BAP: Broadcast Authentication Using Cryptographic Puzzles*

Patrick Schaller, Srdjan Čapkun, and David Basin

Computer Science Department, ETH Zurich
ETH Zentrum, CH-8092 Zurich, Switzerland
{patrick.schaller,srdjan.capkun,david.basin}@inf.ethz.ch

Abstract. We present two broadcast authentication protocols based on delayed key disclosure. Our protocols rely on symmetric-key cryptographic primitives and use cryptographic puzzles to provide efficient broadcast authentication in different application scenarios, including those with resource-constrained wireless devices such as sensor nodes. The strong points of the protocols proposed are that one protocol allows instantaneous message origin authentication, whereas the other has low communication overhead. In addition to formalizing and analyzing these specific protocols, we carry out a general analysis of broadcast authentication protocols based on delayed key disclosure. This analysis uncovers fundamental limitations of this class of protocols in terms of the required accuracy of message propagation time estimations, if the protocols are to guarantee security and run efficiently.

1 Introduction

Recent research in broadcast authentication for wireless networks [26, 25, 31, 22, 21] addresses the question of how to develop efficient mechanisms and protocols for broadcast message authentication in networks of low-cost and resource-constrained wireless devices (e.g., sensor networks). The main challenge here concerns efficiency: reducing the cost of message generation by the sender and the verification of message authenticity by the receiver. The approaches proposed include the use of symmetric-key primitives and delayed key disclosure [26, 27, 16, 8], one-time signatures [25, 6], and solutions based on devices' awareness of presence in the vicinity of the sender [31].

In this work, we propose two new broadcast authentication protocols based on delayed key disclosure. Our protocols are based on symmetric-key cryptographic primitives and rely on cryptographic puzzles to provide efficient broadcast authentication in a wide range of application scenarios, including those with resource-constrained wireless devices such as sensor nodes. The first protocol (BAP-1) achieves instantaneous message-origin authentication upon message

* The work presented in this paper was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

reception. Our second protocol (BAP-2) achieves broadcast authentication using a single transmission per authenticated message.

Similar to previously proposed broadcast authentication protocols based on delayed key disclosure [26], we also use authenticated keys derived from one-way (hash) chain elements. However, instead of relying on delayed key transmission (and consequently, delayed message verification), we use cryptographic puzzles to hide the key up to the point in time when the key can be safely disclosed. In our first protocol (BAP-1), the key is sent in a puzzle before the message, thereby allowing instantaneous message verification. This allows the message to be verified upon reception, assuming that the puzzle is solved by the time the entire message is received. Our second protocol (BAP-2) achieves broadcast authentication with delayed key disclosure by transmitting a single message containing the original message, its message authentication code (MAC), and the key. BAP-2 therefore reduces the communication overhead in terms of the number of messages needed for message authentication. We provide a detailed security analysis of both protocols and use this analysis to highlight applications where each of these protocols is suitable.

In addition to proposing and analyzing our protocols, we carry out a general analysis of broadcast authentication protocols based on delayed key disclosure. This analysis uncovers fundamental limitations of this class of protocols in terms of the required accuracy of message propagation time estimations and of time synchronization, if the protocol is to guarantee security and run efficiently. More specifically, our analysis shows that, if a protocol is to work both securely and efficiently, the message propagation times in the network must be known in advance. This requirement limits the applicability of this class of protocols. However, the design of these protocols makes them well suited for networks with known (or predictable) topologies and for delay-tolerant networks of low-cost and resource-scarce devices.

In summary, we make the following contributions in this work. First, we propose two new protocols for broadcast authentication based on delayed key disclosure. These proposed protocols represent two new points in the security-performance subspace of this class of protocols. Second, we analyze the class of broadcast authentication protocols based on delayed key disclosure, where we highlight the importance of the accurate estimation of message propagation times for the performance of protocols in this class.

The rest of the paper is organized as follows. In Section 2, we state the problem and describe our system and attacker model. In Section 3, we describe our broadcast authentication protocols. In Section 4, we present our analysis of broadcast authentication protocols based on delayed key disclosure. In Section 5, we survey related work and we conclude the paper in Section 6.

2 Problem Statement and Background

The objective of *broadcast authentication* is to guarantee message-origin authentication and hence also the integrity of messages transmitted by a sender to

the receivers. That is, all receivers in the network can verify that each broadcasted message has been generated by the claimed source and that it has not been modified in transmission. This problem is generally solved using asymmetric cryptographic primitives such as digital signatures [28, 11], where the sender signs a message with his private key and broadcasts the signature along with the message. Any receiver holding the sender's public key can verify the correctness of the signature and validate if the message was indeed generated by the claimed sender.

Achieving efficient broadcast authentication using symmetric-key primitives is more challenging. One naive solution is that the sender appends message authentication codes (MACs) to the messages, generated with the keys that the sender shares with the receivers (one MAC per receiver). This solution clearly does not scale and adds substantial overhead to the network, especially in the case of multi-hop wireless networks. Another solution is that the sender shares a single key with all receivers, in which case a single MAC is sufficient to authenticate the sender. The downside of this solution is that a single compromised node is sufficient to compromise the whole scheme.

In this work, our goal is to provide a scalable and efficient broadcast authentication protocol for resource-constrained devices using symmetric-key primitives. We now provide a definition of broadcast authentication, incorporating both message-origin authentication and a parameterized notion of recentness.

Definition 1.

i) *A broadcast protocol guarantees (message-origin) authentication iff whenever a node B receives a message m and concludes that it was sent by node A, then m was indeed sent by A.*

ii) *A broadcast protocol guarantees T-recentness iff whenever a node B receives a message m and concludes that it was sent within T time units before its reception, then m was indeed sent within this time interval.*

iii) *A broadcast protocol guarantees T-authentication iff it guarantees both authentication and T-recentness.*

2.1 System and Attacker Model

We now describe the class of systems we consider. A system consists of a collection of nodes connected via (e.g., wireless) communication links. The nodes can be connected directly or can communicate over multiple hops. We neither impose restrictions on the network topology nor do we assume that the network nodes are aware of the network topology or of their respective locations. The network is operated by an authority. This authority can be on-line, meaning that the authority operates on-line servers (that can be reached by single-hop or multi-hop communication), or off-line, meaning that the services of the authority cannot be reached over the network. We assume that all network nodes can establish pairwise secret keys. This can be achieved by manually pre-loading all keys onto the nodes in a network setup phase, using probabilistic key pre-distribution protocols [12, 5], or through an on-line key distribution center [17]. We also assume

that every network node holds authentic (public) commitments of hash chains generated by other network nodes. These commitments are distributed by the network authority prior to network operation. We discuss this further in Section 2.2. Finally, any network node is a potential broadcast message source and all other nodes are designated message receivers.

