biggest contributor to the overall time is the time required for sending the key as a light signal which depends both on the key length and the clock period Δ_{clock} of the signal transmission. To minimize this time, we experimented with different values of Δ_{clock} trying to select the smallest value possible. The experiments confirmed our expectations that we can operate with much smaller values of Δ_{clock} for the Sensor Node Lamp approach than with the *Enlighten Me!* PDA. Fig. 8 summarizes the results of these measurements showing the average time values determined across 200 successful experiments per setting.

As expected, the key transmission time grows almost perfectly linearly with the key length starting from a small, size-independent base overhead caused by the basic *Enlighten Me!* protocol. Transmitting a 128 bit key with the SNL only takes between 1.8 seconds ($\Delta_{clock} = 10ms$) and 3.1 seconds ($\Delta_{clock} = 20ms$). Transmitting the same key with the *Enlighten Me!* PDA takes considerably longer: between 16.2 and 21.3 seconds ($\Delta_{clock} = 140ms$). This shows the main advantage of the directly controllable LED of the SNL over the LCD display of the PDA.

The second important factor besides the performance is the reliability of key assignment with *Enlighten Me!*. A key assignment can fail mainly due to two reasons. Firstly, noise in the light signal or light level variations caused by external (natural or artificial) light sources can interfere with the light signal decoding at the receiver. Secondly, imprecisions of the signal timings on both the sender and the receiver side can also impede the successful signal transmission.

To evaluate the reliability of the key assignment with *Enlighten Me!*, we measured the success rate for different values of Δ_{clock}. For the Sensor Node Lamp approach, we placed sender and receiver at a distance of 1 meter. Fig. 9 shows the results of these measurements.

The Sensor Node Lamp with $\Delta_{clock} = 20ms$ and the *Enlighten Me!* PDA with $\Delta_{clock} = 140ms$ represent the cases of a reliable assignment of keys with both success rates in the range of 100% independent of the key length. When further decreasing Δ_{clock}, the behavior of the two approaches differs.

On the one hand, for the *Enlighten Me!* PDA with $\Delta_{clock} = 100ms$, the success rate decreases when the key length increases. This is easily explained with the growing window of opportunity for timing errors in the signal or the signal decoding. On the other hand, for the SNL with $\Delta_{clock} = 10ms$, the success rate remains relatively independent of the key length and lies between 69.5% and 73.5%. We found that almost all protocol errors occurred right at the beginning of the key transmission making the error rate independent of the key length. Either on the key sender or the key receiver timing imprecisions occur right at the beginning of the key transmission in some cases, for example caused by effects of the previous protocol steps.

We were able to successfully perform key assignments with the *Enlighten Me!* PDA down to $\Delta_{clock} = 80ms$. However, in this case the success rate was only 11% for a key length of 48 bits and we did not observe any successful key assignments

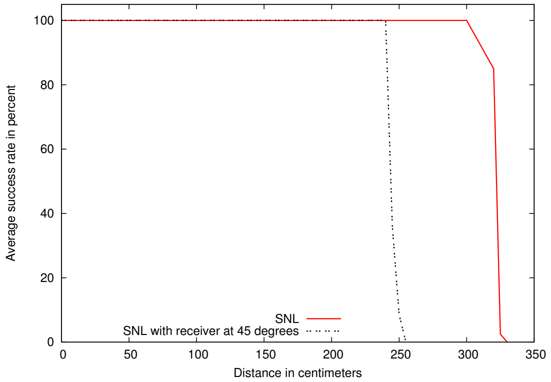


Fig. 10. Success rate over distance

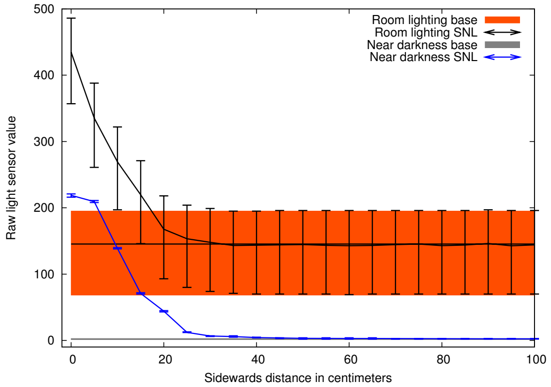


Fig. 11. Light level over sidewards distance

for key lengths of 64 bits or beyond. For the SNL, we were not able to successfully use values of Δ_{clock} smaller than 10ms.

