SUDOKU: Secure and Usable Deployment of Keys on Wireless Sensors

Matthias Wilhelm*, Ivan Martinovic*, Ersin Uzun‡, and Jens B. Schmitt*
*disco | Distributed Computer Systems Lab, TU Kaiserslautern, Germany
{wilhelm, martinovic, jschmitt}@cs.uni-kl.de
‡Computer Science Department, University of California, Irvine, United States
euzun@ics.uci.edu

***Abstract*—Initial deployment of secrets plays a crucial role in any security design, but especially in hardware constrained wireless sensor networks. Many key management schemes assume either manually pre-installed shared secrets or keys authenticated with the aid of out-of-band channels. While manually installing secret keys affects the practicability of the key deployment, out-of-band channels require additional interfaces of already hardware-limited wireless sensor nodes. In this work, we present a key deployment protocol that uses pair-wise ephemeral keys generated from physical layer information which subsequently enables an authenticated exchange of public keys. Hence, this work presents an elegant solution to the key deployment problem without requiring more capabilities than already available on common low-cost devices. To justify the feasibility of this solution, we implement and experimentally evaluate the proposed key deployment protocol using commodity wireless sensor motes.**

I. Introduction

One of the crucial problems in wireless sensor networks is how to establish initial security associations between wireless sensor motes and a base station. Specifically, how to create an authenticated channel that can be used to exchange cryptographic material in a secure, scalable, and user-friendly manner. This question is important regardless of available cryptographic primitives, as even an asymmetric key must be protected from active adversaries abusing the broadcast nature of wireless communication.

There are various conventional approaches to this problem. For example, Kuo *et al*. propose the use of a Faraday cage and a wireless jammer to protect secret keys exchanged in the clear [4]. However, this approach requires a well designed Faraday cage, a careful handling of sensors and an additional verification mechanism to ensure that the Faraday cage is not leaking protected transmissions. Further schemes include secure device pairing protocols that use human perceptible out-of-band channels such as auditory, e.g., [2], [10], visual, e.g., [7], [8] and tactile [6], [9] channels to authenticate secrets by user interaction. Yet, implementing such out-of-band channels assumes additional hardware interfaces, e.g., microphones, cameras, displays, or keypads that are not commonly available on hardware- and battery-limited wireless sensor devices. Additionally, such schemes place the security burden on the user, who may not understand the importance of the security measures.

Recently, there have been research contributions that follow an alternative path towards secret generation. They take advantage of multi-path signal propagation as a source of randomness from which shared secrets can be derived. In particular, the key generation schemes described by Mathur *et al*. [5], Jana *et al*. [3], and Wilhelm *et al*. [12] analyze how the fading behavior of wireless channels can be used to generate secret material. Such protocols only require an exchange of sampling messages to estimate the wireless channel state between legitimate transmitters. Interestingly, an adversary who eavesdrops on these sampling messages from another physical position remains ignorant of the generated secrets. The reason is that its channel estimates de-correlate rapidly with increasing distance to the legitimate transmitters (for more details on the secrecy analysis of such schemes see, e.g., [3]). Hence, if an application setting can ensure that an adversary cannot be located in spatial proximity to one of the legitimate transmitters, these schemes can be used as efficient and secure key derivation protocols.

While existing research on these protocols is focused on identifying and analyzing primitives for such secret generation, the evaluation of their practicability in real-world settings remains an open question. The goal of this work is therefore to systematically analyze the applicability of such schemes to a practical problem of initial secret deployment in WSNs. In this paper, we take advantage of the protocol introduced in [12] to design SUDOKU, a secure and usable key deployment scheme that requires minimal user interaction and has minimal requirements on hardware interfaces of wireless devices. Additionally, we augment the key generation protocol to support simultaneous key exchanges using broadcast transmissions, increasing its performance with multiple sensors.

II. SUDOKU Overview

We introduce the concept and the design goals of the protocol in this section. The role of the participants is introduced as well as the sequence of protocol phases, and the involvement of the user is described.

