Lightweight Protection of Group Content Distribution

Pawel Szalachowski
Institute of Information Security
Department of Computer Science
ETH Zurich, Switzerland
psz@inf.ethz.ch

Adrian Perrig
Institute of Information Security
Department of Computer Science
ETH Zurich, Switzerland
adrian.perrig@inf.ethz.ch

ABSTRACT

Achieving security properties in distributed, hardware-limited, and unattended networks is a challenging task. This setting is challenging because an adversary can capture and physically compromise unattended nodes. In this setting, this paper presents one-way group communication protocols with strong security properties. In particular, how to send messages to a group of hardware-limited nodes with message secrecy and authenticity? We present several protocols and analyze them in terms of security, efficiency, and deployability. The resulting solutions are generic and can be useful in a variety of distributed systems.

Categories and Subject Descriptors

C.2.0 [**COMPUTER-COMMUNICATION NETWORKS**]: General—*Security and protection*

General Terms

Security

Keywords

Broadcast encryption; broadcast authentication; secure sensor networks; internet of things security.

1. INTRODUCTION

In the context of Internet of Things (IoT), where the Internet connects with hardware-constrained environments like Wireless Sensor Networks (WSNs), secure group communication remains an important and challenging topic. Nodes in IoT networks are often unattended and are intended to be low-cost, thus lacking dedicated physical protection with low computation and communication resources. As a consequence, an adversary can capture arbitrary nodes and with physical access extract their local states including secrets. This setting complicates security protocols.

In this paper, we investigate the problem of one-way secure communication in the presence of a powerful adversary in IoT environments. More specifically, a broadcast node establishes a group key with (some) network nodes, and broadcasts a message protected by this key. Such a solution requires two components:

- a group key establishment protocol (which allows to establish the same group key for legitimate receivers),
- a secure content distribution protocol (which uses the previous group key to protect actual content).

Unfortunately, in the literature these two aspects are addressed separately. As a result, several approaches for solving these problems individually exist, but their combinations are often insecure, inefficient, or almost impossible to deploy in a real-world scenario [15, 22, 23]. This paper aims to address that gap.

Our proposals are designed for regular nodes, without special hardware protections, that want to verify the authenticity and freshness of broadcast messages, while ensuring message secrecy if needed. These standard security requirements are challenging on resource-starved nodes and an adversary that can undetectably capture network nodes. In the worst case, assuming an uncompromised sender, can we still achieve secrecy and authenticity for all but one compromised receiver node? The main contributions of this work are:

- We propose an efficient framework for secure content distribution which includes:
  - a protocol that provides authentication and weak freshness,
  - a protocol that provides authentication, confidentiality, and weak freshness.
- We analyse presented solutions in terms of security, efficiency, and deployability.

The considered network consists of regular nodes and one central node. This is a generic broadcast model, and our solutions can be powerful security tools in IoT applications, such as monitoring, command & control, telemetry, or digital right management for content distribution.

The paper is structured as follows. Section 2 introduces a model of the protected system, cryptography background, adversary model, and security goals. Section 3 describes initial naive approaches and shows their drawbacks. Our improved protocols are presented in Section 4, followed by an analysis of the security, efficiency, and deployability of the approaches in Section 5. Section 6 discusses related work followed by our conclusions. The paper includes an appendix that introduces two additional protocols.

IoTPTS'15, April 14-17, 2015, Singapore.

http://dx.doi.org/10.1145/2732209.2732215.

2. BACKGROUND AND PRELIMINARIES

2.1 System model

This work considers *one-way* and *one-to-many* communication models, where one special node called the *Broadcast Center* (BC) broadcasts messages to distributed *regular nodes*. Regular nodes are hardware-limited, i.e., their computational, storage, and transmission capabilities are low. They can receive broadcast transmission, perform some selected cryptographic operations, store received transmission and a few secret values. Additionally, every regular node has a unique ID which is expressed as integer from 1 to n, where n is the number of regular nodes. The BC's capabilities are assumed to be slightly better than a regular node's capabilities, in that it can prepare and broadcast messages, and store at least a few values for every of the n nodes. Before the network's deployment, an administrator is able to pre-load some initial values into regular nodes and BC, for example using a protocol like MiB [11]. Regular nodes are not assumed to be trusted nor resilient to physical attacks, and the only trusted node in the network is the BC.

2.2 Cryptographic primitives

This section introduces cryptographic primitives used in our paper, unfamiliar readers are referred to [2, 9, 17, 18].

The first cryptographic tool that we introduce is a *Pseudo-Random Function* (PRF) $F_k(\cdot)$. Its first argument is a key k; a PRF without key (F_0) denotes a one-way function. F_k takes an arbitrary string ($\{0,1\}^*$) as a second argument and returns a pseudo-random string from $\{0,1\}^\lambda$. Other PRF properties include *non-invertibility* and *key secrecy*.

The next cryptographic primitive, which combines symmetric authentication with encryption, is called *Authenticated Encryption with Associated Data* (AEAD). It is defined as $\mathcal{A} = (Enc, Dec)$:

- $Enc_k(iv, a, m)$ takes a key k, a random and fresh value iv, additional data a, and a message m (may be *null*). The returned value is a ciphertext c (a is only authenticated while m is encrypted and authenticated).
- $Dec_k(iv, a, c)$ for key k, iv, a, and ciphertext c returns the decrypted message m or $\perp$ if the decryption fails.