B. Key Assignment Distance

While the *Enlighten Me!* PDA solution requires the user to place the key receiver node directly on the PDA, it is possible to assign keys with the Sensor Node Lamp from a certain distance. We evaluated the maximum distances for the key assignment in a controlled experiment where we placed the SNL and the key receiver node at different distances facing each other and measured the success rate across 200 experiments performed with the room lighting turned on. In a second experiment, we placed the receiver node with an angle of 45 degrees to the incoming light. This represents the case when the user does not stand directly in front of the key receiver node, for example when it is attached up at a wall.

Fig. 10 shows the results of these experiments. Up to a certain distance (3.00 meters in the standard case, 2.40 meters at 45 degrees), the success rate is not affected at all. Beyond that "threshold distance", the success rate falls very steeply. This can be explained as follows: With a growing distance between sender and receiver, the impact of the light of the LED on the receiver decreases. Above the threshold distance, the light level difference between the 'light on' and the 'light off' states becomes too small to compensate the continuous oscillations of the light levels caused by the artificial light in the room – the range of sensor values recorded in the 'light on' state begins to overlap with the range of values in the 'light off' state. Therefore, a reliable distinction of the states is not possible anymore and the detection of the light signal fails in most cases.

We were able to achieve higher maximum key assignment distances in experiments with the room lighting turned off (with exact values heavily depending on the current ambient light level). The results shown in Fig. 10 represent the performance in the worst case scenario.

Overall, with a possible key assignment distance between 2 and 3 meters, it should be possible to reach a large fraction of sensor nodes already deployed in a building or a similar environment. Assigning keys from even larger distances might not be desirable anyway as it reduces the control over which nodes are actually able to receive the light signal.

C. Security of the Light Channel

Transmitting keys over the light channel effectively prevents the overhearing of key information in neighboring rooms or areas. However, in some scenarios it might also be critical to limit the propagation of the signal within an area. To evaluate the properties of the Sensor Node Lamp in this regard, we measured the impact on light sensors not lying in the center of its light beam. For the experiment, we placed the SNL and the receiver node 2 meters apart and then gradually moved the SNL sidewards to move the light beam away from the light sensor. In each position, we recorded 1000 sensor values and determined the maximum, minimum and average values for comparison with the base values recorded without the SNL.

Fig. 11 shows the results of these measurements in a room for two situations both recorded at night: In the first experiment, the room lighting was turned on. In the second experiment, the room lay in nearly complete darkness. Clearly, when moving the SNL sidewards, the light level quickly approaches the range recorded without the SNL in both situations. For the room lighting scenario, already at a sidewards distance of 30 centimeters no significant difference between the recordings with and without the SNL can be detected. In the dark room, the average light level recorded with the SNL remains slightly higher than the average base level even at a sidewards distance of 100 centimeters. However, the range of values recorded with and without the SNL heavily overlaps starting at a sidewards distance of 50 centimeters so that a reliable signal detection will not be possible anymore.

Note that in the dark room, the red light of the SNL – albeit too weak for the TelosB light sensor – was visible over a much larger area, partly due to reflections. So with specialized equipment, it might still be possible to record the signal farther away from the target area. In the room lighting situation, the high ambient light level and the noise present in the ambient light actually help in limiting such an unwanted propagation of the SNL light signal.

We also performed experiments in daylight scenarios where

TABLE I
MEMORY OVERHEAD ANALYSIS (ALL VALUES IN BYTES)

		Blink	**Osc’cope**	**Osc’cope2**	**MViz**
Original	ROM	2650	13426	16592	28134
	RAM	55	394	438	1912
Key Receiver	ROM	18820	20912	19376	30558
	RAM	556	666	632	2100

it showed that the sidewards decrease of the SNL influence is similar to the room lighting scenario. The main influence here is the comparatively high ambient light level which quickly outshines the SNL. However, systematic measurements like shown in Fig. 11 are difficult to produce as the base light level varies significantly over time.

As a result, our measurements have shown that the SNL allows to effectively limit the reception area of the key signal. In daylight and room lighting scenarios, the high ambient light level and light level noise help in limiting the propagation of the light signal. If, however, the key assignment has to be performed in a particularly hostile environment, we still recommend to use an auxiliary device like the black cup described before to prevent the light signal from reaching other nodes than the destination node.

D. Memory Overhead

The memory consumption of the key receiver functionality implemented in TinyOS is a relevant factor as both program memory and main memory are very limited resources on typical sensor node platforms (e.g., 48 kB of program memory and 8 kB of main memory on the TelosB nodes).

The actual memory overhead of our solution strongly depends on the application the key receiver is integrated with as it can share code with other application parts running on the sensor node (e.g., the radio communication or the modules required for accessing the light sensor). To evaluate this, we have integrated our key receiver implementation with three representative applications from the TinyOS source tree: **Blink**, **Oscilloscope** (one configuration using the temperature sensor (Osc’cope); one using the light sensor (Osc’cope2)) and **MViz** (using the light sensor). Table I summarizes the resulting size values for both program memory (ROM) and main memory (RAM).