We adopt the following attacker model. We assume that the attacker Mallory (M) controls the communication channel in the sense that he can insert, eavesdrop, delay, schedule, modify, or block transmitted messages. Additionally the attacker can send messages ahead of time, if he can predict them. We assume that the attacker is not a part of the network controlled by the authority and cannot gain access to network keys or disclose any messages exchanged between the nodes or between the nodes and the authority in the setup phase. We also assume that network nodes can be compromised by the attacker. However, we will assume that the sender participating in the broadcast authentication protocol is not compromised, as then broadcast authentication would be meaningless. Finally, we require M to be *computationally bounded*; specifically we assume that we can create cryptographic puzzles that are time-consuming for the attacker to solve (see Section 3.3).

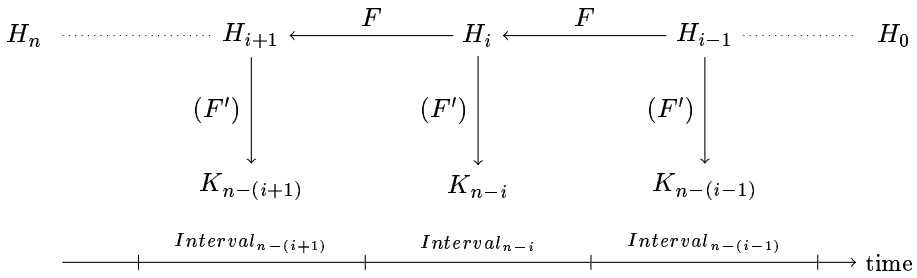
Before presenting our solutions to the broadcast authentication problem, we briefly describe authentication based on one-way chains using delayed key disclosure.

2.2 Authentication using One-Way-Chains and Delayed Key Disclosure

The use of hash-chains for authentication was first introduced by Lamport in [19]. Hauser et al. used hash-chains in [16] to authenticate routing updates in routing protocols by assigning the elements of a hash-chain to points in time. Cheung [8] added the notion of late key disclosure, which was then used by Perrig et al. for broadcast authentication in TESLA [26].

The basic idea is as follows: The sender (whose messages should be authenticated) creates a hash-chain by selecting a random element H_0 as the root and by iteratively applying to it a one-way (pre-image-resistant) function F. This produces the sequence H_0, H_1, ..., H_n, where $H_i = F^i(H_0)$, for $1 \leq i \leq n$. As F is one-way, a receiving node possessing H_i cannot feasibly compute the predecessor H_{i-1}; only the owner of the root can do so, by computing forward from H_0. However, given a string s, any node possessing H_i can easily check if $s = H_{i-1}$ by checking if $F(s) = H_i$. The sender then commits to the hash chain by distributing H_n in an authentic way to each receiver. Moreover, if required, the receiver synchronizes his clock with the sender's at this point.

We associate hash values with keys in the following way. For each H_i, we apply another one-way function F' to derive a key K_{n-i}, for the corresponding time interval $n - i$. F' is used to avoid using the string H_i for two different purposes: as a hash value in the chain and as a key. In the following, we will use the chain element and the corresponding key interchangeably, since this does not affect our observations.



To authenticate a messages m, the sender assigns the message to a time interval. Then, to send m in the $(n-i)$th time interval, the sender appends to m a keyed MAC, $MAC_{K_{n-i}}(m)$, as well as the chain element for the preceding time interval, H_{i+1}, in clear text. This hash value opens the commitment to H_{i+1}, and hence the receiver can determine the key $K_{n-(i+1)}$ and thereby authenticate the previous message.

3 Broadcast Authentication using Cryptographic Puzzles

In this section, we present our broadcast authentication protocols. We begin by stating our assumptions. We assume that the sender and the receivers have synchronized clocks. Recently, several proposals for (secure) time synchronization in wireless networks have emerged that successfully address this problem [13, 23, 30]. In wired networks, we assume that the nodes are securely synchronized using online synchronization servers or precise local clocks. In some application scenarios, we can also rely on nodes synchronizing their clocks using GPS [15] receivers. In Section 4, we will analyze the implications of time synchronization in more detail. We further assume that a broadcasting node (A) has generated a one-way (hash) chain and distributed its corresponding commitment to the designated receivers (B) in an authentic manner. As described in Section 2.2, the elements of this chain are used by the sender to derive message authentication keys, whereas the chain commitment value is used by the receivers to verify the authenticity of the used keys. Finally, we assume that the sender can create, and receivers can solve (within some given time), cryptographic puzzles. In Section 3.3, we discuss different puzzle schemes and their implications for the proposed broadcast authentication protocols.

We now present two broadcast authentication protocols based on cryptographic puzzles and delayed key disclosure, which we call BAP-1 and BAP-2.

3.1 BAP-1

BAP-1 is designed to achieve instantaneous message verification upon message receipt. The protocol is shown on Figure 1. In this protocol, the message sender first chooses the cryptographic key k_i, which corresponds to the time interval $i = [t_i, t_{i+1}[$ (where $[t_i, t_{i+1}[$ denotes the set $\{t \in \mathbb{R} | t_i \leq t < t_{i+1}\}$), according