AEAD can be used as a Message Authentication Code (MAC). Then, c denotes only an authentication tag.

It is assumed that AEAD is resistant to adaptive chosen-message attacks (when used in stand-alone MAC mode), and resistant to adaptive chosen-ciphertext attacks (when used for authenticated encryption).

The last mechanism that we introduce is called *Broadcast Encryption (BE)*, and its goal is to establish a group key in such a way that only members of a *privileged* set can obtain that key. Let U denote the set of all nodes and S be a privileged set, then $R = U \setminus S$ is an *unprivileged* set whose members cannot obtain a group key. BE in a symmetric setting is a triple $\mathcal{B} = (Gen, Enc, Dec)$ where:

- $Gen(n)$ takes a number of users $n = |U|$ and returns their secret keys $k_1, ..., k_n$.
- $Enc(S)$ for a privileged set $S \subseteq \{1, ..., n\}$ generates a shared *session key* SK and header hdr (party which executes Enc has to possess all keys from S).
- $Dec_{k_{id}}(S, hdr)$ takes a user's secret key k_{id}, set $S \subseteq \{1, ..., n\}$, and hdr. This function returns a session key SK if $id \in S$ or an (undefined) pseudo-random value otherwise.

The transmission of a message with BE-related *metadata* (information required for key-establishment) is called a *session*, and a set of legitimate receivers (S) can be changed by BC every session. The BE scheme deployed throughout this paper is assumed to hold the following properties:

***Correctness*:** without all secret keys from S a party which executes $Enc(S)$ is not able to produce a session key for S (i.e., without k_{id} it is impossible to determine an output of $Dec_{k_{id}}(S, hdr)$),

***Resiliency*:** an adversary, despite having all keys from set R, is not able to compute a session key,

***Backward Secrecy*:** a new member of set S is not able to compute previous session keys,

***Forward Secrecy*:** a node removed from set S cannot compute the following session keys,

***Break-Backward Protection*:** an attacker who is able to capture arbitrary nodes (even from set S) cannot compute any previous session key.

For all introduced primitives, initial keys are selected uniformly at random (unless stated otherwise) from $\{0,1\}^\lambda$.

2.3 Attacker model

Throughout the paper, the following adversary model is assumed. The adversary is able to control network traffic (inject, modify, and block arbitrary messages), perform effective computations (in polynomial-time), and capture regular nodes. Furthermore, the last capability is not restricted to some portion of nodes or to some special type of them. The attacker is able to capture arbitrary sets of nodes, even privileged ones. However, we assume that the BC cannot be captured. The adversary's goal is to create a malicious session which would be accepted as a legitimate by at least one non-captured node. This also encompasses modification of a legitimate transmission, change of session order, or replay of a past transmission.

2.4 Security goals

Our proposals protect sessions within our adversary model as well as weaker adversaries achieving the following properties:

- no adversary is able to produce a session that would be accepted by a non-captured node,
- an adversary who captures unprivileged nodes or a passive adversary is not able to decrypt any encrypted session,
- an adversary who captures privileged nodes is not able to decrypt previous encrypted sessions.

These goals are realized through: *a) authentication*, which ensures that sessions are indeed broadcast by BC, *b) confidentiality* (optional) for keeping messages secret, *c) weak freshness*, which ensures that order of received sessions is correct, and *d)* assumed BE's properties.

One challenge is to achieve these strong security guarantees, the other is to realize the protocols in an efficient manner to be viable on resource-starved devices in a long-term deployment.

2.5 Notation

We use the following notation:

m: message indented for a group of legitimate receivers (can be authenticated or both authenticated and encrypted),
S: privileged set representation (set of privileged nodes IDs),
hdr: header generated by BE scheme,
SK: session key generated by BE scheme,
i: session number (sessions are numbered in ascending order from 1),
$prev$: local variable stored by a regular node. It denotes number of a last accepted session (that value is updated when a new session is *accepted*),
k_j^i: is secret key of j-th node in i-th session,
λ: security parameter,
$a \xleftarrow{R} B$: a is selected uniformly at random from set B,
$\|$: string concatenation,
$\{0,1\}^l$: set of all binary strings with length of l bits,
θ: null string.

3. APPROACH

3.1 Broadcast encryption scheme

Our content distribution method requires a secure and efficient BE scheme, as it is intended for hardware-limited network deployment in a presence of a powerful adversary. In our previous work we propose a lightweight, resilient broadcast encryption scheme [19, 20], which we assume as a default one throughout this paper. The scheme is based upon Berkovits's protocol [4], and in every session a new instance of this protocol is created. Deployed BE scheme is presented in two variants: *flat* where header increases linear with size of S, and *tree-based*, that optimizes this overhead. It holds all desired properties (Section 2.2), and in order to achieve break-backward protection, a special *key-update* routine is used. For every new session i the scheme uses a key-update procedure:

$$k_j^i = F_{s_j}(k_j^{i-1}). \quad (1)$$

This operation is performed by the BC (for every secret key of node from S) and by every privileged node in a given session. Every j-th node and BC are pre-loaded with a long-term secret s_j and an initial secret key k_j^0. The security requirement for that key-update scheme is that after a new key (k_j^i) is generated, its predecessor (k_j^{i-1}) must be removed permanently from the node's memory.