The work in this paper was partially funded by the Carl-Zeiss Foundation and the Landesforschungsschwerpunkt Ambient Systems (AmSys).

A. Protocol Goals

We consider a very common application of WSNs in residential monitoring and living-assistance scenarios. A user wishes to deploy wireless sensors as part of an assisted-living application where environmental conditions are collected and sent to a central base station (BS), which makes more sophisticated decisions. One of the main problems in such scenarios is how to initially establish secure associations between wireless sensors and the BS to protect the application from various impersonation and injection attacks inherently possible due the broadcast nature of wireless communication. Examples of such attacks are *(i)* the *evil twin* attack where a sensor mote is impersonated, *(ii)* eavesdropping and violating the confidentiality of the communication, and *(iii)* the injection of *rogue* (attacker-controlled) sensor devices into the network. The goals is to mitigate these attacks with the following measures:

- Device identification: the user can physically identify which devices are being paired.
- Authenticated messages: it is not possible for adversaries to inject messages into the network.
- Confidentiality: a strong secret key is derived to achieve long-term security.

The protocol is designed to meet these goals in an user-friendly way such that usage errors are mitigated.

B. Protocol Participants

Three different parties participate in the SUDOKU protocol: the base station that manages the execution of the protocol, a group of sensor motes that wants to establish secure associations with the BS, and the user who deploys the network.

a) Base Station: The base station is the central element in the execution of the protocol, it coordinates the participating sensor motes and protects them against message injection attacks by constantly monitoring the wireless channel. We assume sufficient performance and a user interface that enables comfortable interaction (such as a high resolution display and multiple buttons), comparable to desktop-class computers.

b) Sensor Motes: For the sensor motes cost-efficiency is the most important aspect in the hardware design, which results in minimal platform capabilities. Therefore, we only require the availability of three common interfaces for SUDOKU: a single LED light, an on/off switch and a wireless transceiver to communicate and to measure the received signal strength (RSS). The switch can be simulated, e.g., by inserting or removing the battery of the device. We use the status LED in three different states for user feedback: OFF shows that the device is either un-powered, searching for a BS or in an error state, BLINKING means that the device is associated with the base station but the key is not ready, and ON indicates successful key deployment. In contrast to other secure key deployment protocols, the LEDs are not used for generation or distribution of secret material, but only as a status-reporting interface.

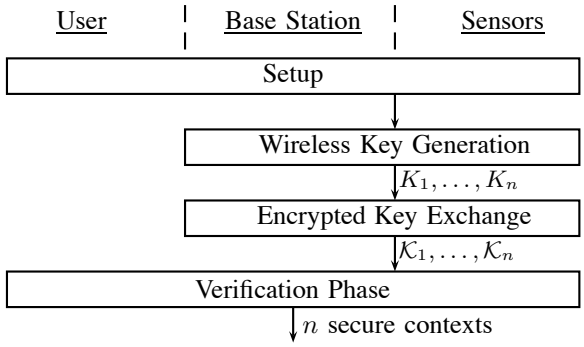

Figure 1. Key deployment concept of SUDOKU. The protocol proceeds in four phases: *(i)* the user sets up the devices, *(ii)* ephemeral keys K_i are generated from channel measurements, *(iii)* encrypted key exchanges are performed to establish long-term secrets $\mathcal{K}_i$ with the help of the ephemeral keys, and *(iv)* the user verifies that the correct devices are paired by checking the devices' LEDs.

c) User: The user involvement and interaction in the protocol must be designed to be simple enough for the user to understand and complete with minimal errors. In SUDOKU, users are only required to carry out very simple tasks such as counting the sensor motes or verifying that their status LEDs are on. The user can abort the protocol if an attack is detected and can initiate countermeasures such as a delayed key deployment or even law enforcement if the attack persists.

