Detecting and Preventing Beacon Replay Attacks in Receiver-Initiated MAC Protocols for Energy Efficient WSNs*

Alessio Di Mauro, Xenofon Fafoutis,
Sebastian Mödersheim, and Nicola Dragoni

Department of Applied Mathematics and Computer Science,
Technical University of Denmark, 2800 Kgs. Lyngby, Denmark
{adma,xefa,samo,ndra}@dtu.dk

Abstract. In receiver-initiated MAC protocols for Wireless Sensor Networks (WSNs), communication is initiated by the receiver of the data through beacons containing the receiver's identity. In this paper, we consider the case of a network intruder that captures and replays such beacons towards legitimate nodes, pretending to have a fake identity within the network. To prevent this attack we propose RAP, a challenge-response authentication protocol that is able to detect and prevent the beacon replay attack. The effectiveness of the protocol is formally verified using OFMC and ProVerif. Furthermore, we provide an analysis that highlights the trade-offs between the energy consumption and the level of security, defined as the resilience of the protocol to space exhaustion.

Keywords: Beacon Replay Attack, Receiver Initiated Medium Access Control, Wireless Sensor Network Security

1 Introduction

Wireless Sensor Networks (WSNs) are collections of many small, resource and power constrained, miniaturized sensing devices, equipped with an on-board radio transceiver which enables them to interconnect to each other. Their use covers a broad spectrum of applications, from temperature monitoring, to home automation and from medical to military applications. Deploying WSNs in unmanned, unsurveilled and hostile areas is not uncommon, making security a primary concern for the whole application. One of the very common attacks performed against WSNs is the so called *replay attack* [6], where a previously sent piece of information is recorded and re-transmitted at a later time. A replay attack is very commonly used as an essential building block for more complex and effective attacks (*Sinkhole* and *Blackhole* attacks [13], to mention only a few). Alongside security, research in the field of WSNs keeps on expanding in other interesting directions, primarily energy efficiency. The *Receiver Initiated*

* This work was partially supported by the IDEA4CPS project granted by the Danish National Research Foundation.

paradigm, introduced in [17], was proposed to provide an energy efficient way of establishing a link-layer connection. In Receiver-Initiated MAC (Medium Access Control)[1] protocols, the communication is started with a special frame called *beacon*, sent by what will be the receiver of the data. In this new scenario, the typical solutions used to address common security issues do not apply anymore.

The contribution of the paper is as follows; we define and introduce the *Beacon Replay Attack*, an attack specific for receiver-initiated MAC protocols for energy-efficient WSNs (Section 2). We analyze the attack in depth and show how it can be used to bring severe harm to a sensor network and how countering it at a the link-layer level will preclude other more sophisticated attacks. To achieve the latter, we introduce and discuss *RAP*, the Receiver Authentication Protocol, a challenge-response authentication protocol specifically designed to detect and prevent the beacon replay attack (Section 3). We also include a formal verification of RAP through the automated verification tools *OFMC* [3] and *ProVerif* [4] (Section 4.1) and a space exhaustion analysis (Section 4.2). Ultimately, we present an overhead assessment of RAP by means of an energy consumption analysis (Section 4.3). Section 5 concludes the paper.

2 Attack Definition and Related Work

2.1 Receiver-Initiated MAC Protocols

A MAC protocol is responsible for the establishment of a communication link. Its primary role is to coordinate access to and transmission over a medium common to several nodes. Furthermore, it plays a key role in the design of energy-efficient WSNs, as it controls the active and sleeping state of a node, known as *duty cycling*. The energy consumption of a wireless sensor node is dominated by the power needs of its radio component [2]. As a result, duty cycling the radio plays a fundamental role towards the realization of low-power wireless networks. Radio duty cycling introduces the problem of coordinating the sender and the receiver to a moment in time where both are active, so that a wireless link can be established. One of the common approaches to this issue is the *receiver-initiated paradigm* of communication for duty cycling nodes, which was originally introduced by Lin *et al.* in 2004 (RICER [17]). Later, in 2008, the paradigm was popularized by RI-MAC [29], whose authors also provided an implementation of the protocol for TinyOS [15].

Receiver-Initiated MAC protocols use beacons to establish a link between duty cycling nodes, as shown in Fig. 1. In particular, a node is generally in a sleeping state, in which its radio is turned off. Periodically, it interrupts its sleep to transmit a small frame, called *beacon*, which indicates its availability to receive data. After the beacon transmission and for a predefined time, the node awaits with the radio tuned on, for a reply. In case of no reply, the node goes back to the sleeping state. A node with data to transmit interrupts its sleep and passively listens to the channel for a beacon that originates from the intended receiver.

[1] Mind the unfortunate clash of acronym with Message Authentication Code.

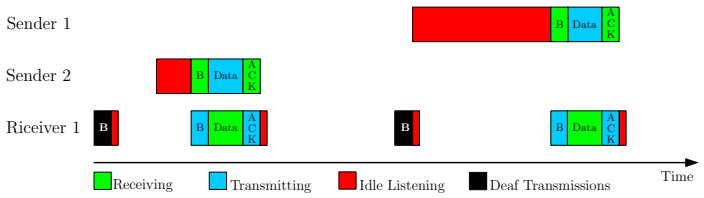


Fig. 1: Receiver-Initiated paradigm of communication.