The secure BE scheme enables session key establishment for privileged nodes. In order to enable that establishment, a few values must be added to the transmission. A privileged set representation S, a header hdr, and a session's number (counter) i are required. The last value is introduced for two purposes. First, it is required for key synchronization between BC and nodes (Equation (1)). Secondly, it provides weak freshness, because consecutive numbers of the sessions allow to order them.

3.2 Protocol with digital signatures.

The complete protocol has to protect metadata (S, i, hdr) as well as message m. First, the natural choice for authentication is a digital signature scheme, and the construction based on this primitive is presented in Appendix A.1. However, in a hardware-limited environment, long-term deployment of digital signatures is inefficient, as we point out in Section 5. Moreover, the protocol requires implementation of four cryptographic primitives at nodes, which increases program storage. Because of these drawbacks, we provide similar security properties using more efficient symmetric cryptography.

3.3 Symmetric approaches.

In contrast to digital signatures, symmetric authentication does not provide *non-repudiation* (e.g., when a group shares the same key it is impossible to prove that a given message was produced by a given node, except when leveraging time [16]). In this setting, we resolve this issue through the asymmetry introduced by the BE scheme (i.e., BC holds all secret keys, while a regular node has only its own secret key).

BC		**j-th node**
for $n \in S$		
$\quad k_n^i = F_{s_n}(k_n^{i-1})$		
$(SK, hdr) = \mathcal{B}.Enc(S)$		
$iv \xleftarrow{R} \{0,1\}^\lambda$		
$a = S\|i\|hdr\|iv$		
$c = \mathcal{A}.Enc_{SK}(iv, a, m)$		
broadcast(S, i, hdr, iv, c)	$\longrightarrow$	**receive**(S, i, hdr, iv, c)
$i = i + 1$		**if** $j \notin S$ **or** $i \leq prev$
		$\quad$ **return** $FAIL$
		$k_j^i = F_{s_j}(k_j^{i-1})$
		$SK = \mathcal{B}.Dec_{k_j^i}(S, hdr)$
		$a = S\|i\|hdr\|iv$
		$m = \mathcal{A}.Dec_{SK}(iv, a, c)$
		if $m \neq \perp$
		$\quad prev = i$

Figure 1: Initial method of content distribution based on AEAD, BE, and key-update scheme. (When confidentiality is not required, BC sets null message to encrypt, and sends m instead of c. Next j-th node omits decryption.)

A protocol that deploys the AEAD scheme is presented in Figure 1. The BC sets the session number i, updates nodes' secret keys, and creates a session key SK with header hdr. Then, the AEAD encryption procedure with generated initialization vector iv and the session key is executed. This guarantees message confidentiality and authenticity. The message m is encrypted into a ciphertext c and the transmission S, i, hdr, iv, c is finally broadcast by BC. A regular node receives it, checks the privileged set and session number, synchronizes the key to session i, and obtains a session key from the header. Then, the node verifies the authenticity of the transmission and decrypts the ciphertext.

An alternative design is presented in Appendix A.2, where a combination of symmetric encryption with a MAC scheme is used.

An adversary without a valid session key cannot forge or create any fake transmission and cannot replay an old one. Due to BE's correctness, a capturing adversary has to know a node's secret key to send a new acceptable session to this node. Whenever a capturing adversary wants to create a new, malicious session, that session will be only accepted by captured nodes, as it is impossible to create a valid header for a given node without that node's secret keys. The following section, however, presents a more serious attack.

3.4 Attack on the symmetric approach

In the above-presented protocols, a capturing adversary cannot create a new valid session which would be accepted by any non-captured node, but can unfortunately undetectably modify broadcast transmission.

BC	Adversary	j-th node
...		
(as in Fig. 1)		
...		
broadcast		
$(S, i, hdr, iv, c) \longrightarrow$	**receive and block**	
$i = i + 1$	$k_x^i = F_{s_x}(k_x^{i-1})$	
	$SK = \mathcal{B}.Dec_{k_x^i}(S, hdr)$	
	$a = S\|\|i\|\|hdr\|\|iv$	
	$c' = \mathcal{A}.Enc_{SK}(iv, a, m')$	
	broadcast	
	$(S, i, hdr, iv, c') \longrightarrow$	**receive**
		...
		(as in Fig. 1)
		...

Figure 2: Successful attack on AEAD based protocol with one captured node ($x \in S$).

A successful attack on the presented protocol is shown in Figure 2. First, the adversary receives and blocks the session. The adversary can compute session key SK as it has access to secret keys of a privileged node (s_x and k_x^{i-1}). Then, the adversary creates his own plaintext m', encrypts and authenticates it. Finally, he sends the transmission to legitimate recipients with the original set representation, session number, header, and iv but with altered ciphertext. Regular nodes obtain the session key (because S, i, and hdr are valid) and verify the ciphertext, which passes because the session key is correct. Finally, the entire privileged set considers the malicious message as authentic. The attack is thus very effective, as the adversary only needs to capture a single node.

4. IMPROVED SCHEMES

The previous attack is possible because the adversary could reuse a legitimate header generated by the BC. The session key is known to the adversary, and can be used to send a fake message without modifying the header. One observation is that BE metadata (S, i, hdr) is sent along with protected content but is not connected with that content. Local secret keys of the nodes are only connected with the current session number and are not associated with the protected message. Such association would protect from the previous attack as secret nodes' keys for a given session would be strictly related to a given transmission.