C. Protocol Overview

The protocol execution proceeds in four phases, the conceptual overview is depicted in Fig. 1. First, the user sets up the devices, starting with the base station that begins to monitor the wireless channel and periodically sends out beacon frames with control information. New sensor motes associate with the base station, which detects the number of sensors that are active in the vicinity and displays this number to the user to prevent rogue sensor motes. When all sensors are ready, the user initiates the second phase, wireless key generation. Ephemeral shared secrets are derived from channel measurements between the base station and the sensors. In the third phase, key deployment, these generated keys are used to protect cryptographic key exchanges with the sensors against active attacks. The derived long-term secrets are then verified by the devices in the last phase, and the sensors indicate successful key agreement with their LEDs. The user checks if all sensors were successful, at this point the base station shares pair-wise strong security contexts with all sensors. After this, the nodes can be placed in their final locations; the derived keys can either be used for secure single-hop communication or to support end-to-end encryption in multi-hop topologies.

D. User Perspective

In order to achieve a secure key deployment by non-expert users, error mitigation and error resistance are major goals of SUDOKU. With our proposed protocol, the steps a user must perform can be summarized as follows:

1) The user powers on the base station first, which starts to assist the user in the progression of the protocol.

$A \rightarrow B: \langle M \rangle$	A sends message M to B
$A \Rightarrow *: \langle M \rangle$	A broadcasts message M to all participants
BS	Base station, manages the protocol execution
S_i	Sensor mote $i \in \{1, \ldots, n\}$
$\mathcal{S}$	Set of all legitimate sensors, $\mathcal{S} = \{S_1, \ldots, S_n\}$
n	Number of sensors participating in the protocol
c_i	Wireless channel available for probing, $i \in \{1, \ldots, m\}$
$\mathcal{C}$	Set of available channels, $\mathcal{C} = \{c_1, \ldots, c_m\}$
m	Number of available channels
k	Number of sampling rounds in the KeyGen Phase
$slots$	Number of time slots/associated sensor motes
Pub_{DH}	Public parameters for the encrypted key exchange
$Ping_{c,j}$	The jth probe message on channel c from BS
$Pong^{i}_{c,j}$	The jth probe message on channel c from S_i
$RSSMAP_c^{S_i}$	Received signal strength means of S_i on channel c
$RSSMAP_c^{B_i}$	Received signal strength at BS from S_i on channel c
T_i, P_i	Tolerance and repair values for KeyGen phase
$h()$	Strong cryptographic one-way hash function
K_i	Pair-wise ephemeral key between BS and S_i
a_i, b_i	Large random numbers used in the DH key exchange
A_i, B_i	Public keys used in the DH key exchange
$E_K(), D_K()$	Symmetric en- and decryption using key K
$\mathcal{K}_i$	Pair-wise long-term secret between BS and S_i

Table I
PROTOCOL NOTATION

2) Then, each sensor device at a time, the user activates the sensor motes. A sensor tries to associate with the BS, and notifies the user by a blinking LED that it is successfully associated. The user can then place the sensor in the vicinity of the BS and go on with the other sensors until all are ready.
3) Before starting the key deployment, the user ensures that the number of sensors shown on the display of the BS matches with the number the user wishes to associate. He then triggers the key generation phase with a button.
4) After approx. 20–30 seconds, the BS notifies the user that the key deployment has finished and asks him to check that all sensors have their LEDs constantly ON.
5) If all sensors indicate successful completion, the user presses OK on the base station to finish the key deployment.

Each of the instructions require the user to handle the hardware, check for LEDs or information displayed on the base station; no keying material must be entered by the user manually. And by using the BS to guide him through the process, many possible further error sources can be avoided.

III. THE SUDOKU PROTOCOL

This section presents the key deployment protocol in detail, discussing every step of the protocol. To assist the reader, the notation used in the protocol description is summarized in Table I.