Upon reception of a beacon, data transmission follows, typically acknowledged by an additional control frame (ACK). The latter concludes the communication cycle and both nodes go to the sleeping state.

Since the publication of RI-MAC, several MAC protocols that build on the receiver-initiated paradigm have been proposed. Such protocols mostly focus on optimizing the performance of the network and/or extending some features. For instance, proposed protocols focus on different aspects such as mitigating the time a node awaits for a beacon (e.g. EE-RI-MAC [35] and PW-MAC [31]), dynamically adapting the duty cycles (e.g. ODMAC [9] and CyMAC [24]), adding broadcasting support (ADB [28] and YA-MAC [34]) and adding multi-channel support (DCM [16] and EM-MAC [30]). Despite their differences, all these MAC protocols are based on the same receiver initiated communication paradigm.

2.2 Related Work: Mitigating Replay Attacks in WSNs

The replay attack is a well known threat for WSNs. It can be used as a building block for other attacks such as PDoS (Path Denial of Service) [5] where a whole path from one sensor node to the base station is filled with bogus packets. Given the typical structure of a WSN, i.e. a tree rooted in the base station, not only the node at one end of the attacked path can not use the communication medium, but also all the nodes *along* the path are prevented from forwarding their own messages. Furthermore, depending on the specific application that is being run on top of the network, replayed data messages could pose different kind of threats according to their specific meaning. One of the well known security suites for WSNs, TinySec [14], explicitly leaves replay attacks out of consideration.

Other previous works have addressed and mitigated replay attacks. The most common solution is to make each packet unique by means of adding either a counter or a timestamp. Timestamps are usually harder to implement because they require an agreement between the sender and the receiver which, in turns, translates to a global agreement for forwarded packets. An alternative is represented by monotonically increasing counters that are generally included within a message authentication code, making sure that each message will be different from the previous one. The authors in [25], use two different techniques one for each part of the protocol. In SNEP a counter is added within the MAC code, whereas time synchronization and hash chains are used in μTesla. Similarly, the authors in [18] use a sequence number in the message exchange. The work found

in [8] makes use of hash chains and a two step scheme composed of detection and response. For the detection part each node adds its own ID value to the message, along with an always increasing common hop count. The authors in [10] use the LEACH [11] protocol in a query driven paradigm and build upon it a mechanism that exploits the cluster organization, relaying on the cluster heads to compare timings of the messages from the registered nodes. Finally, [26] presents a time synchronization scheme that makes use of beacon messages that could somehow resemble the idea of beacons in the receiver-initiated paradigm. Once more the authors make use of a sequence number in order to prevent replay attacks.

Replaying beacons in the receiver-initiated world presents a very different approach to the typical replay attack. In the next section we will see why commonly adopted solutions are inapplicable or ineffective for this class of protocols.

2.3 Beacon Replay Attack in the Receiver-Initiated Paradigm

A replay attack is defined as an attack against a protocol where previously exchanged messages are reused in order to fool legitimate participants into thinking that the current run of the protocol is valid and exchanged data is fresh [6].

Replay attacks can be deployed against WSNs using a receiver-initiated MAC protocol. The key idea is to capture and replay *beacon* frames. As mentioned before, these frames manifest the availability of a particular node to receive a message. Among other things, beacons contain the identity of their creator which is the main piece of information needed to determine whether or not a specific beacon can be used by a potential sender, according to the overlying routing algorithm. By replaying beacons containing good identities (typically from a routing point of view), it is possible to deploy a series of other attacks.

First of all, it is possible to flood the channel with these frames, trying to accumulate as many data packets as possible, therefore performing what is known as a *Sinkhole attack* [13]. After the acquisition, packets can be completely dropped thus performing a *Blackhole attack* [13]. A subtler possibility is to implement a *Selective Forwarding attack* [13] (sometimes also called *Grayhole attack*), where the packets are not dropped indiscriminately, but rather according to their source. This yields a harder to detect and yet still very effective attack. Another possible attack is the *Sybil attack* [13] shown in Fig. 2, where a node relates to other nodes with more than one identity. This could lead to routing paths to be invalidated, or even nodes that are physically not within range one another, to be led to believe so; turning this into a rudimentary one-man *Wormhole attack* [13]. One last meta-attack, specific to duty-cycling wireless networks, is what we call the *Sleepwalker attack*. The idea behind this attack is that all the previous attacks can be deployed by a malicious node that is within range of the attacked node, by exploiting the notion of duty-cycle. Beacons can be collected from a node and replayed in the same neighborhood when the original sender is asleep. In this way a malicious node can effectively masquerade itself as another node.

Well-known techniques to prevent this attack (shortly introduced in Section 2.2) do not apply in this scenario. One of the advantages of a receiver-initiated approach is the fact that no synchronization is needed for the protocol to operate.

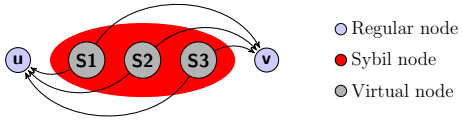


Fig. 2: Sybil attack: a Sybil node (red) sends beacons to regular nodes (u,v) claiming different legit identities (S1, S2, S3).