4.1 Authentication-only protocol

First, we investigate a protocol where BC broadcasts messages that are only authenticated. The protocol is presented in Figure 3.

The main idea is to change the key-update phase from Equation (1). Now, the secret key of the j-th node for session i is generated using the previous secret key and a message

BC	j-th node
for $n \in S$	
$\widehat{k}_n^i = F_{s_n}(\widehat{k}_n^{i-1})$	
$k_n^i = F_{\widehat{k}_n^i}(m)$	
$(SK, hdr) = \mathcal{B}.Enc(S)$	
$a = S\|\|i\|\|hdr\|\|m$	
$c = \mathcal{A}.Enc_{SK}(\theta, a, \theta)$	
broadcast$(S, i, hdr, m, c) \longrightarrow$	**receive**(S, i, hdr, m, c)
$i = i + 1$	**if** $j \notin S$ **or** $i \leq prev$
	return $FAIL$
	$\widehat{k}_j^i = F_{s_j}(\widehat{k}_j^{i-1})$
	$k_j^i = F_{\widehat{k}_j^i}(m)$
	$SK = \mathcal{B}.Dec_{k_j^i}(S, hdr)$
	$a = S\|\|i\|\|hdr\|\|m$
	if $\mathcal{A}.Dec_{SK}(\theta, a, c) \neq \perp$
	$prev = i$ (success)

Figure 3: Secure authentication-only protocol.

transmitted during that session, as presented below:

$$\widehat{k}_j^i = F_{s_j}(\widehat{k}_j^{i-1}), \tag{2a}$$

$$k_j^i = F_{\widehat{k}_j^i}(m). \tag{2b}$$

Initially, the BC and the j-th node share $\widehat{k}_j^0$ and s_j, then the secret key for every session k_j^i is derived from $\widehat{k}_j^i$ and m from that session. To broadcast a message, the BC, with updated keys of privileged nodes, creates a session key SK and a header hdr, authenticates the message m using SK and broadcasts the metadata, message, and authentication tag. A regular node receives these, updates its key using the session number and the received message m (see Equation (2)), then obtains SK and verifies the authentication tag.

The presented protocols thwart the previous attack. An adversary cannot produce the appropriate secret keys of non-captured nodes for messages other than the one transmitted, without knowledge of the secret keys of these nodes. BE metadata is unique and verifiable for every message, thus it is impossible to reuse metadata with a message different from m.

4.2 Protocol with authenticated encryption

Unfortunately, the *trick* from the last section does not work when a message has to be also encrypted. It is impossible to generate secret keys using the ciphertext, because it creates a logic loop as the ciphertext is produced using the session key, which is derived from a header and the secret keys. Hence secret keys of the nodes must be connected with a given session in another way.

Our solution is to create and send a value associated with a message, but which does not provide any information to an adversary about that message (as a transmission should still remain secret to the adversary without a privileged key). The protocol sequence is depicted in Figure 4. The BC chooses a random r from $\{0,1\}^\lambda$ and concatenates it with the message m. From this concatenation, a fresh and pseudo-random iv is generated and used for key-update (as in the previous protocol). Next, the BC computes the header hdr, the session key, generates the ciphertext, and broadcasts a session. A regular node receives the transmission, computes

BC	j-th node
$r \xleftarrow{R} \{0,1\}^\lambda$	
$M = r \| m$	
$iv = F_0(M)$	
for $n \in S$	
$\widehat{k}_n^i = F_{s_n}(\widehat{k}_n^{i-1})$	
$k_n^i = F_{\widehat{k}_n^i}(iv)$	
$(SK, hdr) = \mathcal{B}.Enc(S)$	
$a = S \| i \| hdr \| iv$	
$c = \mathcal{A}.Enc_{SK}(iv, a, M)$	
broadcast(S, i, hdr, iv, c) $\longrightarrow$	**receive**(S, i, hdr, iv, c)
$i = i + 1$	**if** $j \notin S$ **or** $i \leq prev$
	return $FAIL$
	$\widehat{k}_j^i = F_{s_j}(\widehat{k}_j^{i-1})$
	$k_j^i = F_{\widehat{k}_j^i}(iv)$
	$SK = \mathcal{B}.Dec_{k_j^i}(S, hdr)$
	$a = S \| i \| hdr \| iv$
	$M = \mathcal{A}.Dec_{SK}(iv, a, c)$
	if $M \neq \perp$ **and** $iv = F_0(M)$
	$prev = i$ (success)

Figure 4: Secure protocol with authentication and encryption.

its secret key according to the session's number and iv, obtains SK and decrypts the ciphertext. Additionally, the node checks if iv was generated correctly from M and can then drop the first λ bits of M and process the original message m. As now iv is strictly tied to message and resultant ciphertext c, an adversary cannot modify it or reuse for any different message.

5. ANALYSIS

The initial scheme had a security flaw related to the key-update process, hence we modified this process. Now, secret keys for a given session are produced from previous keys and a message. This forms a one-way chain presented in Figure 5. It requires two executions[1] of a PRF for consecutive sessions, and $i + 1$ executions in the worst case (when a node joins S for the first time). Initially, BC shares two secret values with every node. Intuitively, it is easy to see that the new key-update procedure (Equation (2)) does not violate the properties of key-update from Equation (1), as keys generated by the last procedure are directly used to derive keys from the previous one.