A. Initial Setup Phase

This initial phase is used to distribute parameters for the protocol execution. We require a deterministic medium access scheme in SUDOKU. Any contention-free MAC can be used; we implemented a simple TDMA-based MAC that divides the medium into 2 ms time slots allotted according to the number of associated sensors. The BS starts sending out beacons periodically that mark the beginning of a new turn. New sensors contend for the free time slot at the end of a turn, while associated sensors transmit in their respective time slot to show presence to the base station and to claim the medium. This is the only period with contention in the protocol, i.e., the period where collisions are possible without indicating an attack.

	User:	Power on BS
	$BS \Rightarrow *$:	$\langle slots, k, \mathcal{C}, Pub_{DH} \rangle$
for $S_i \in \mathcal{S}$:	User:	Turn on S_i
	S_i:	Find a free time slot using $slots$
	$S_i \rightarrow BS$:	$\langle slots, k, \mathcal{C}, Pub_{DH} \rangle$
	S_i:	Start BLINKING
	User:	Put sensor S_i near BS
	User:	Confirm the number of sensors n on BS Initiate the key generation phase on BS

Figure 2. Setup Phase.

As shown in Fig. 2, the user turns on the base station first, which starts to monitor the wireless channel to protect against message injection. Also, it starts sending out beacons with the protocol parameters periodically; control information for the key generation phase, i.e., the number of samples k to collect on each channel and the set of channels $\mathcal{C}$ to measure on, the number of currently associated sensors in $slots$ to help new sensor motes aligning into the TDMA scheme and to give them feedback whether they are successfully associated with the BS, and finally the public protocol parameters Pub_{DH}, in the case of Diffie-Hellman g and p, that are later used in the key deployment phase (Section III-C).

The user then proceeds to sequentially turn on the sensor motes that he wishes to securely deploy keys to. A joining sensor starts scanning for beacon frames and associates with the BS by re-broadcasting the received parameters to verify with the BS that no forged parameters were injected. The BS can detect the presence of a joining sensor, and adjusts the number of associated sensors in $slots$ to notify the sensor. In turn, the sensor informs the user that it is associated by BLINKING. This ensures the required device identification. The user can then place the sensor device in the vicinity of the BS and proceed with the next mote.

The base station keeps track of the number n of already detected sensors and displays it to the user to ensure that no rogue sensors are present that want to enter the network. When everything is set up and ready, the user starts the key generation phase by pressing the respective button on the base station. At this point, he confirms that the number of devices talking to the BS matches the number of sensors present to prevent the pairing of malicious sensors, and to enable the BS to detect the completion of the protocol.

B. Key Generation Phase

The goal of this phase is to use the randomness of the wireless channel to extract an authenticated secret key without user interaction. This is achieved by the creation of a received

signal strength (RSS) map containing channel measurements for every sensor in $\mathcal{S}$ and every wireless channel in $\mathcal{C}$. We denote this map $RSSMAP_c^{S_i}$, it belongs to S_i and contains the RSS means for the BS–sensor pair, and $RSSMAP_c^{B_i}$ for the corresponding measurements by BS. Using this measurement data, an ephemeral pair-wise and symmetric secret key K_i is generated, which is used to authenticate the key deployment in the next phase of the protocol.

1) Wireless Key Generation: The ephemeral key is generated using the protocol described in [12]. Due to the reciprocity of the wireless channel, two devices are able to extract shared secret information simply by exchanging probing messages over the wireless channel. These measurements may not be equal as the reciprocity is not perfect and measurement errors are introduced, so both devices must perform a secret reconciliation step to correct these deviations. In contrast to related work that requires device mobility, the use of multiple channels in our protocol enables the generation of secure keys even in static scenarios.

Our previous work shows that wireless key generation can successfully be applied in the context of resource-constrained devices in WSNs, enabling the generation of unpredictable keys and a high success ratio, by experiments in real-world scenarios. However, the previous protocol is designed for a pair-wise key generation between two devices only; this paper proposes improvements to the basic protocol that enable an elegant way to increase the number of protocol participants, thus decreasing the duration of the protocol, as the channel sampling is the major factor in the overall runtime. The key idea is that by using the broadcast nature of the wireless medium, we are able to run several key generations simultaneously; each device broadcasts its sampling messages to all other devices, who record the received signal strength. This scheme works because of the long coherence time of the wireless channel due to the static network infrastructure. In Section IV, we show that this new approach is feasible, even using currently available sensor mote hardware, to improve the overall performance of SUDOKU.