Timestamps, in order to be meaningful, require some form of clock synchronization among the nodes. This usually comes for free within protocols that use synchronized duty-cycles, but is a costly feature to obtain in receiver-initiated protocols. The other common alternative is the use of counters or session numbers. The latter are random non-reusable numbers that uniquely identify a particular message, or in this case a beacon. In order to check if a received beacon is fresh or replayed, a table of all the previously used session numbers should be kept. Given the highly constrained resources of a node, and the fact that there should be such a table for each one of the neighboring nodes, this solution is inapplicable. One way of simplifying this mechanism is to replace the random number with a monotonically increasing counter. This eliminates the need of having to store a whole table, only the latest value is needed. Upon receiving a message the new counter value can be compared against the last received one and if newer (i.e. the receiver value of the counter is bigger than the previous one) it will be accepted and discarded otherwise. The reason why this mechanism does not work with a receiver-initiated protocol is the following. Beacons are sent with a periodic cadence, which is typically randomized in order to minimize collisions. If we also consider all the neighboring nodes, from the point of view of a specific node, the arrival time of a beacon is virtually uniformly distributed. This means that there is no way for a sleeping node to know how many beacons were sent between the current and the previous active period, allowing the attacker to replay beacons that were not received by sleeping nodes. Moreover, a major downside of both timestamps and counters, is that some extra information (i.e. overhead) has to be sent with every beacon, even the ones that will never be received, because all the other nodes are asleep.

Lastly, despite the fact that Message Authentication Codes (MAC) can be used to authenticate beacon, they cannot prevent a replay attack. All that can be guaranteed upon receiving a beacon whose message authentication code correctly matches, is that the at some moment in time that beacon was genuine, created by a legitimate node and intended for another legitimate node. However, it is not possible to establish whether or not the beacon that has just been received is actually *that* beacon.

For all these reasons, we introduce *RAP*, a novel authentication scheme specifically designed to detect and prevent the beacon replay attack in receiver-initiated MAC protocols.

3 Receiver Authentication Protocol (RAP)

RAP (*Receiver Authentication Protocol*) is a challenge-response authentication protocol that aims to authenticate the receiver, i.e. the beacon transmitter, in a receiver-initiated data transmission, securing the receiver-initiated paradigm of communication in general. RAP is compatible and can be used on top of every MAC protocol that follows the receiver-initiated paradigm, essentially securing the whole class of protocols from beacon replay attacks; moreover it can and should be used together with security suites that provide other important features such as data integrity and confidentiality (e.g. TinySec [14]).

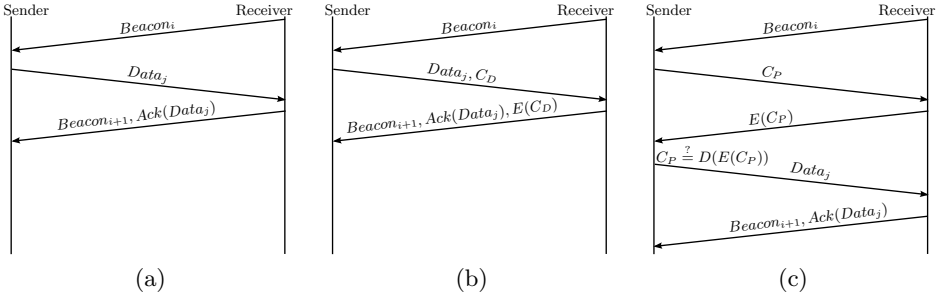


Fig. 3: A typical receiver-initiated protocol (a), RAP-D (b), RAP-P (c).

RAP has two modes of operation as shown in Fig. 3, namely *detection* and *prevention* mode. In a nutshell, the detection mode (RAP-D) is a low overhead scheme and aims at detecting an intruder that replays beacons without preventing it from doing so. The prevention mode (RAP-P), on the other hand, is a more costly scheme that prevents the attack altogether. As described in the following sections, the key difference between the two modes is the timing of the challenge-response message exchange. In RAP-P, the challenge-response message exchange takes place *before* the data transmission. Thus, the sender transmits the data packet only if the receiver is authenticated. The low overhead nature of RAP-D, on the other hand, is maintained by piggybacking the challenge and its response on top of the frames normally exchanged in the MAC protocol. In other words, the authentication of the receiver takes place *after* the data transmission (thus, the attack is not prevented). Having energy efficiency as a primary system priority, the idea is that a node normally operates at the low overhead detection mode and switches to the expensive prevention mode only if necessary.

3.1 Detection Mode (RAP-D)

RAP-D is aiming at detecting beacon replay attacks with low communication overhead. The protocol works as shown in Fig. 3b. Consider that a sender node

A wants to transmit some data to a receiver node B. After B broadcasts a beacon, A answers back with a data packet and a challenge value C_D. On its following beacon, B acknowledges the reception of the data packet, and attaches the encrypted version of the challenge $E_{k_{RAP}}(C_D)$ using the protocol specific, shared key k_{RAP}. At this point B can validate the response to the challenge by decrypting it and checking it against its original value. Should these two values not match, then B can conclude that the initial beacon was not genuine.

RAP-D adds a minimal overhead in the whole communication scheme, as the challenge and the response are piggybacked on top of a regular message exchange. Furthermore, if the challenge, C_D, is transmitted as part of the payload and encrypted with it, its size can be relatively small without risking increasing the chances of a space exhaustion attack (see Section 4.2).