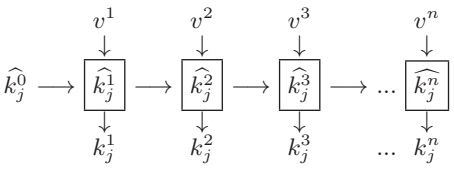

Figure 5: One-way chain produced by key-update scheme (depending on protocol's version v^i is m or iv in session i).

[1]Second execution can be optimized by XOR operation.

One could ask, instead of Equation (2), why a key-update scheme as follows is not used:

$$k_j^i = F_{s_j}(k_j^{i-1} \| v^i), \tag{3}$$

Unfortunately, such a scheme would be prone to a *blocking attack*. Let us assume that an adversary has blocked a few sessions intended for a given node. Then, that node cannot synchronize its key for the following sessions because it does not know the previous values v. However, in the current setting related attacks are possible. An attacker can send a message with a large session number, then the receiving node must perform many executions of a pseudo-random function, which can cause a *Denial of Service* (DoS). Some mitigation from such an attack may be given by *Proof-of-Work* protocols [13], where the sender has to prove that he has performed a given amount of computation. Another approach can be a special unicast message (with an encrypted key) to the node that was in the unprivileged set recently. Another issue is that previous keys should be removed permanently only when the node has successfully verified the authenticity of a given session. Without such checking, a similar de-synchronization attack can be carried out. Our schemes eliminate replay attacks through consecutive ordering of sessions.

Due to the resiliency of the BE scheme (see Section 2.2), a passive adversary or even an adversary with all keys from the unprivileged set cannot obtain any information about the transmitted ciphertext (if m is encrypted). When an adversary captures a privileged node, he is able to read the current and subsequent sessions (if the node remains in the privileged set), but he cannot obtain the previous secret key and previous session keys, due to the break-backward protection property of the BE scheme. Such an attacker cannot modify sessions created by protocols from Figures 3 and 4, in a way that this transmission would be accepted by a privileged non-captured node. This is because privileged secret keys for every session are tied to the transmitted message, hence even a capturing attacker cannot predict secret keys of non-captured nodes for a different message.

If the goal of an attacker is to create a new fake session, he must declare the set S in the header. Declaring set S without knowledge of all privileged secret keys results in a header hdr that will not produce an intended session key for non-captured nodes. Thus an attacker cannot authenticate and encrypt a message correctly. A non-captured node that was declared by an attacker in S would detect this attack, because it is declared as receiver but the message is not authenticated properly. In such a situation, a node is sure that the attack has occurred and, e.g., can trigger an alarm in a network. This is a significant advantage of the scheme, because an attacker can create undetectably a fake session for a given set, if and only if he knows all keys from that set. In other words, an adversary can *fool* only the nodes that he has captured, thus he has no gain from the attack.

This reasoning can be supported by formal arguments. Let us consider the scheme from Figure 4 and denote transmission in session i as t^i:

$$t^i = (S^i, i, hdr^i, iv^i, c^i), \tag{4}$$

and the session key in session i as SK^i. Let us consider a network where a strong adversary denoted as $\mathcal{O}$: *a)* has captured all nodes within the network except BC and node n (i.e., he knows $s_1, s_2, ..., s_{n-1}, \widehat{k}_1^0, \widehat{k}_2^0, ..., \widehat{k}_{n-1}^0$) , *b)* knows all

past transmission, *c)* has knowledge of all previous secret keys of n-th node $(\widehat{k}_n^1, \widehat{k}_n^2, ...)$ but does not know the long term secret s_n. Now, assume that $\mathcal{O}$, after l sessions, can produce such a session that is accepted as a valid one by a non-captured node n. Such a successful attack can be formalised as follows:

$$(S^x, x, hdr^x, iv^x, c^x) = \mathcal{O}(s_1, ..., s_{n-1}, \widehat{k}_1^0, ..., \widehat{k}_{n-1}^0, \widehat{k}_n^0, ..., \widehat{k}_n^l, t^1, ..., t^l), \quad (5)$$

and the n-th node after receiving and processing $(S^x, x, hdr^x, iv^x, c^x)$ terminates with success (see Figure 4).

As we assume that such attack is possible, it implies that:

$$n \in S^x \wedge x > prev, \quad (6)$$

because otherwise n-th node would discard transmission. From Equation (6) and Figure 4, one of the following must occur if $(S^x, x, hdr^x, iv^x, c^x)$ is accepted by the n-th node:

1. $\mathcal{O}$ is able to compute s_n or is able to compute k_n^x without s_n,
2. $\mathcal{O}$ is able to determine $\mathcal{B}.Dec_{k_n^x}(S^x, hdr^x)$'s execution without k_n^x,
3. $\mathcal{O}$ is able to create a valid c^x without the correct session key SK.

All these conditions lead to a contradiction as we have assumed corresponding security properties of the cryptographic primitives.

As an important element of our protocols, we employ AEAD, which is proved to be an effective cryptographic primitive even on hardware-limited platforms [21]. Our protocols require three implementations of cryptographic primitives (PRF, BE, and AEAD), while alternative approaches (see Appendix) require four primitives. Moreover, an advantage of many AEAD schemes is the ability to operate in a *one-pass* manner (both authentication and encryption are processed simultaneously). In the case of the protocol from Figure 4, a small transmission overhead is introduced. In order to create fresh and pseudo-random iv, the concatenation $r\|m$ is used and sent. The value r protects m from guessing attacks of passive adversaries (without r it would be possible to guess m as iv is known).