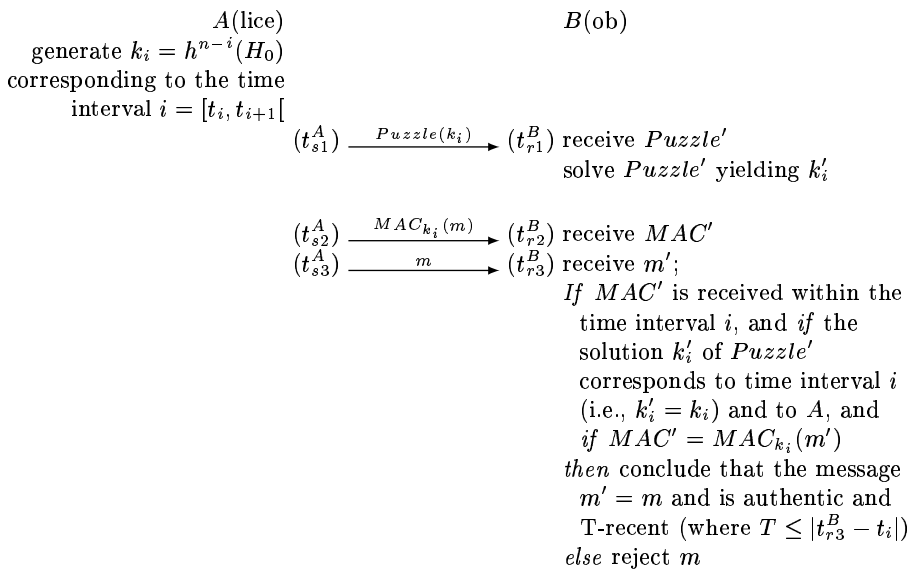


Fig. 1. BAP-1 protocol. The protocol achieves broadcast authentication through delayed key release based on cryptographic puzzles. Instant message authentication is achieved if the receiver solves the puzzle, and therefore obtains the key, before receiving the message. All messages received by B are marked with $'$ to denote that they might have been modified in transit by an attacker.

to the scheme described in Section 2.2. The sender then encapsulates k_i within a cryptographic puzzle $Puzzle(k_i)$, and broadcasts the puzzle at time t^A_{s1}. The puzzle serves to hide the key for a given time, which depends on the puzzle complexity and on the solver's processing speed. Immediately after the last bits of the puzzle have been sent (at t^A_{s2}), the sender starts transmitting the message authentication code $MAC_{k_i}(m)$, computed over the broadcast message m, using the key k_i contained in the puzzle. Finally, when the last bits of $MAC_{k_i}(m)$ are sent (at t^A_{s3}), the sender transmits the broadcast message m.

From the receiver's side, the protocol proceeds as follows: At time t^B_{r1}, the receiver B receives the puzzle $Puzzle'$ and starts solving it to retrieve the key k'_i. Here, all messages received by B are marked with $'$ to denote that they may have been modified in transit by the adversary. Concurrent to solving the puzzle, B receives MAC' and subsequently the message m'. In order to verify the authenticity of the message immediately upon its receipt, the receiver must solve the puzzle before receiving the last bits of the message (i.e., prior to t^B_{r3}). This case is optimal in terms of verification speed and we study it in more detail in Section 3.1. After the receiver solves the puzzle, he then verifies (i) if MAC' was received within the time interval i, (ii) if the key k'_i is indeed authentic and corresponds to the current time slot i and to the claimed sender A (i.e., if k'_i can be bound to the public commitment H_n previously distributed

by the sender and to its intended time release slot, see Section 2.2), and (iii) if the message authentication code $MAC_{k_i}(m')$ computed with the derived key over the received message equals the received authentication code MAC'. If all verifications succeed, then the receiver concludes that the message $m' = m$ is both authentic (i.e., generated by the claimed source A) and T-recent (i.e., has been sent by A within T time units before reception, where $T \leq |t_{r3}^B - t_i|$). Hence, the receiver concludes that the message is T-authentic according to Definition 1.

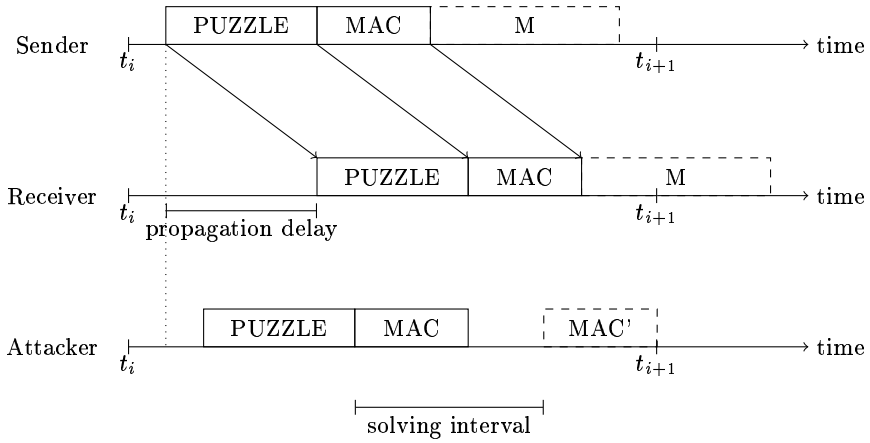


Fig. 2. The diagram shows the sender and a receiver of a broadcast message. At the bottom, the attacker receives the message earlier than the receiver. In order to create a valid, spoofed message, he must not only solve the puzzle and create a valid MAC (MAC'), he must also have it delivered to the receiver inside the validity window of the key.

Security and Performance Analysis From the previous protocol description, we can derive the main protocol performance condition: the BAP-1 protocol achieves optimal performance in terms of message verification time, if the receiver can solve the puzzle by the time that he receives the last bit of the message. If this condition is met, the receiver can verify the message immediately upon reception. The fulfillment of this condition depends on the receiver's processing speed (i.e., the speed that it can solve puzzles), the MAC and the message propagation delays (depending on the speed of the underlying communication channel and the network topology), and on the transmission time (depending on the bit rate and the size of the MAC and the message). Note that for the message verification to succeed, the MAC needs to reach the receiver within the key's validity window.

The main security condition of this protocol is that the attacker M is not able to solve the puzzle before the validity of the key expires (each key is valid

for only a certain time interval, as described in Section 2.2). This prevents the attacker from being able to create a valid MAC for his own messages. Besides solving the puzzle and jamming the original MAC, the attacker has to create his own MAC and deliver it before the validity window ends. The relationship between the message propagation time, key validity window, and the minimum time within which the attacker needs to solve the puzzle is shown on Figure 2. The sender can enhance the security level of the protocol by creating puzzles that are harder (and thereby require more time) to solve or by tightening the key validity window. For a more detailed discussion on cryptographic puzzles and how they are constructed, see Section 3.3. Note that the receiver has much more time to solve the puzzle than the attacker. Namely, to achieve instant verification, the receiver needs to solve the puzzle before the message has been received, which will typically be after the validity period of the key has expired (depending on the message size). Alternatively, the receiver can continue solving the puzzle after he receives the message, in which case the message will be verified with a delay. We want to point out that a puzzle in combination with the hash-chain and the corresponding disclosure schedule protects the MAC (and therefore the integrity and authenticity of the corresponding message) during transportation up to the point of reception. Therefore the required restrictions on the computational resources of the adversary do not depend on the computational power of the honest nodes involved, but mainly depend on the transmission time given by the communication medium and the network topology.