2) Execution of the Capturing Phase: In this phase, all devices are still operating according to the TDMA scheme. In each time slot, the sensors and the base station broadcast their probing message to all other devices. Due to the fixed channel switching pattern (the timing, sending pattern, and number of sensors is distributed by the BS), there is no contention for the medium and the BS can make sure that it can monitor the correct channel and that no sensor loses its synchronization, which is crucial to protect the sensor motes from attacks.

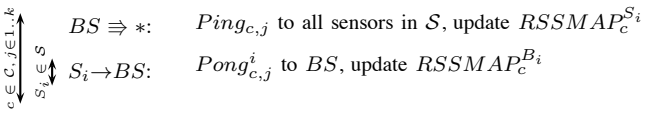

Figure 3. Randomness Capturing Phase.

As shown in Fig. 3, the BS starts by broadcasting $Ping_{c,j}$ frames on the first channel after the user has started the operation. For every received frame, each sensor S_i responds with a broadcast frame $Pong_{c,j}^i$ in its time slot. After k rounds, all participants know from the deployed parameters the time for channel switching. The sampling is repeated until all m channels have been sampled. The overall number of transmissions required for this phase is in the order of $O(kmn)$. After all samples have been taken, the protocol proceeds by correcting the measurements, generating and checking the ephemeral key K_i as described in [12] with each of the sensors in $\mathcal{S}$.

BS:	Generate T_i, P_i, and calculate K_i
$BS \rightarrow S_i$:	$\langle T_i, P_i \rangle$
S_i:	Correct the $RSSMAP^{S_i}$ and calculate K_i
$S_i \rightarrow BS$:	$\langle h(K_i) \rangle$
BS:	Check K_i is equal on both sides

Figure 4. Key Reconciliation Phase for Sensor S_i.

The base station chooses a vector of tolerance values $T_i = (t_1, \ldots, t_m)$ with one entry for each channel. These values describe the amount of errors than can be present in the measurements until the protocol results in different keys. However, this presents a tradeoff as larger tolerance values lower the uncertainty for attackers and therefore the security of the generated key. There is a direct connection between the precision of the measurements, the number of channels used for sampling and the length of the generated secret; all aspects influence the security of the protocol. This interrelation has been quantified in our previous work, which supports us here to choose suitable protocol parameters.

The public reconciliation string P_i sent to S_i helps sensor S_i to repair its signal measurements. The values (T_i, P_i) are generated and broadcasted for each of the sensors S_i, which can in turn correct their $RSSMAP^{S_i}$ and generate K_i. Then, the keys are checked for equality by exchanging the hash value of the secret. If K_i does not match, it is possible to repeat the repair mechanism; this step is omitted in this paper for the sake of clarity.

After this phase, the base station has successfully established ephemeral keys K_i generated from the wireless channel with all sensor motes S_i, which can then be used to authenticate key agreements on long-term secrets.

C. Key Deployment Phase

The keys K_i generated in the previous phase can directly be used as authenticated shared secrets. However, as current hardware platforms only support a small number of channels and have limited measurement precision, the resulting secrets currently have approximately 20–30 bit of information entropy as experiments in [12] show, which is comparable to password-based security, but not useful as long-term secrets. Therefore, we combine public key cryptography for confidentiality and the wireless key generation approach to enable an authenticated key exchange. The communication is authenticated as only devices at the correct positions can know the secret generated by the wireless channel.

1) Authenticated Key Exchange: There are several ways to establish stronger keys from low entropy shared secrets. In the following, we provide a concrete example of how a secret generated from the wireless channel can be utilized for an authenticated key exchange. The protocol in its original form is known as Password-based Encrypted Key Exchange (EKE), which is originally described in [1] and extended in [11]. The general idea is to use the short secret key to encrypt randomly chosen key material. This is an elegant way of avoiding off-line dictionary attacks on weak shared secrets, since an adversary cannot guess the random values. We use a Diffie-Hellman approach for key agreement in SUDOKU.