3.2 Prevention Mode (RAP-P)

RAP-P is aiming to prevent the beacon replay attack at the cost of an increased overhead. In particular, the challenge-response messages are exchanged before the data transmission, in order to distinguish the legitimate from the replayed beacons. The protocol works as shown in Fig. 3c. Instead of sending the data right after a beacon, A sends out a longer challenge C_P, and awaits for its encrypted version $E_{k_{RAP}}(C_P)$ from B. Only if the received value decrypts correctly (i.e. matches against C_P), then data is sent. This scheme is more expensive because it requires two additional messages to be exchanged. Additionally, the size of the challenge needs to be significantly larger than the detection mode to prevent space exhaustion attacks.

3.3 Transition Policies

Depending on the security goal of an application, RAP can be configured to switch between the two modes, using several policies. If the application cannot tolerate a few beacons getting replayed, the protocol should always operate in prevention mode for maximum security. In the opposite case, the detection mode should be the default mode to promote energy efficiency. Here, the transition from RAP-D to RAP-P should be done after a defined number of challenge mismatches. This number should be configured accordingly to account for channel errors. Furthermore, the intruder detection may trigger an alarm that can be piggybacked onto data packets and beacons in order to warn the neighboring nodes and the sink. The transition back to detection mode can be done either automatically or manually depending on the level of desired of security. In cases of high security requirements, it may be desired that RAP-D is re-activated manually by the system administrator only after an investigation. An automatic transition to RAP-D, can be done after a predetermined number of successful challenge matches. To avoid the exploitation of the latter transition policy, this number can be exponentially increased each time a new replay attack is detected.

4 Verification and Analysis

4.1 Verification with OFMC and ProVerif

In order to formally verify RAP, we modeled it using the AnB language. AnB [21] is a specification language based on the popular Alice-and-Bob notation for security protocols. Besides giving us a way to describe the protocols of interest in a succinct way, AnB is also a formal language with an unambiguous semantics of the honest agents, the intruder, and the goals of the protocol. The semantics of AnB is defined by translation to infinite-state transition systems and its attack states, described in the AVISPA Intermediate Format [1]. The Intermediate Format can be directly read by several tools, such as the model-checker OFMC [3]. We also manually translate AnB specification to the abstraction-based tool ProVerif [4]. The main idea for using two tools lies in their complementary strengths. OFMC is effective in finding attacks, but can verify a protocol only for a bounded number sessions; on the other hand ProVerif abstracts from the concrete search space, sometimes producing false attacks (especially for replay-protection goals), requiring adaptations of the specification. Therefore, verifying the protocols with different approaches gives a higher confidence.

The core of the AnB specification is the definition of the behavior of each role of the protocol when it is played by an honest agent, namely how this agent decomposes the messages it receives (and what parts of a received message it can actually check), and how the agent composes outgoing messages based on its initial knowledge and the previously received messages. Here, all variables that do not appear in the knowledge section of the AnB specification are values that are *freshly* created by the agent who first uses them. For instance in the detection protocol RAP-D, A freshly creates the challenge C and the data $Data$. For the full details of the AnB semantics we refer to the original paper [21].

The standard intruder model of AnB is the common Dolev-Yao intruder [7] who controls the entire communication medium, it can arbitrarily overhear, send and even intercept messages. This is clearly inspired by communication in wired networks, and for many questions this is unrealistically strong for WSNs: an intruder may not control all locations spanned by the WSN and also it may not be able to hear a message when it is blocking it (e.g. by jamming). However, verifying the protocol under such a strong intruder gives higher confidence.

Moreover, unless explicitly excluded in the specification, the intruder can also play as a legal participant of the protocol. In the case of WSNs, this amounts to modeling compromised or intruder-controlled nodes. These dishonest nodes do not need to comply with the protocol, but can send whatever messages the intruder can compose from its knowledge. The initial intruder knowledge is determined also by the knowledge section of the AnB specification: for each instance of a role that the intruder is playing, he gets the associated initial knowledge. For example, consider in the RAP-D protocol a session where A is played by honest agent a and B is played by the intruder i. Then the intruder gets the knowledge of B under this instantiation, i.e., $a, i, mac, sk(a, i)$, and thus he has the shared key needed for communicating with a.

Furthermore, we use authentication goals which correspond to Lowe's injective agreement [19]. For the concrete example of the goal A *authenticates* B *on* B, C, as soon as B learns the fresh challenge C, it produces (in our model) an auxiliary event $witness(B, A, C)$ formalizing the intention to run the protocol with A and using challenge C. When A successfully finishes her run of the protocol, she produces also an event $request(A, B, C)$ to formalize that she finished the protocol, apparently with B and using challenge C. It counts as an attack if a trace contains more request events than corresponding witness events, i.e., when A either believes in receiving something from B that B actually has never sent, or if A is tricked into accepting something more times than B actually sent.

Finally, we use Maurer's channel notation [20], which is supported by the AnB language (for the formal definitions in AnB see [23]). Informally $A \bullet\!\!\rightarrow B$ means that A sends a message *authentically* to B (so B can be sure it really comes from A and was meant for B), $A \rightarrow\!\!\bullet B$ means that the message is sent confidentially (so A can be sure only B can receive it), and $A \bullet\!\!\rightarrow\!\!\bullet B$ means both authentic and confidential transmission. We use this notation to abstract from how the transmission of the actual data is organized, i.e., how authentication and confidentiality is achieved if they are desired. In fact, this problem is orthogonal to the replay-protection for the beacon that we study here, and the channel notation allows us to abstract from that. We note however that the actual realization of such channels (e.g. by MAC and/or encryption) needs to compose with our replay-protection, as explained in [23]. In short, if both our replay protection and the secure channel implementation use symmetric encryption with the same shared key, this can lead to misunderstandings in the WSN that may be exploitable. If they use however different keys (possibly derived from the same root key) this is prevented and the composition is sound.