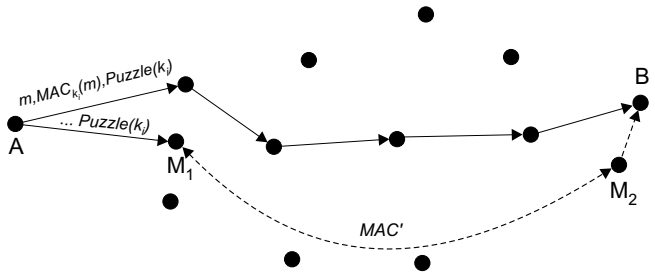


Fig. 3. This figure shows a scenario in which the attacker controls two devices (M_1 and M_2), M_1 located close to the sender (A), whereas M_2 is located close to the receiver (B). Furthermore the attacker nodes are connected by a fast link (dashed line).

From Figure 2, we can draw conclusions about how the attacker's physical distance from the sender and the receiver affects his ability to successfully break the scheme. Namely, if the attacker is located close to the sender, he will quickly obtain the puzzle (i.e., the propagation delay of the puzzle from the sender to the attacker will be short), and therefore can start solving it earlier. However, after solving the puzzle, the attacker still needs to forge a new message, create a new MAC', and send MAC' to the attacked receiver. Therefore being close

to the sender (instead of to the receiver) means that the propagation time of MAC' from the attacker to the receiver will be long, thus reducing the attacker's ability to solve the puzzle and send MAC' before the key becomes invalid. This is illustrated in Figure 3, where the attacker M_1 eavesdrops on the puzzle sent by the sender and, after solving it, forges MAC', and sends it to the receiver (using multi-hop communication). Equally, if the attacker is placed close to the receiver, then he will get the puzzle at the same time as the receiver, but will have almost no time to solve it before the first bits of $MAC_{k_i}(m)$ are received by the receiver or before the key's validity expires. From these considerations we conclude that the optimal placement for the attacker is the one that minimizes the sum of the puzzle propagation delay from the sender to the attacker and the MAC' propagation delay from the attacker to the receiver. This means that the attacker is best located along the fastest (e.g., shortest) communication path between the sender and the receiver.

Note that, if the attacker controls two devices, M_1 and M_2, one located close to the sender and the other close to the receiver, connected with a fast (e.g., wired) link, then the attacker can shorten the communication time of MAC' to the receiver, and therefore gain time to solve the puzzle and obtain the key. This scenario is illustrated in Figure 3. However, if the broadcast source has a direct link to all receivers, the attacker cannot increase his chances of success, as he cannot speed up the propagation of the signal between the sender and the receiver.

These arguments show that the key validity interval and the puzzle hardness need to be appropriately set to reflect the attacker's expected strength and his possible locations. In the following analysis, we consider the worst-case scenario, in which the attacker controls two devices placed in the vicinity of the sender and the receiver, connected via a fast wired link (i.e., effectively forming a worm-hole between two locations). We start by stating the conditions for a message to be successfully authenticated by an honest receiver:

- The key k_i included in the puzzle must be an element of the hash chain.
- If the puzzle has been received at t^B_{r1}, then t^B_{r1} must be in the time interval associated to the key k_i in the puzzle.
- The arrival time t^B_{r2} of $MAC_{k_i}(m)$ must be before $t^B_{r1} + \delta_M$, the point of time, when Bob assumes that Mallory would have solved the puzzle and therefore is able to create his own message $\tilde{m}$ with a valid MAC $MAC_{k_i}(\tilde{m})$. Note that this is where the assumption about the computational power of Mallory comes into play.
- The message m is successfully authenticated by $MAC_{k_i}(m)$.

Therefore we get the following conditions on the parameters for the protocol to achieve the desired security properties:

- $t^B_{r1} \in [t_i, t_{i+1}[$, i.e., the puzzle has been received in the validity period of the corresponding key (element of the hash-chain).
- $t^M_{r1} + \delta_M \notin [t_i, t_{i+1}[$, i.e., the intruder is not able to solve the puzzle in the validity interval i of the associated key.

- $t^M_{r1} + \delta_M \geq t^B_{r2}$, i.e., the intruder cannot solve the puzzle before Bob receives $MAC_{k_i}(m)$ and therefore cannot create a valid MAC for his own message.
- In order for the protocol to achieve instantaneous message authentication, we need $t^B_{r1} + \delta_B \leq t^B_{r3}$, i.e., Bob can solve the puzzle before he receives the message authenticated by the key in the puzzle.

The above analysis shows that secure broadcast authentication can be achieved if $t^M_{r1} + \delta_M \notin [t_i, t_{i+1}[$ and $t^M_{r1} + \delta_M \geq t^B_{r2}$ hold. Furthermore, the protocol achieves instantaneous message authentication if $t^B_{r1} + \delta_B \leq t^B_{r3}$ is fulfilled.
In terms of our definition of broadcast authentication, the protocol authenticates the message m, since no node other than the owner of the one-way chain knows k_i before t_{i+1} and therefore only the owner can create a valid MAC for the message. Similarly m is T-recent since it is verified by the corresponding MAC using the key of the time interval $i = [t_i, t_{i+1}[$. Therefore the protocol achieves T-recentness, where $T \leq |t^B_{r3} - t_i|$, in case of the successful authentication of the message.

This analysis also shows that the ability of the receiver to verify a given message depends on the assumption that the sender can estimate message propagation delays. If the sender can reach all the receivers through a direct link (e.g., a sender is a node with a high-power radio), then estimating propagation delays is not difficult, as it only depends on the link communication speed, which is predictable. However, if the sender broadcasts in a multi-hop network, the estimation of propagation delays will depend on network topology and is more challenging. As we show in Section 4, incorrect propagation delay estimates affect the performance (but not the security) of all protocols that are based on delayed key disclosure by resulting in valid messages being rejected by a subset of network nodes.

Note that the given analysis makes worst-case assumptions in terms of the attacker's abilities, i.e., the attacker receives each message instantaneously from the sender and similarly can deliver messages to the intended receiver without any delay. In reality, it will be more difficult to successfully attack the protocol. For example, if we assume that the puzzle and the MAC are concatenated together within the message, then the attacker has to jam the entire message, solve the puzzle, and create a valid MAC for his own message. Finally the new MAC has to be concatenated with the puzzle and must be delivered to the receiver before the validity interval of the key (inside the puzzle) ends. In a scenario where we have optimal conditions in terms of propagation delay (to the receivers), synchronized clocks, and predefined message-sizes, we can choose a sufficiently small validity interval for keys that minimizes the possibility of a successful attack.