BS:	$a_i = random(), Nonce_i = random()$
	$A_i = g^{a_i} \bmod p$
$BS \rightarrow S_i$:	$\langle Nonce_i, E_{K_i}(A_i) \rangle$
S_i:	$b_i = random()$
	$B_i = g^{b_i} \bmod p$
	$\mathcal{K}_i = A_i^{b_i} \bmod p$
$S_i \rightarrow BS$:	$\langle E_{\mathcal{K}_i}(Nonce_i), E_{K_i}(B_i) \rangle$
BS:	$\mathcal{K}_i = B_i^{a_i} \bmod p$
	Check $Nonce_i = D_{\mathcal{K}_i}\left(E_{\mathcal{K}_i}(Nonce_i)\right)$
	Count S_i as successful on match
$BS \rightarrow S_i$:	$\langle E_{\mathcal{K}_i}(Nonce_i + 1) \rangle$
S_i:	Check $Nonce_i + 1 = D_{\mathcal{K}_i}\left(E_{\mathcal{K}_i}(Nonce_i + 1)\right)$
	Turn LED ON if successful

Figure 5. Encrypted Key Exchange with sensor S_i.

Concretely, in this protocol the ephemeral secret key K_i, generated in the previous protocol step, is used to mask the transmission of the cryptographic material A_i and B_i, which are then used for the generation of the strong encryption key $\mathcal{K}_i$ (in case of Diffie-Hellman, this key can be derived as $\mathcal{K}_i = g^{a_i b_i} \bmod p$, where modulus p and base exponent g are the parameters Pub_{DH} exchanged in the setup phase).

The public parameters A_i and B_i are exchanged as shown in Fig. 5. At the same time, the Diffie-Hellman key $\mathcal{K}_i$ is verified for its freshness. The BS chooses $Nonce_i$ randomly and sends it to S_i, which in turn shows that it is in possession of the generated ephemeral key K_i by extracting the public parameter B_i and using it to compute $\mathcal{K}_i$. The nonce is send back by the sensor, encrypted with $\mathcal{K}_i$ to verify that it has successfully generated this key. Additionally, BS can now detect the successful key deployment. BS sends $Nonce_i + 1$ back to S_i to achieve mutual authentication, and the sensor can report this success to the user by turning the LED ON. This phase is repeated for all sensors.

D. Verification Phase

The sensors turn their LEDs ON when a successful key exchange is detected, and in the last phase it is necessary to ensure that the keys were deployed to the correct devices. The number of LEDs in the state ON must match.

BS:	Check if the number of successful key agreements matches
	Show SUCCESS on display
User:	Verify that all sensor motes in $\mathcal{S}$ have LEDs ON
	Press OK if the number of sensors matches

Figure 6. Verification Phase.

This phase is a countermeasure against rogue, missing and impersonated sensors. The base station can also notice if $K_i \neq K_i'$ or if $\mathcal{K}_i \neq \mathcal{K}_i'$ caused by problems in the key generation process, enabling it to decide if previous phases must be repeated to reach agreement. If the numbers match, the user can be sure that the devices he expects share a secure key with the base station, and in case of error the BS can identify the possible causes (attacks, strong deviations in the channel measurements, usage errors), warn the user and suggest countermeasures.

IV. EXPERIMENTS

This section describes experiments that focus on the feasibility of broadcast key generation introduced in Section III-B. This is an extension to our key generation protocol analyzed in [12], the hardware setup for this work is the same, so our previous results for robustness and strength of keys between pairs of devices still apply in this work.