In Fig. 4 it is possible to see how we modeled RAP using the AnB notation [21]. It should be noted that we decided to strip down the protocols in order to focus the attention on the beacon replay attack, hence we kept only the messages relevant in this sense. Furthermore, due to space limitation, we also decided not to include the basic version of the paradigm which does not include any form of authentication. This protocol is essentially modeled like the basic version (Fig. 4a) but without an authentication code for the beacon. This yields the trivial attack of beacon forgery due to the complete lack of authentication.

In the case of basic authentication (Fig. 4a), OFMC can detect the beacon replay attack, shown in Fig. 5, within a few seconds. For the intruder i it is simply enough to store a previously received beacon and replay it to a victim node in order to receive the data. Another interesting fact is that by adding the *weakly* clause to the authentication goal, hence turning it into Lowe's non-injective agreement [19], no attack is found. This helps to build confidence in the model and its correctness. When running OFMC on RAP-D and RAP-P we can verify them for 3 sessions in 2 and 24 minutes respectively, without any attack. Note that in each session, OFMC considers all possible instantiations of the roles with concrete agents, both honest and the intruder. Thus, whenever a protocol is verified for a given number of sessions, then there is no instantiation

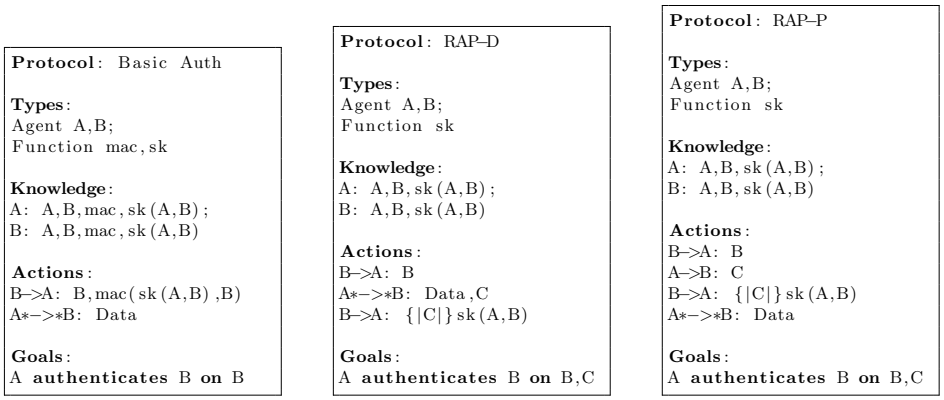


Fig. 4: The protocols used in OFMC described with AnB notation. A basic authentication model (a) is only enough to prevent beacon forgery. RAP-D (b) and RAP-P (c) are not affected by beacon replay attacks.

of the roles for these parallel sessions that can lead to an attack. As a rule of thumb, attacks are usually detected within 2 sessions.

ProVerif computes on first-order Horn clauses [12] that represent an over-approximation of the reachable events and messages the intruder can ever learn. There is therefore no notion of timeline, posing some difficulties for the analysis of replay, even though ProVerif offers the notion of *injective* events for this purpose. In order to experiment with different settings, we used the AIF framework [22] built on top of ProVerif, allowing to specify a state-transition system with a number of sets of data. In this particular case we can define for each agent the set of challenges that are sent out and have not been responded to, as well as those that have been responded to (and are therefore *used*). The AIF framework also allows for producing the Horn clauses for a different tool (on which ProVerif was originally based): the automatic first-order theorem prover SPASS [33]. It is therefore without extra cost to check the verification also with SPASS. ProVerif needs 5 and 3 minutes, respectively for RAP-D and RAP-P, while SPASS has a large discrepancy in run times: 73 minutes for RAP-D and only 1.5 minutes for RAP-P. In fact, the two tools have often different performance and termination behavior due to very different strategies, another reason to often try out both.

4.2 Space Exhaustion Analysis

In this section we conduct a space exhaustion analysis on RAP. Specifically, an attacker can passively monitor the communication of legitimate nodes and collect pairs of challenge and response messages. This way, the attacker can gradually build a dictionary that can be used to bypass RAP. The size of such a dictionary is a direct indication of the resilience of the protocol against space exhaustion.

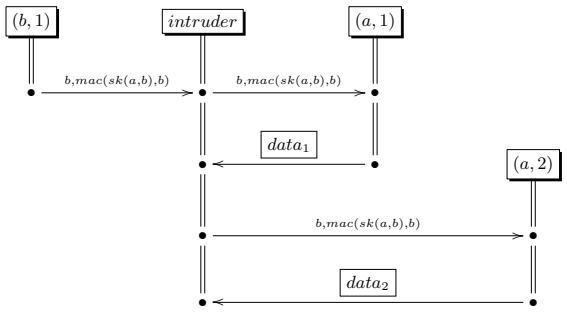


Fig. 5: Trace of the beacon replay attack found by OFMC in the basic version of a receiver-initiated protocol.