3.2 BAP-2

BAP-2 is based on an approach similar to BAP-1 in that late key disclosure is achieved using cryptographic puzzles. The main difference is that, in BAP-2, not only the key, but also the message and its MAC is encapsulated within

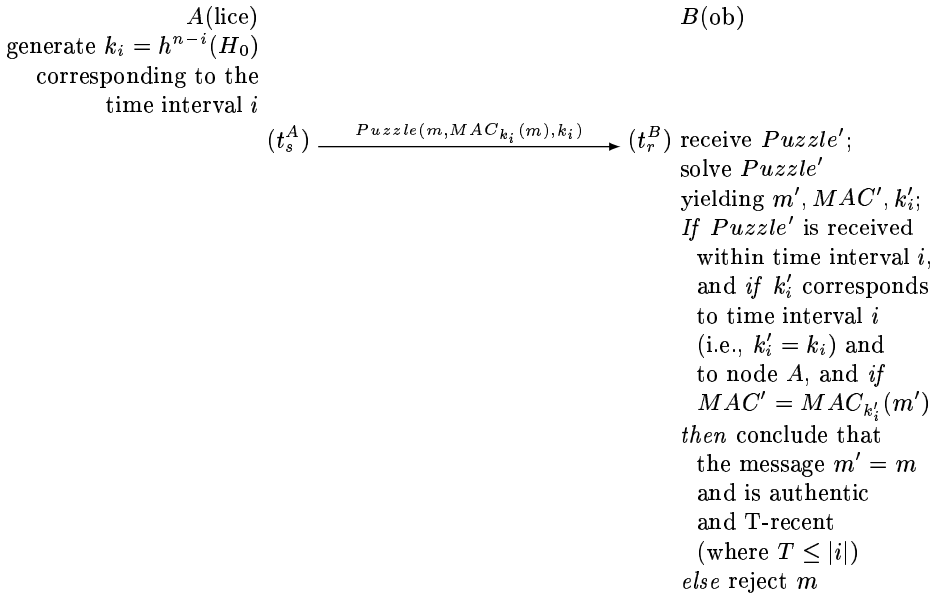


Fig. 4. BAP-2 protocol. The protocol achieves broadcast authentication through delayed key release based on cryptographic puzzles. Message authentication is achieved if the receiver receives the puzzle before the attacker has solved it. All messages received by B are marked with $'$ to denote that they might have been modified in transit by the adversary.

a puzzle. This collapses three messages into one and also reduces the time that the attacker has to solve the puzzle in order to break the scheme. The BAP-2 protocol is shown on Figure 4. In this protocol, the sender generates the key $k_i = h^{n-i}(H_0)$ for time interval i. Hence the sender encapsulates the message m, its message authentication code $MAC_{k_i}(m)$, and the key k_i in a puzzle $Puzzle(m, MAC_{k_i}(m), k_i)$. After receiving the puzzle $Puzzle'$, the receiver solves it and then verifies (i) that the $Puzzle'$ was received during the time interval i, (ii) that the key k_i' (derived from $Puzzle'$) is indeed authentic and that it corresponds to the current time slot i and to the claimed sender A, and (iii) that the message authentication code MAC' derived from the puzzle corresponds to $MAC_{k_i'}(m')$ computed with the derived key k_i' over the derived message m'. If and only if all three verifications succeed, the receiver concludes that the message $m' = m$ is both authentic (i.e., generated by the claimed source A) and T-recent (where $T \leq |i|$). Consequently, BAP-2 reaches T-authentication according to Definition 1, where $T \leq |i|$.

One advantage of BAP-2 over BAP-1 is that the attacker has less time to solve the puzzle. Namely, as soon as the first bits of the puzzle are received by the receiver, the attacker looses the possibility to forge the message. Therefore, the key validity time intervals can be shortened in BAP-2 with respect to the

intervals in BAP-1, assuming the same message size, key size, and propagation delays. One drawback of this solution is the loss of instantaneous message verification and the inability to prepare the puzzles beforehand (unless the messages are largely predictable or drawn from a small, well-defined set).

The security analysis of BAP-2 closely resembles that of BAP-1 and we therefore omit further details. Similar to BAP-1, we require that the attacker cannot generate a valid message prior to solving the puzzle and cannot solve the puzzle before the validity of the key expires.

3.3 Cryptographic Puzzles

In this section, we discuss possible realizations of the cryptographic puzzles used in our protocols. Cryptographic puzzles were first suggested by Merkle [9] and led to the invention of public-key cryptography. In [29], Rivest et al. present a construction of time-lock puzzles based on repeated squaring. The main contribution of time-lock puzzles is that they are non-parallelizable as solving them requires iterated application of an inherently sequential set of operations. However, solving this kind of puzzle requires the use of modular arithmetic and is therefore prohibitively expensive in networks composed of resource-constrained devices. Juels and Brainard [18] propose client puzzles based on one-way hash functions with partially disclosed hash input values. Their client puzzles use light-weight cryptographic primitives, but as they rely on exhaustive search, they are parallelizable. The main advantage of both time-lock and client puzzles is that they are simple to construct, but take significant time to solve. If the time required for puzzle construction is not prohibitive (e.g., can be performed during idle time), another puzzle construction scheme can be used. Namely, puzzles can be constructed by iterating a strong encryption function (such as AES [24]) a predefined number of times over a protected message, while either partially or entirely disclosing the used encryption keys. To solve this puzzle, a receiver needs to decrypt the ciphertext the same number of times as it was encrypted. Puzzles constructed based on iterated encryption are therefore non-parallelizable.

We now summarize desirable properties for puzzle schemes usable for broadcast authentication. (i) Puzzle generation should be computationally inexpensive for the sender. (ii) A puzzle should be solvable by a receiver within a given, finite time interval. In particular, a sender should be able, with the same puzzle scheme, to generate puzzles which a receiver can solve in short time as well as puzzles for which a receiver needs more time to solve. (iii) Solving a puzzle should not be parallelizable. This prevents a puzzle being solved faster by several colluding devices.