A. Channel Reciprocity with larger Time Gaps

One important aspect of the SUDOKU protocol is the feasibility to generate ephemeral secrets from the wireless channel using broadcast messages. Our previous work shows that the scheme works for pair-wise scenarios, but due to the larger time gaps between the transmission of sampling messages, the channel reciprocity may be diminished in the multi-node case. This section describes the feasibility evaluation in a realistic scenario for our proposed extensions.

1) Methodology: The experiment takes place on a university floor, an indoor location comparable to assisted-living application scenarios. The role of BS is taken by a MICAz sensor mote on a programming board, so that the operation can directly be controlled by a host computer attached via serial connection. The six sensors S_i are MICAz sensor motes as well, but operate on battery power. These motes are placed randomly in a distance of 1–2 meters away from the BS and that the devices are at least 16 cm apart to ensure independent wireless channels [5]. During the experiments, several factors accounted for short-term disturbances to the channel. The wireless medium is shared with several wireless access points using the IEEE 802.11g standard, and an operator was near the sensor motes during the sampling phase influencing the signal propagation properties, which is realistic as the normal user will also remain close during the protocol's execution, and wireless interference is typical in such applications.

The experiment was repeated 20 times, and in every experiments, $k = 32$ samples were exchanged on $m = 16$ IEEE 802.15.4 channels to generate the RSS maps. The sensor motes $S_1, \ldots, S_6$ are programmed to measure the signal strength and report back these values to BS where the RSS of these

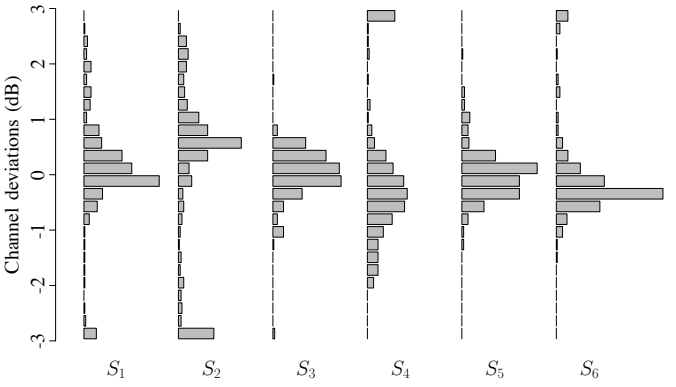

Figure 7. Deviations in the reciprocity of the wireless channel measurements for six sensors and a base station. The deviations are bounded and allow for a successful key generation.

reports are logged. At this point, BS has a pair of values $\left(RSSMAP_{c,k}^{S_i}, RSSMAP_{c,k}^{B_i}\right)$ for both views of the wireless channel state. In this experiment, we consider the deviations between these two measurements. Smaller deviations enables a more robust generation of keys.

2) Robustness w.r.t. Number of Sensors: The result of this experiment is shown in Fig. 7. Each stacked histogram shows the distribution of deviations of the channel measurements between BS and S_i. Due to the timing scheme, the last sensor S_6 has the largest delay between the broadcast of BS and the response. The experiments show that the wireless channel is stable when the sensors stay in position, the deviations are bounded and comparable to the experimental results in [12], even for larger time lags in the channel measurements. Thus, it is possible to apply the key generation protocol with minor modifications and generate keys with a success probability of over 95%.

However, the experiments also show that the protocol is more susceptible to short-term deviations, such that the number of exchanged samples should be increased to ensure successful key agreement. These short-term deviations are caused, e.g., by collisions with WLAN transmissions that increases the received signal strength, or changing multipaths due to movement of the operator, leading to occasional large deviations. Additional measures must be taken to ensure successful key generation, e.g., by filtering out samples that differ significantly from the sample mean.

To summarize, we can conclude that the key generation phase works with multiple devices at once. As a practical deployment guideline, a tolerance value of approximately $t = 2$ can be used that results in 20–30 secret bits, which is enough for the authentication in the key deployment phase, and ensures a high success probability.