Computational efficiency of the protocols mainly depends on selected cryptographic primitives. For instance, AVR-Crypto-Lib [1] provides a wide range of PRFs and symmetric encryption schemes optimized for hardware-limited environments.

However, the major computational cost is associated with authentication, and the performance of various methods are compared in Table 1. Storage required by every method is expressed in bytes, while initialization and execution costs are expressed in milliseconds of required CPU's time. The table sums up significant results from literature, which were obtained using the standard WSN platform (ATmega128 or similar). The upper part of the table describes symmetric methods (MACs) while the bottom part presents digital signatures (classical schemes as well as lightweight solutions). The results show a difference of several orders of magnitude between these two types of authentication, hence schemes employing digital signatures may be unacceptable for long-term deployment in the considered environment.

Table 1: Efficiency of authentication methods measured on standard WSN platform. Results (except Code size) are given in milliseconds.

Method	Code size	Init cost	Messages length 16B	32B	48B	64B
CMAC [21]	2240B	0.7	0.4	0.7	1.0	1.4
GMAC [21]	5706B	2.6	1.7	2.5	3.2	4.0
GMAC$_{256B}$ [21]	6220B	3.1	1.2	1.7	2.3	2.8
GMAC$_{4KB}$ [21]	10271B	7.2	0.7	1.0	1.3	1.6
GMAC$_{8KB}$ [21]	14108B	25.1	0.5	0.7	0.9	1.2
HMAC$_{SHA1}$ [1]	5252B	0.0	4.7	4.7	4.8	4.8
HMAC$_{MD5}$ [1]	6348B	0.0	3.6	3.6	3.7	3.7
ECDSA^{Sign} [12]	19308B	3493	2001	2001	2001	2001
ECDSA^{Vrfy} [12]	19308B	3493	2436	2436	2436	2436
ECDSA^{Sign} [5]	43200B	-	918	918	918	918
ECDSA^{Vrfy} [5]	43200B	-	938	938	938	938
ECDSA$^{Sign}_{160}$ [7]	3682B	-	810	810	810	810
ECDSA$^{Sign}_{192}$ [7]	3979B	-	1240	1240	1240	1240
ECDSA$^{Sign}_{224}$ [7]	4812B	-	2190	2190	2190	2190
RSA$^{Sign}_{1024}$ [7]	6292B	-	10990	10990	10990	10990
RSA$^{Vrfy}_{1024}$ [7]	1073B	-	430	430	430	430
RSA$^{Sign}_{2048}$ [7]	7736B	-	83260	83260	83260	83260
RSA$^{Vrfy}_{2048}$ [7]	2854B	-	1940	1940	1940	1940
NTRUSign^{Sign} [5]	11300B	-	619	619	619	619
NTRUSign^{Vrfy} [5]	11300B	-	78	78	78	78
XTR-DSA^{Sign} [5]	24300B	-	965	965	965	965
XTR-DSA^{Vrfy} [5]	24300B	-	2009	2009	2009	2009
Ed25519^{Sign} [8]	28883B	-	1451	1451	1451	1451
Ed25519^{Vrfy} [8]	28883B	-	2039	2039	2039	2039

6. RELATED WORK

The problem of key establishment and agreement for hardware-limited networks is widely discussed in the literature, and Kim et al. [10] proposed a systematization of security properties for these protocols. However, a majority of the proposed methods provide security properties in the face of weaker adversaries than these considered in our work. An interesting security service for considered environments is pairwise key establishment. The protocols [6] of this class are efficient and can tolerate a threshold number of compromised users, however the communication scenario is different than considered in this work.

Broadcast encryption schemes that satisfy the properties listed in Section 2.2 are presented in our previous work [19, 20]. The proposals take advantage an *one-time* BE scheme to provide additional security properties. These approaches also try to deal with capturing adversary in a distributed wireless sensor network and [20] also outlines an effective content distribution protocol, which is similar to the protocol from Figure 7 in Appendix A.2. Unfortunately, these content distribution schemes are prone to the attack presented in Section 3.4 based on our strong adversary model.

Combination of confidentiality and authentication is investigated in the literature for the one-to-one symmetric case. Bellare and Namprempre [3] analyze compositions of symmetric encryption with message authentication codes. Although many compositions exist, their work points out the Encrypt-then-Mac scheme (first, the message is encrypted and then an authentication tag is computed over a ciphertext) as most secure. It is also the most efficient approach as a node first checks the authentication tag and does not process a message with an invalid tag. Rogaway [18] proposes and analyzes the AEAD method, which achieves the same level of security as the Encrypt-then-Mac composition but in

one cryptographic primitive. Both papers analyze security in a two-party symmetric setting where two communicating parties share the same secret key.

Mauw et al. [14] propose a set of the mechanisms for forward secure communication in WSN. That set includes mechanisms for authentication, confidentiality, and freshness. The paper also proposes a key-update scheme for achieving certain security properties, but these methods focus on communication from nodes to the BC. While our paper focuses on security from BC to the nodes, these methods can be combined for two-way secure communication.