Although none of the described puzzle construction schemes satisfies all these requirements, both hash-based client puzzles and puzzles based on iterative encryption can be used for broadcast authentication in networks of resource scarce devices (e.g., sensor networks). Hash-based client puzzles neither incur heavy cost in terms of storage nor in terms of puzzle generation; these puzzles are therefore well suited for scenarios in which *any* sensor node is a potential broadcasting node. A drawback of client puzzles is that they are parallelizable and therefore

introduce a security risk in the broadcast authentication scheme if the attacker's strength (in terms of the number and type of devices that it holds) is underestimated. Puzzles based on iterated encryption functions are costly to generate, but are non-parallelizable; these puzzles can therefore be used in scenarios in which a subset of selected network nodes (i.e., network sinks, base stations, or cluster heads) broadcast messages to the network of resource-constrained (sensor) nodes. Given that these selected nodes have higher computational and storage capabilities than "regular" sensor nodes, they can compute and store puzzles for all the keys prior to their use in broadcast communication. These puzzles can therefore be computed for all keys in parallel with the creation of hash chains from which the keys are derived.

4 Analysis of Broadcast Authentication Protocols using delayed Key Disclosure

In this section, we analyze common properties shared by all broadcast authentication protocols based on one-way chains and delayed key disclosure. The common elements found in all these schemes are *validity windows* and a *disclosure schedule* for the keys, i.e., for the elements of the one-way chain (see Section 2.2 for further explanations). We will focus our analysis on the impact of clock synchronization and propagation delay on the security and performance of these protocols.
In the next subsection, we model the commonalities of this protocol class as a protocol pattern that can be found in any of these schemes. Therefore security properties related to this skeleton are relevant for this entire class of protocols. Examples of protocols in this class are the BAP-1 and BAP-2 protocols just presented, Cheung's authentication scheme [8], and TESLA, including all its variants [27].

4.1 The Protocol Pattern

As explained in Section 2.2, the key idea of using one-way chains for authentication is to associate elements of the chain in reverse order to time intervals. Therefore there are two important notions related to an element of the one-way chain: (i) the validity window of the element, i.e., the interval in which it should be used, and (ii) the point in time when the key is published, in order to prove the correctness of a MAC (built with a key derived from the element) and to prove the key's membership in the one-way chain.

In the protocol pattern below, we consider that the sender holds a one-way chain consisting of elements $(H_k)_{0 \leq k \leq n}$. To prove the origin of a message to a receiver, the sender appends the message authentication code $MAC(m_i, K_i)$ to a message m_i sent in time interval $[t_i, t_{i+1}[$, where K_i is derived from H_i. $[t_i, t_{i+1}[$ denotes the validity window of the element H_i, i.e., K_i is only accepted in this

interval as a key. Finally the chain element H_i is released (published) at $t_i + d$, i.e., d time units after the start of K_i's validity window.

Note that these parameters may vary in different implementations. For example, in TESLA [26] the chain element H_i is released two validity windows after it has been used to create a key for a MAC. Therefore $d_{TESLA} = 2 \cdot |t_{i+1} - t_i|$. In our protocols, we do not send the chain element after the message, but implement the delayed key disclosure by hiding the key in a puzzle. Therefore d_{BAP} is determined by the difficulty of the puzzle and the assumptions made about the attacker's computational power.

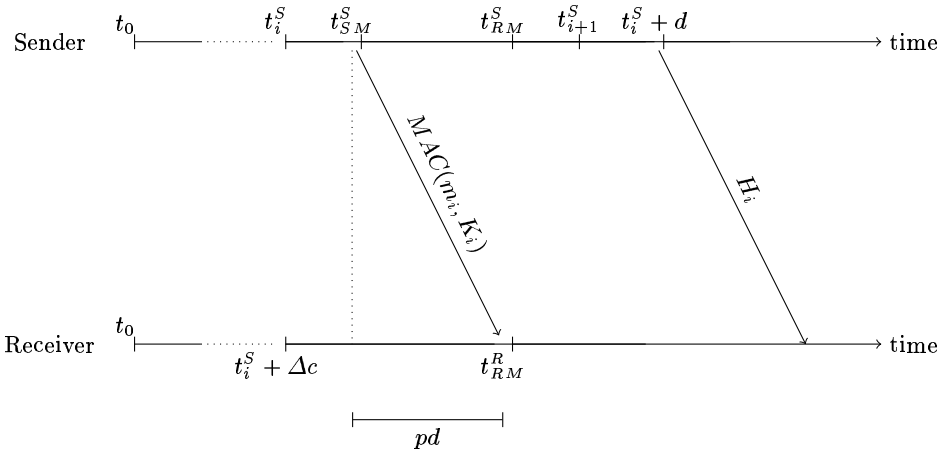


Fig. 5. Common Pattern for Delayed Key Disclosure Protocols

In Figure 4.1, the superscripts S (Sender) and R (Receiver) indicate the clock taken as a reference. Furthermore we make the following definitions and assumptions:

t_0: sender and receiver synchronize their clocks.
t_i^S, t_{i+1}^S: start and end of the validity interval for key K_i.
t_{SM}^S: $MAC(m_i, K_i)$ is sent.
t_{RM}^R: the receiver has received $MAC(m_i, K_i)$.
d: key disclosure delay, i.e. H_i is disclosed at $t_i^S + d$.
pd: the propagation delay of the message.
Δc: clock difference between sender and receiver.

Concerning the synchronization of the receiver's and sender's clock, we assume that at time t_i^S on the sender's clock, the receiver's clock shows $t_i^S + \Delta c$. For our observations, it is suffices to assume that Δc is constant since we consider here a relatively short time interval.

In order for the protocol to guarantee T-authentication, the following conditions must be fulfilled:

i) The MAC for message m_i is computed using the key K_i, which is valid in the interval when the puzzle and the MAC are sent.
ii) In order for the receiver to accept the message as being authentic, $MAC(m_i, K_i)$ (but not necessarily the message m_i) must be received within the validity interval of K_i.

From the sender's point of view, the receiver receives the message at $t^S_{RM} = t^S_{SM}+pd$. At this point in time, the receiver's clock shows $t^R_{RM} = t^S_{RM}+\Delta c+pd$. In order for the receiver to accept the message as valid, the inequality $t^R_{RM} < t^S_{i+1}$ must be fulfilled. Note at this point, that the boundaries (t_i and t_{i+1}) of the validity interval are given in the protocol specification and therefore $t^S_i = t_i$. Given $t^R_{RM} = t^S_{RM} + \Delta c + pd$, we have the condition $t^S_{RM} + \Delta c + pd < t_{i+1}$ for the receiver to accept the message as being sent by the holder of the one-way chain. Since $t^S_{RM} \geq t_i$, we conclude $t_i + \Delta c + pd < t_{i+1}$. Note that $t^S_{RM} = t_i$ is the earliest point in time when the key K_i would be used to authenticate a message. Therefore for the message to be successfully authenticated and therefore to be accepted by the receiver, we have the condition that:

$$\Delta c + pd < t_{i+1} - t_i \tag{1}$$

From the security standpoint, the time when the key is disclosed is particularly critical. Under the assumption of perfect cryptography, this is the only possibility for the attacker to acquire the key and therefore to create a valid message.
To analyze the performance and security properties of the schemes, we will now consider the possible values of the parameters involved, case by case.