V. Conclusion

The goal of this paper was to understand how recently proposed key generation protocols based on unpredictable signal propagation can be applied to the practical problem of initial key distribution in WSNs. For this purpose, we designed SUDOKU, a secure and practical key distribution scheme based on the protocol described in [12]. In contrast to previous work, the focus of this paper was to systematically analyze the applicability of this protocol in a scenario where non-expert users need to distribute initial secrets among wireless sensors in a secure and scalable manner. For this reason, this paper presented a detailed analysis of the steps required for such key deployments. The experiments show that the unicast key generation protocol can be further optimized, reducing the message complexity and runtime. As future work, we are working on a detailed security analysis and plan to conduct user studies to show the practicability of SUDOKU.

References

[1] Steven M. Bellovin and Michael Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *Proceedings of the 1992 IEEE Symposium on Security and Privacy*, pages 72–85, May 1992.

[2] Michael T. Goodrich, Michael Sirivianos, John Solis, Gene Tsudik, and Ersin Uzun. Loud and clear: Human-verifiable authentication based on audio. In *ICDCS '06: Proceedings of the 26th IEEE International Conference on Distributed Computing Systems*, page 10, Washington, DC, USA, July 2006. IEEE Computer Society.

[3] Suman Jana, Sriram Nandha Premnath, Mike Clark, Sneha K. Kasera, Neal Patwari, and Srikanth V. Krishnamurthy. On the effectiveness of secret key extraction from wireless signal strength in real environments. In *MobiCom '09: Proceedings of the 15th annual international conference on Mobile computing and networking*, pages 321–332, New York, NY, USA, September 2009. ACM.

[4] Cynthia Kuo, Mark Luk, Rohit Negi, and Adrian Perrig. Message-in-a-Bottle: user-friendly and secure key deployment for sensor nodes. In *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*, pages 233–246, New York, NY, USA, November 2007. ACM.

[5] Suhas Mathur, Wade Trappe, Narayan Mandayam, Chunxuan Ye, and Alex Reznik. Radio-telepathy: Extracting a Secret Key from an Unauthenticated Wireless Channel. In *MobiCom '08: Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, pages 128–139, New York, NY, USA, September 2008. ACM.

[6] Rene Mayrhofer and Hans Gellersen. Shake well before use: authentication based on accelerometer data. In *PERVASIVE'07: Proceedings of the 5th international conference on Pervasive computing*, pages 144–161, Berlin, Heidelberg, May 2007. Springer-Verlag.

[7] Jonathan M. McCune, Adrian Perrig, and Michael K. Reiter. Seeing-is-believing: using camera phones for human verifiable authentication. *International Journal of Security and Networks*, 4(1/2):43–56, 2009.

[8] Ramnath Prasad and Nitesh Saxena. Efficient device pairing using "human-comparable" synchronized audiovisual patterns. In *ACNS'08: Proceedings of the 6th international conference on Applied cryptography and network security*, pages 328–345, Berlin, Heidelberg, June 2008. Springer-Verlag.

[9] Claudio Soriente, Gene Tsudik, and Ersin Uzun. BEDA: Button-enabled device pairing. In *International Workshop on Security for Spontaneous Interaction, UbiComp 2007 Workshop Proceedings*, Berlin, Heidelberg, September 2007. Springer-Verlag.

[10] Claudio Soriente, Gene Tsudik, and Ersin Uzun. Hapadep: Human-assisted pure audio device pairing. In *ISC '08: Proceedings of the 11th international conference on Information Security*, pages 385–400, Berlin, Heidelberg, September 2008. Springer-Verlag.

[11] Michael Steiner, Gene Tsudik, and Michael Waidner. Refinement and extension of encrypted key exchange. *ACM SIGOPS Operating Systems Review*, 29:22–30, July 1995.

[12] Matthias Wilhelm, Ivan Martinovic, and Jens B. Schmitt. Secret Keys from Entangled Sensor Motes: Implementation and Analysis. In *Proceedings of the ACM Conference on Wireless Network Security (WiSec 2010)*, pages 139–144, Hoboken, NJ, USA, March 2010. ACM Press.