7. CONCLUSIONS

The main results of our work are two methods that provide message secrecy and authenticity for broadcast transmissions in hardware-limited environments, in the face of a powerful adversary. We employed AEAD scheme and suggest it for standard deployment, as it decreases storage and computational overheads compared to prior approaches. Our methods with only symmetric cryptography achieve strong security guarantees in an adversarial environment. A passive adversary or an adversary with unprivileged keys cannot obtain any secret or session key, and cannot decrypt a ciphertext or modify any message undetectably as a consequence. Even if the adversary is able to capture privileged nodes, his abilities are significantly restricted. He is able to decrypt current session, but he cannot determine previous secret keys of captured nodes, hence he is not able to get any previous session key. Such adversary can create a fake session only for a set of nodes, which is already captured, thus the adversary does not have any advantage. The only possible attacks are blocking and other DoS attacks, which may be an interesting subject for future work.

8. ACKNOWLEDGMENT

We would like to thank Laurent Chuat and anonymous reviewers for their valuable feedback. This work is supported by the National Science Center (NCN), under Grant with decision's number DEC-2011/01/N/ST7/02995, by a gift from Google, and by ETH Zurich.

9. REFERENCES

[1] AVR-Crypto-Lib. `http://www.das-labor.org/wiki/AVR-Crypto-Lib/en`.

[2] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *Proceedings of Foundations of Computer Science (FOCS)*, 1997.

[3] M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *J. Cryptol.*, September 2008.

[4] S. Berkovits. How to broadcast a secret. In *Theory and Application of Cryptographic Techniques*, 1991.

[5] B. Driessen, A. Poschmann, and C. Paar. Comparison of innovative signature algorithms for wsns. In *Proceedings of the first ACM conference on Wireless network security*, 2008.

[6] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of ACM conference on Computer and Communications Security (CCS)*, 2002.

[7] N. Gura, A. Patel, W. Arvinderpal, H. Eberle, and S. C. Shantz. Comparing elliptic curve cryptography and RSA on 8-bit CPUs. 2004.

[8] M. Hutter and P. Schwabe. NaCl on 8-bit AVR microcontrollers. In A. Youssef and A. Nitaj, editors, *Progress in Cryptology – AFRICACRYPT*, 2013.

[9] J. Katz and Y. Lindell. *Introduction to modern cryptography*. CRC Press, 2008.

[10] Y. Kim, A. Perrig, and G. Tsudik. Communication-efficient group key agreement. In M. Dupuy and P. Paradinas, editors, *SEC*, 2001.

[11] C. Kuo, M. Luk, R. Negi, and A. Perrig. Message-In-a-Bottle: User-friendly and secure key deployment for sensor nodes. In *Proceedings of the ACM Conference on Embedded Networked Sensor System (SenSys)*, Nov. 2007.

[12] A. Liu and P. Ning. TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks. In *Proceedings of the 7th international conference on Information processing in sensor networks*, 2008.

[13] D. Liu and L. J. Camp. Abstract and overview proof of work can work, 2006.

[14] S. Mauw, I. V. Vessem, and B. Bos. Forward secure communication in wireless sensor networks. In *Proceedings of International Conference on Security in Pervasive Computing (SPC)*, 2006.

[15] A. Pathan, H.-W. Lee, and C. S. Hong. Security in wireless sensor networks: issues and challenges. In *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, 2006.

[16] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler. Spins: security protocols for sensor networks. *Wirel. Netw.*, 2002.

[17] D. H. Phan, D. Pointcheval, and M. Strefler. Security notions for broadcast encryption. In *Proceedings of international conference on Applied Cryptography and Network Security (ACNS)*, 2011.

[18] P. Rogaway. Authenticated-encryption with associated-data. In *Proceedings of the 9th ACM conference on Computer and communications security*, 2002.

[19] P. Szalachowski and T. H.-J. Kim. Secure broadcast in distributed networks with strong adversaries. *Security and Communication Networks (under revision)*, 2015.

[20] P. Szalachowski and Z. Kotulski. One-time broadcast encryption schemes in distributed sensor networks. *International Journal of Distributed Sensor Networks*, 2012.

[21] P. Szalachowski, B. Ksiezopolski, and Z. Kotulski. Cmac, ccm and gcm/gmac: Advanced modes of operation of symmetric block ciphers in wireless sensor networks. *Inf. Process. Lett.*, March 2010.

[22] Y. Wang, G. Attebury, and B. Ramamurthy. A survey of security issues in wireless sensor networks. *IEEE Communications Surveys & Tutorials*, 2006.

[23] T. Zia and A. Zomaya. Security issues in wireless sensor networks. In *Proceedings of the International Conference on Systems and Networks Communication*, 2006.

APPENDIX

A. ADDITIONAL SCHEMES

In order to present alternative schemes, few additional cryptographic primitives have to be introduced. Primitive used for providing data confidentiality is *Symmetric Encryption Scheme* (SE) and it is defined as following algorithms $\mathcal{SE} = (Enc, Dec)$:

- $Enc_k(m)$ encrypts message (plaintext) m under key k and returns ciphertext c as a result,
- $Dec_k(c)$ takes a key k, ciphertext c, and returns decrypted message m or $\perp$ if decryption failed.

For ensuring authentication *Message Authentication Code* (MAC) is introduced $\mathcal{M} = (Mac, Vrfy)$, where:

- $Mac_k(m)$ for key k and message m produces authentication message (tag) σ,
- $Vrfy_k(m, \sigma)$ for key k, message m, and tag σ returns *True* if σ is a valid tag for m under k or *False* otherwise.