4.2 Impact of the Accuracy of Time Synchronization

As we mentioned in the description, we assume that sender and receiver have synchronized their clocks at t_0. At t_i we assume the clocks to show a difference of Δc time units, due to clock drift. We therefore have the following two possibilities:

$\Delta c < 0$: The receiver's clock is slower than the sender's clock. By (1), even with a larger propagation delay pd, the message would still be accepted by the receiver. Therefore the receiver would accept messages authenticated by keys that are, according to the sender's clock, no longer valid. From the security point of view, the critical point is reached if $|\Delta c| > d - (t_{i+1} - t_i)$. This opens the possibility of an attack against the scheme if the intruder is able to jam the original message and waits (or solves the puzzle) for the key, which he could then use to create a valid MAC for his own message.

$\Delta c > 0$: The receiver's clock is faster than the sender's clock. In this case, the validity window on the client side shrinks and therefore no attacks are possible, except breaking the one-way chain (or the puzzle). On the other hand, a message could be invalidated by the receiver, although it would still be valid from the sender's perspective.

4.3 Impact of the Propagation Delay

In this section, we analyze the impact of propagation delay on the performance of such a scheme. For simplicity, in this analysis, we assume perfect synchronization (i.e., $\Delta c = 0$). Therefore the inequality reduces to $pd < t_{i+1} - t_i$, as a necessary condition for the receiver to accept messages. If he receives a message after a key is no longer valid, he will reject the message. This implies that, under the assumption of perfectly synchronized clocks, this class of protocols can only be used in an environment where the propagation delays of all nodes involved are known in advance, since the validity window and the disclosure delay have then to be chosen appropriately. In multi-hop networks, propagation delays are difficult to estimate in advance [13]. Since the disclosure delay has then to be set according to the maximum propagation delay, this would delay the time of verification drastically.

4.4 Differences to Schemes using Public Key Cryptography

As indicated in Section 2, the broadcast authentication problem is easily solved using asymmetric cryptography, but the computational cost is high. Hence, in order for a protocol to meet the requirements of resource-constraint devices, schemes using symmetric cryptography were introduced. But clearly, some form of asymmetry is needed in the solutions. The direct use of symmetric cryptography fails as the ability to verify the authenticity of a message is equivalent to the ability to authenticate an arbitrary message. One-way chains introduce the required asymmetry by letting the holder of the root element commit to a set of keys (the chain), while only publishing a single element (the last element of the chain). By using the elements of the chain as keys in keyed MAC functions, we transfer the asymmetry of knowing a certain key (chain element) to the ability to create a valid message authentication code. Since the key is needed to prove the validity of the MAC, the key must be disclosed. Upon disclosure, the asymmetry of the MAC is lost since, from this point on, anybody can create MACs using this key.

The asymmetry of the scheme is preserved by using a predefined disclosure schedule: if a MAC is created before the disclosure of the key, only the holder of the root of the one-way chain could have created it. Therefore we gain an additional property, namely evidence of when the message has been sent by the sender given by the time interval in which the key is intended to being used (before disclosure). This gives us *T-recentness*, where T is defined by the disclosure schedule and the arrival time of the message. These considerations also help illustrate the tradeoff in schemes using this technique. For efficiency

reasons, the key should be disclosed as early as possible for the receiver to be able to verify instantaneously the authenticity of the corresponding message. But for security reasons, the key should not be disclosed too early, so that an attacker cannot create a valid MAC by using a disclosed key.

5 Related work

Efficient broadcast (and multicast) authentication in wireless networks is an active field of research. In recent years, a number of proposals emerged that address this problem. To our knowledge, the first authentication scheme based on delayed key disclosure was introduced by Cheung [8] in the context of secure link state updates in routing protocols for wired networks. This technique was later used by Perrig et al. in [26], who propose TESLA, a broadcast authentication protocol based on delayed key disclosure. This approach relies on explicit key disclosure after the message (i.e., message and MAC). In [27], Perrig and Tygar present a suite of broadcast authentication protocols, including a Tesla variant named TIK, that achieves instant message verification with a single message release. A number of extensions of TESLA are presented in [20, 22, 21].

In TESLA and its variants, the time of message verification is delayed until the point in time when the key is disclosed by the message source. In comparison, our schemes achieve similar properties by sending the authentication key hidden in a puzzle with, or prior to, the message. Therefore our schemes reduce the number of messages needed to guarantee authenticity. Since our protocols are based on cryptographic puzzles, the security and also the performance of BAP-1 and BAP-2 depend on the computational power of the attacker and the honest nodes involved.

In the case of instantaneous message authentication, all schemes based on delayed key disclosure exhibit a trade-off between security and performance. As we have pointed out in the Section 4, the smaller the validity window of a key is chosen, the smaller the propagation must be. Since the message should be authenticated right after reception, the disclosure delay (and therefore also the validity window) of the key must be chosen small in order to guarantee authentication. As a consequence, the allowed propagation delay is equally small. Therefore TIK (the TESLA variant guaranteeing instantaneous message authentication) and BAP-1 suffer from the same restrictions on the allowed propagation delay. BAP-1 achieves broadcast authentication under the assumption that the puzzle has been solved up to the point where the message is received, whereas in TIK, the key bits need to be sent right after the message.

Besides the approaches based on delayed key disclosure, alternatives based on one-time signatures have been proposed in the context of broadcast authentication [27, 7]. Other approaches include broadcast authentication based on receiver proximity awareness [31]. Moreover, there are similar contributions in the field of multicast authentication that are more related to wired networks, e.g., [3], [14], [4].

Information-theoretically secure broadcast authentication mechanisms were proposed by a number of researchers [1], [10]. These protocols typically have a high overhead with many receivers and do not scale in large (sensor) networks. Canetti et al. present a broadcast authentication protocol in which a message is authenticated with k different MAC's [4] and in which no coalition of w (corrupted) receivers can forge a packet for a specific receiver.