When RAP is in prevention mode, an attacker can trivially map the challenge to the respective response, as they are both distinct messages. Thus, the size of each word $D_{\text{RAP-P}}$ in the dictionary is equal to the size C_P of the challenge in bits, translating to $2^{D_{\text{RAP-P}}}$ words.

$$D_{\text{RAP-P}} = C_P \tag{1}$$

When RAP is in detection mode, we aim at a small challenge to keep the overhead low. However, the dictionary size can be significantly increased by encrypting the challenge together with the data, using Cipher-Block Chaining (CBC) encryption [27]. Essentially, CBC hides the challenge within the data, preventing the attacker from mapping the challenge to the response. As a result, a dictionary can only be built by mapping the whole message (that contains both the data and the challenge) to the respective response. Therefore, the size of each word $D_{\text{RAP-D}}$ in the dictionary, which translates to a dictionary size of $2^{D_{\text{RAP-D}}}$ words, is equal to the aggregate size L_D of the data and C_D of the challenge.

$$D_{\text{RAP-D}} = C_D + L_D \tag{2}$$

As an attacker can force the system to change the mode of operation, we note that the overall resilience of RAP to space exhaustion is equal to the smallest of the two dictionaries, $D_{\text{RAP-D}}$ and $D_{\text{RAP-P}}$. Furthermore, the sizes of the two challenges, C_D and C_P, which constitute configurable protocol parameters, define the level of security in the same manner the size of a key defines the level of security of an encryption algorithm. In the following section, we attempt to model the energy overhead of RAP and highlight the trade off between security and energy constraints.

4.3 Energy Consumption Analysis

Let L_D be the size of a data packet in bits, L_B be the size of a beacon in bits and R the transmission rate of the radio in bits per second. Additionally, let P_{tx} and

P_{rx} be power consumption for transmitting and receiving / listening respectively. First, we estimate the energy consumption for a single packet transmission in the case of not using RAP. For the receiver, B, the energy consumption is estimated by (3), where t_G is a time guard during which the radio is turned on while waiting for a answer right after a transmission. The purpose of such a guard is to account for the propagation and the processing delay.

$$E_B^{\text{Default}} = \frac{L_B}{R} P_{tx} + t_G P_{rx} + \frac{L_D}{R} P_{rx} + \frac{L_B}{R} P_{tx} \tag{3}$$

For the sender, A, the energy consumption is estimated similarly.

$$E_A^{\text{Default}} = \frac{L_B}{R} P_{rx} + \frac{L_D}{R} P_{tx} + t_G P_{rx} + \frac{L_B}{R} P_{rx} \tag{4}$$

Note that this energy model disregards the energy consumed while the sender awaits for the beacon, as this source of energy consumption is independent of the security protocol.

In the case of RAP-D, the energy consumption for a single packet transmission, for the receiver (B) and the sender (A), is given by the following formulae.

$$E_B^{\text{RAP-D}} = \frac{L_B}{R} P_{tx} + t_G P_{rx} + \frac{L_D + C_D}{R} P_{rx} + \frac{L_B + C_D}{R} P_{tx} \tag{5}$$

$$E_A^{\text{RAP-D}} = \frac{L_B}{R} P_{rx} + \frac{L_D + C_D}{R} P_{tx} + t_G P_{rx} + \frac{L_B + C_D}{R} P_{rx} \tag{6}$$

In the case of RAP-P, the energy consumption for a single packet transmission, for the receiver (B) and the sender (A), is estimated similarly.

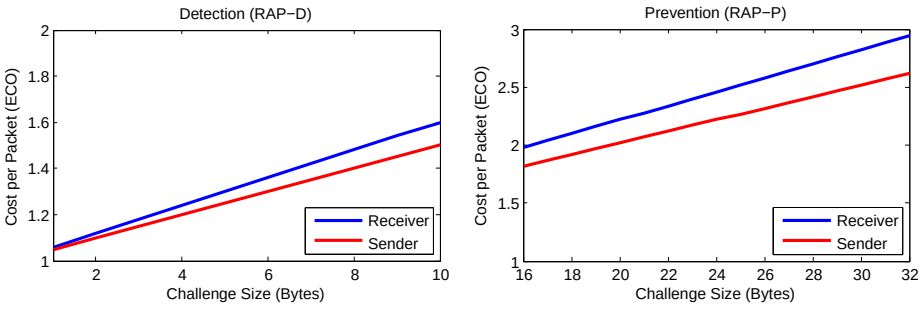
$$E_B^{\text{RAP-P}} = \frac{L_B}{R} P_{tx} + t_G P_{rx} + \frac{C_D}{R} P_{rx} + \frac{C_D}{R} P_{tx} + t_G P_{rx} + \frac{L_D}{R} P_{rx} + \frac{L_B}{R} P_{tx} \tag{7}$$

$$E_A^{\text{RAP-P}} = \frac{L_B}{R} P_{rx} + \frac{C_D}{R} P_{tx} + t_G P_{rx} + \frac{C_D}{R} P_{rx} + \frac{L_D}{R} P_{tx} + t_G P_{rx} + \frac{L_B}{R} P_{rx} \tag{8}$$

We define the energy consumption overhead (ECO) of a protocol as the ratio of the energy consumption for a single packet transmission (while using the respective protocol) over the case of a plain communication (without using it). The subscript j is equivalent to B for the receiver and A for the sender.