Digital Signatures (DS) is the only introduced asymmetric primitive and it is a triple $\mathcal{D} = (Gen, Sign, Vrfy)$ of next procedures:

- $Gen()$ produces pair of keys (sk, pk), where sk is private (signing) key and pk is public (verifying) key,
- $Sign_{sk}(m)$ for private key sk and message m returns signature σ,
- $Vrfy_{pk}(m, \sigma)$ takes public key pk, message m, and signature σ. Function returns *True* if σ is a valid signature for m or *False* otherwise.

It is assumed that MAC and DS are resistant to adaptive chosen-message attacks, and SE is resistant to adaptive chosen-plaintext attacks while. We also assume that BC's asymmetric keys are generated in advance with effective security level of λ bits, and a public-key is pre-loaded on every regular node.

A.1 Scheme with Digital Signatures

Figure 6 depicts a scheme with digital signatures. First BC sets session number i, then updates nodes' keys for that session, and using them computes header hdr and session key SK. Plaintext message m is now encrypted[1] using SK and then digital signature is created. Whole transmission encompasses S, i, hdr, c, and σ. Node receives it, checks freshness of the session ($i \leq prev$) and its membership in set S, finally node verifies digital signature. Then (if session is fresh, node is privileged, and signature is genuine) node updates his key using $k_j^i = F_{s_j}(k_j^{i-1})$ and finally computes session key SK. With that key ciphertext is decrypted and the last action is to update $prev$ to value i, what means that session i was accepted and message m's processing can be started.

Because only BC has its signing key, even capturing attacker is not able to forge digital signature, thus he cannot modify legitimate messages or create new one which would be accepted by regular nodes. To deploy digital signatures each node must have legitimate public key pk of the BC, what can be pre-loaded before deployment.

BC	j**-th node**
for $n \in S$	
$\quad k_n^i = F_{s_n}(k_n^{i-1})$	
$(SK, hdr) = \mathcal{B}.Enc(S)$	
$c = \mathcal{SE}.Enc_{SK}(m)$	
$\sigma = \mathcal{D}.Sign_{sk}(S\|\|i\|\|hdr\|\|c)$	
broadcast$(S, i, hdr, c, \sigma) \longrightarrow$	**receive**(S, i, hdr, c, σ)
$i = i + 1$	**if** $j \notin S$ **or** $i \leq prev$ **or not**
	$\quad \mathcal{D}.Vrfy_{pk}(S\|\|i\|\|hdr\|\|c, \sigma)$
	$\quad$ **return** $FAIL$
	$k_j^i = F_{s_j}(k_j^{i-1})$
	$SK = \mathcal{B}.Dec_{k_j^i}(S, hdr)$
	$m = \mathcal{SE}.Dec_{SK}(c)$
	$prev = i$ (success)

Figure 6: Scheme with Digital Signatures.

A.2 Scheme with Encrypt-then-Mac

Protocol which uses MAC with symmetric encryption is presented in Figure 7. Like before BC sets session number, updates nodes secret keys and creates session key with header. Then EtM composition [3] is deployed, what means that first plaintext is encrypted[1] and resultant ciphertext is authenticated using message authentication code. Both these operations require a separate secret key. Message m is encrypted into ciphertext c and transmission S, i, hdr, c is authenticated by MAC tag σ. Finally BC broadcasts S, i, hdr, c, σ that is received by regular nodes. A regular node checks privileged set and session number, synchronizes key to session i, obtains session key from header, and generates keys for authentication and decryption. Finally the node verifies the tag of the transmission and decrypts the ciphertext.

That method has few disadvantages in comparison to AEAD-based scheme (Figure 3). First, regular node needs four implementations of cryptographic protocols, what can be problematic in resource-constrained environments. The second issue is that two distinct, cryptographic keys must be generated from session key SK. Scheme is also vulnerable to the attack presented in Section 3.4.

BC	j**-th node**
for $n \in S$	
$\quad k_n^i = F_{s_n}(k_n^{i-1})$	
$(SK, hdr) = \mathcal{B}.Enc(S)$	
$K_m = F_{SK}(0)$	
$K_e = F_{SK}(1)$	
$c = \mathcal{SE}.Enc_{K_e}(m)$	
$\sigma = \mathcal{M}.Mac_{K_m}(S\|\|i\|\|hdr\|\|c)$	
broadcast$(S, i, hdr, c, \sigma) \longrightarrow$	**receive**(S, i, hdr, c, σ)
$i = i + 1$	**if** $j \notin S$ **or** $i \leq prev$
	$\quad$ **return** $FAIL$
	$k_j^i = F_{s_j}(k_j^{i-1})$
	$SK = \mathcal{B}.Dec_{k_j^i}(S, hdr)$
	$K_m = F_{SK}(0)$
	$K_e = F_{SK}(1)$
	if $\mathcal{M}.Vrfy_{K_m}(S\|\|i\|\|hdr\|\|c, \sigma)$
	$\quad m = \mathcal{SE}.Dec_{K_e}(c)$
	$\quad prev = i$ (success)

Figure 7: Scheme with Encrypt-then-Mac.

[1]When confidentiality is not required, BC omits encryption and sends m instead of c and j-th node omits decryption.