Boneh, Durfee, and Franklin show in [2] that a compact collusion-resistant broadcast authentication protocol cannot be built without relying on digital signatures or on time synchronization.

6 Conclusion

In this paper, we have introduced two broadcast authentication protocols based on symmetric-key primitives and delayed key disclosure. We showed that these protocols achieve T-authentication, a form of authentication that includes a notion of recentness. By identifying the core components of broadcast authentication protocols using delayed key disclosure, we were able to uncover fundamental limitations of this class of protocols. Our results show that there is a trade-off between security and performance for protocols of this class. Furthermore, for performance and security reasons, the propagation delay inside a network must be known beforehand in order to choose the protocol parameters correctly. Since these delays are difficult to predict in multi-hop wireless networks, this class of protocols is not well suited for such networks. However, these protocols are well suited for wireless broadcasts where the receivers are in the direct range of the transmitter.

Acknowledgment The authors would like to thank Ueli Maurer for a useful discussion on cryptographic puzzles.

References

1. Shimshon Berkovits. How to broadcast a secret. In *EUROCRYPT*, 1991.
2. Dan Boneh, Glenn Durfee, and Matt Franklin. Lower bounds for multicast message authentication. *Lecture Notes in Computer Science*, 2001.
3. Ran Canetti, Pau-Chen Cheng, Frederique Giraud, Dimitrios Pendarakis, Josyula R. Rao, and Pankaj Rohatgi. An IPSec-based host architecture for secure internet multicast. In *Internet Society Symposium on Network and Distributed Systems Security*, 2000.
4. Ran Canetti, Juan Garay, Gene Itkis, Daniele Micciancio, Moni Naor, and Benny Pinkas. Multicast security: A taxonomy and some efficient constructions. In *INFOCOMM'99*, 1999.
5. Haowen Chan, Adrian Perrig, and Dawn Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Security and Privacy*, 2003.
6. Shang-Ming Chang, Shiuhpyng Shieh, Warren W. Lin, and Chih-Ming Hsieh. An efficient broadcast authentication scheme. In *ACM Symposium on Information, Computer and Communications Security*. ACM, 2006.

7. Shang-Ming Chang, Shiuhpyng Shieh, Warren W. Lin, and Chih-Ming Hsieh. An efficient broadcast authentication scheme in wireless sensor networks. In *ASIACCS '06: Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, 2006.
8. S. Cheung. An efficient message authentication scheme for link state routing. In *ACSAC*, 1997.
9. Ralph C.Merkle. Secure communications over insecure channels. *Communications of the ACM*, 1978.
10. Yvo Desmedt, Yair Frankel, and Moti Yung. Multi-receiver/multi-sender network security: efficient authenticated multicast/feedback. In *IEEE INFOCOM '92: Proceedings of the eleventh annual joint conference of the IEEE computer and communications societies on One world through communications*, 1992.
11. Wenliang Du, Ronghua Wang, and Peng Ning. An efficient scheme for authenticating public keys in sensor networks. In *MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, 2005.
12. Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*, 2002.
13. Saurabh Ganeriwal, Srdjan Capkun, Chih-Chieh Han, and Mani B. Srivastava. Secure time synchronization service for sensor networks. In *WiSe '05: Proceedings of the 4th ACM workshop on Wireless security*, 2005.
14. Rosario Gennaro and Pankaj Rohatgi. How to sign digital streams. *Lecture Notes in Computer Science*, 1997.
15. Ivan Getting. The Global Positioning System. *IEEE Spectrum*, December 1993.
16. Ralf Hauser, Antoni Przygienda, and Gene Tsudik. Reducing the cost of security in link state routing. In *Internet Society Symposium on Network and Distributed Systems Security*, 1997.
17. Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Ariadne:: a secure on-demand routing protocol for ad hoc networks. In *MobiCom '02: Proceedings of the 8th annual international conference on Mobile computing and networking*, 2002.
18. Ari Juels and John Brainard. Client puzzles: A cryptographic countermeasure against connection depletion attacks. *Proccedings of NDSS '99 (Network and Distributed Security Systems)*, 1999.
19. Leslie Lamport. Password authentication within insecure communication. *Communications of the ACM*, 1981.
20. Donggang Liu and Peng Ning. Efficient distribution of key chain commitments for broadcast authentication in distributed sensor networks. Technical report, Raleigh, NC, USA, 2002.
21. Donggang Liu and Peng Ning. Multilevel μ-tesla: Broadcast authentication for distributed sensor networks. *Trans. on Embedded Computing Sys.*, 3(4), 2004.
22. Donggang Liu, Peng Ning, Sencun Zhu, and Sushil Jajodia. Practical broadcast authentication in sensor networks. In *MOBIQUITOUS '05: Proceedings of the The Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, pages 118–132, Washington, DC, USA, 2005. IEEE Computer Society.
23. Michael Manzo, Tanya Roosta, and Shankar Sastry. Time synchronization attacks in sensor networks. In *SASN '05: Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks*, 2005.
24. National Institute of Stanards and Technology. Announcing the advanced encryption standard (aes). *Federal Information, Processing Standards Publication 197*, 2001.

25. Adrian Perrig. The biba one-time signature and broadcast authentication protocol. In *ACM Conference on Computer and Communications Security*, pages 28–37, 2001.
26. Adrian Perrig, Ran Canetti, Doug Tygar, and Dawn Song. The tesla broadcast authentication protocol. *RSA Cryptobytes*, 2002.
27. Adrian Perrig and Doug Tygar. *Secure Broadcast Communication in Wired and Wireless Networks*. Kluwer Academic Publishers, 2003.
28. Ronald L. Rivest, Adi Shamir, and Leonard M. Adelman. A method for obtaining digital signatures and public-key cryptosystems. Technical Report MIT/LCS/TM-82, 1977.
29. Ronald L. Rivest, Adi Shamir, and David A. Wagner. Time-lock puzzles and timed-release crypto. (MIT/LCS/TR-684):21, 1996.
30. Kun Sun, Peng Ning, and Cliff Wang;. Secure and resilient clock synchronization in wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 2006.
31. Mario Čagalj, Srdjan Čapkun, RamKumar Rengaswamy, Ilias Tsigkogiannis, M. Srivastava, and Jean-Pierre Hubaux. Integrity (I) codes: Message Integrity Protection and Authentication Over Insecure Channels. In *IEEE Symposium on Security and Privacy*, 2006.