$$\text{ECO}_j^{\text{RAP-D}} = \frac{E_j^{\text{RAP-D}}}{E_j^{\text{Default}}} \ , \qquad \text{ECO}_j^{\text{RAP-P}} = \frac{E_j^{\text{RAP-P}}}{E_j^{\text{Default}}} \tag{9}$$

For the following numerical results, we assume using the CC2500 radio [32] which has the following characteristics: $R = 500\ Kbps$, $P_{tx} = 53.8\ mW$, $P_{rx} = 42.5\ mW$. Additionally, we consider the following values for the protocol parameters: $L_B = 2\ bytes$, $L_D = 32\ bytes$ and $t_G = 10\ \mu s$. Fig. 6 shows the cost for a single packet transmission of the two protocols, as defined in (9). Notice that



(a) (b)

Fig. 6: Energy Consumption Overhead (ECO) for a single packet transmission for RAP-D (a) and RAP-P (b).

the cost of the sender and the receiver increase linearly with the challenge size while the cost for the latter is relatively higher. The difference between them also increases as the challenge size increases.

In Fig. 7, we to compare the cost of RAP-D and RAP-P, showing the low-overhead nature of the former. Particularly, we compare the cost overhead ECO_B for the receiver of the two protocols keeping the same dictionary word size D, as defined in (1) and (2). Note that the dictionary word size indicates the resilience of each protocol to space exhaustion. In the case of RAP-D, we make sure the value of the challenge is at least 1 byte by setting it to $C_D = max(D_{\text{RAP-D}} - L_D, 1)$. As shown in the figure, the cost of using RAP-P is significantly higher than the cost of using RAP-D for the same level of security.

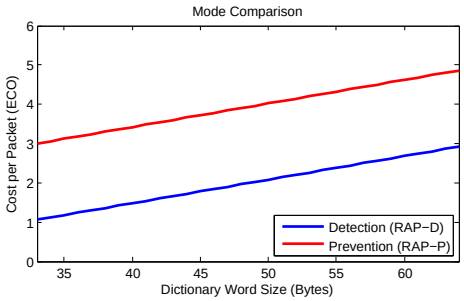


Fig. 7: The relative cost between RAP-D and RAP-P for the same level of resilience to space exhaustion.

Fig. 8 investigates the relative cost of the two protocols for different data sizes, by comparing the cost overhead ECO_B for the receiver of the two protocols. Additionally, we consider different dictionary word sizes as requirements

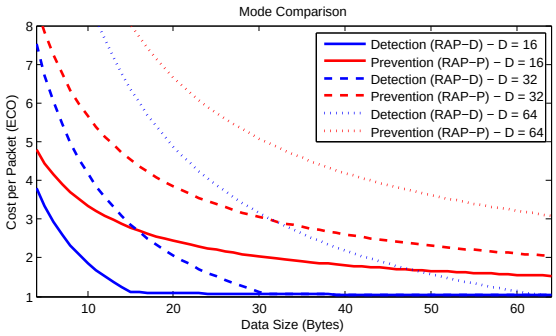


Fig. 8: The relative cost between RAP-D and RAP-P for different data sizes (L_D) and required levels of resilience to space exhaustion (D).

for resilience to space exhaustion. The results suggest that increasing the data packet drops the energy cost down for both protocols. The energy overhead of RAP-D can be kept at a minimal level as long as the data size is above the dictionary word size requirement.

5 Conclusion

In this paper, we focused on securing the class of receiver-initiated MAC protocols for WSN against the Beacon Replay attack. According to the receiver-initiated paradigm of communication, beacons are used to initiate the communication between two nodes. By collecting and replaying such beacons, an intruder can pretend a fake identity and perform a series of attacks. In particular, we proposed a challenge-response authentication protocol, named RAP, that is able to detect and prevent beacon replay attacks. RAP has two modes of operation. RAP-D is a low-overhead protocol that is able to detect intruders who replay beacons. RAP-P, on the other hand, is a more expensive prevention mechanism. We validated the effectiveness of RAP against beacon replay attacks using various tools, including OFMC and ProVerif. Furthermore, we have modeled the energy consumption of both protocols and we have exposed the trade-off between the level of security, measured by the resilience of the scheme to space exhaustion, and the level of energy consumption. Furthermore, we have shown that the energy consumption of RAP-P is significantly higher than RAP-D.

Our future work will be focused on two different directions. First, RAP can be extended to provide dynamically adaptable security. Specifically, a node can adapt the size of the challenge and, therefore, the resilience of the protocol to space exhaustion, according to the energy constraints of the node and the security requirements of the application. Such adaptability has particular interest in scenarios where the energy constraints are unpredictable, such as an energy harvesting scenario. The second direction is to extend RAP into a multi-key environment. While the size of the challenge can make a space-exhaustion attack

unfeasible, a periodical replacement of the encryption key can further increase the security of the system. It is, therefore, interesting to compare the energy overhead of increasing the size of the challenge to the respective cost of a key update and distribution mechanism and investigate the related trade-offs.