Overall, the overhead is reasonably small both in program memory and main memory (e.g., only 2424 bytes of program memory and 188 bytes of main memory for the MViz application) and should allow the integration with a variety of applications. As expected, the more of the required modules the application already contains, the smaller is the overhead of integrating the key receiver mechanism – ranging from the very simple Blink application that only includes minimal functionality by itself (i.e., no sensor access modules and no communication components) to the complex MViz application which allows reusing large parts of the code.

VII. CONCLUSIONS

In this paper we have presented *Enlighten Me!*, a novel approach for the secure assignment of keys to wireless sensor nodes that is easy and convenient to use. We have introduced the basic concept of *Enlighten Me!*, discussed the types of attacks it is able to deal with, the details of the protocol and two types of key sender devices. In the evaluation section we have demonstrated that *Enlighten Me!* provides an efficient solution to key assignment in wireless sensor networks.

As part of future work, we are planning to experiment with a wider variety of sensor node platforms. While we expect the basic mechanism to work on a variety of platforms, it will be interesting to see how the achievable key assignment performance differs among node and sensor types. We also want to investigate coding schemes that are more efficient than Manchester coding but that might be more susceptible to timing errors. For the *Enlighten Me!* PDA solution, it might also be possible to improve the efficiency of the data transmission by working with multiple light levels (by displaying different colors) instead of doing the binary ‘light on’ / ‘light off’ encoding. Finally, we are also very interested in the integration of our key sender functionality into other commercial off-the-shelf devices. For example, we imagine using the small LEDs which are integrated as a photoflash replacement in some mobile phones today for sending key data.

REFERENCES

[1] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, “Spins: security protocols for sensor networks,” *Wireless Networks*, vol. 8, pp. 521–534, 2002.

[2] H. Chan, A. Perrig, and D. Song, “Random key predistribution schemes for sensor networks,” in *Proc. of the 2003 IEEE Symp. on Security and Privacy*, 2003.

[3] A. Wacker, M. Knoll, T. Heiber, and K. Rothermel, “A new approach for establishing pairwise keys for securing wireless sensor networks,” in *Proc. of the 3rd Int. Conf. on Embedded Networked Sensor Systems*, 2005.

[4] C. Karlof and D. Wagner, “Secure routing in wireless sensor networks: Attacks and counter- measures,” in *Proc. of the 1st Int. W’shop on Sensor Network Protocols and Applications*, 2003.

[5] S. Capkun and J. Hubaux, “Secure positioning in wireless networks,” *IEEE Journal on Selected Areas in Communications*, vol. 24, pp. 221–232, 2006.

[6] J. Hwang and Y. Kim, “Revisiting random key pre-distribution schemes for wireless sensor networks,” in *Proc. of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks*, 2004.

[7] H. Chan and A. Perrig, “Pike: peer intermediaries for key establishment in sensor networks,” in *Proc. of the 24th Annual Joint Conf. of the IEEE Computer and Communications Societies*, 2005.

[8] D. Balfanz, D. Smetters, P. Stewart, and H. C. Wong, “Talking to strangers: Authentication in ad-hoc wireless networks,” in *Proc. of the Network and Distributed System Security Symposium*, 2002.

[9] F. Stajano and R. J. Anderson, “The resurrecting duckling: Security issues for ad-hoc wireless networks,” in *Proc. of the 7th Int. Workshop on Security Protocols*, 2000.

[10] C. Kuo, M. Luk, R. Negi, and A. Perrig, “Message-in-a-bottle: user-friendly and secure key deployment for sensor nodes,” in *Proc. of the 5th Int. Conf. on Embedded Networked Sensor Systems*, 2007.

[11] C. Castelluccia and P. Mutaf, “Shake them up!: A movement-based pairing protocol for cpu-con- strained devices,” in *Proc. of the 3rd Int. Conf. on Mobile Systems, Applications, and Services*, 2005.

[12] W. Xu, W. Trappe, Y. Zhang, and T. Wood, “The feasibility of launching and detecting jamming attacks in wireless networks,” in *Proc. of the 6th ACM Int. Symp. on Mobile ad hoc networking and computing*, 2005.

[13] A. D. Wood and J. A. Stankovic, “Denial of service in sensor networks,” *Computer*, vol. 35, pp. 54–62, 2002.

[14] W. Stallings, *Data and Computer Communications*. Prentice Hall, 2004.

[15] M. Näslund (Edt.), “ECRYPT yearly report on algorithms and keysizes (2007-2008),” ECRYPT European Network of Excellence in Cryptology, Tech. Rep., 2008.