References

1. AVISPA: Deliverable 2.3: The Intermediate Format (2003), available at `http://www.avispa-project.org`
2. Bachir, A., Dohler, M., Watteyne, T., Leung, K.: MAC Essentials for Wireless Sensor Networks. IEEE Commun. Surveys Tutorials 12(2), 222–248 (2010)
3. Basin, D., Mödersheim, S., Viganò, L.: OFMC: A symbolic model checker for security protocols. Int. Journal of Information Security 4(3), 181–208 (2005)
4. Blanchet, B.: An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In: 14th IEEE Computer Security Foundations Workshop (CSFW-14). pp. 82–96. IEEE Computer Society, Cape Breton, Nova Scotia, Canada (Jun 2001)
5. Deng, J., Han, R., Mishra, S.: Limiting dos attacks during multihop data delivery in wireless sensor networks. Int. J. Secur. Netw. 1(3/4) (2006)
6. Denning, D.E., Sacco, G.M.: Timestamps in key distribution protocols. Commun. ACM 24(8), 533–536 (1981)
7. Dolev, D., Yao, A.: On the security of public key protocols. IEEE Trans. Inf. Theor. 29(2), 198–208 (2006)
8. Dong, J., Ackermann, K.E., Bavar, B., Nita-Rotaru, C.: Mitigating attacks against virtual coordinate based routing in wireless sensor networks. In: Proc. of the first ACM Conf. on Wireless network security. pp. 89–99. ACM (2008)
9. Fafoutis, X., Dragoni, N.: ODMAC: An On-Demand MAC Protocol for Energy Harvesting-Wireless Sensor Networks. In: Proc. 8th ACM Symp. on Performance Evaluation of Wireless Ad-Hoc, Sensor, and Ubiquitous Networks (PE-WASUN). pp. 49–56. ACM (2011)
10. Ghosal, A., Halder, S., Sur, S., Dan, A., DasBit, S.: Ensuring basic security and preventing replay attack in a query processing application domain in wsn. In: Proc. of the 2010 Int. Conf. on Computational Science and its Applications - Volume Part III. pp. 321–335. Springer-Verlag (2010)
11. Heinzelman, W., Chandrakasan, A., Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In: Proc. of the 33rd Annual Hawaii Int. Conf. on System Sciences. pp. 10 vol.2– (2000)
12. Horn, A.: On sentences which are true of direct unions of algebras. J. Symb. Log. pp. 14–21 (1951)
13. Karlof, C., Wagner, D.: Secure routing in wireless sensor networks: attacks and countermeasures. In: Proc. of the First IEEE Int. Workshop on Sensor Network Protocols and Applications. pp. 113–127 (2003)
14. Karlof, C., Sastry, N., Wagner, D.: Tinysec: a link layer security architecture for wireless sensor networks. In: Proc. 2nd ACM Int. Conf. on Embedded Networked Sensor Syst. (SenSys). pp. 162–175. ACM (2004)
15. Levis, P., Madden, S., Polastre, J., Szewczyk, R., Whitehouse, K., Woo, A., Gay, D., Hill, J., Welsh, M., Brewer, E., Culler, D.: TinyOS: An Operating System for Sensor Networks. In: Ambient Intelligence, pp. 115–148. Springer (2005)
16. Li, J., Zhang, D., Guo, L.: DCM: A Duty Cycle Based Multi-channel MAC Protocol for Wireless Sensor Networks. In: IET Int. Conf. on Wireless Sensor Network (IET-WSN). pp. 233–238 (2010)

17. Lin, E.Y.A., Rabaey, J.M., Wolisz, A.: Power-efficient rendez-vous schemes for dense wireless sensor networks. In: Proc. IEEE Int. Conf. on Communn. (ICC). vol. 7, pp. 3769–3776. IEEE (2004)
18. Liu, D., Ning, P.: Efficient distribution of key chain commitments for broadcast authentication in distributed sensor networks. Tech. rep. (2002)
19. Lowe, G.: A hierarchy of authentication specifications. In: CSFW'97, pp. 31–43. IEEE Computer Society Press (1997)
20. Maurer, U.M., Schmid, P.E.: A calculus for security bootstrapping in distributed systems. J. Comp. Sec. 4(1), 55–80 (1996)
21. Mödersheim, S.: Algebraic properties in alice and bob notation. In: Int. Conf. on Availability, Reliability and Security (ARES). pp. 433–440 (2009)
22. Mödersheim, S.: Abstraction by set-membership: verifying security protocols and web services with databases. In: ACM Conf. on Computer and Communications Security. pp. 351–360 (2010)
23. Mödersheim, S., Viganò, L.: Secure Pseudonymous Channels. In: Proc. of Esorics'09. pp. 337–354. LNCS 5789, Springer-Verlag (2009)
24. Peng, Y., Li, Z., Qiao, D., Zhang, W.: Delay-Bounded MAC with Minimal Idle Listening for Sensor Networks. In: Proc. 30th Ann. Joint Conf. IEEE Comput. and Communn. Soc. (INFOCOM). pp. 1314–1322. IEEE (2011)
25. Perrig, A., Szewczyk, R., Tygar, J.D., Wen, V., Culler, D.E.: Spins: security protocols for sensor networks. Wirel. Netw. 8(5), 521–534 (Sep 2002)
26. Song, H., Zhu, S., Cao, G.: Attack-resilient time synchronization for wireless sensor networks. In: Int. Conf. on Mobile Adhoc and Sensor Systems. pp. 8 pp.–772 (2005)
27. Stallings, W.: Cryptography and Network Security. Prentice Hall (2005)
28. Sun, Y., Gurewitz, O., Du, S., Tang, L., Johnson, D.B.: ADB: An Efficient Multihop Broadcast Protocol based on Asynchronous Duty-cycling in Wireless Sensor Networks. In: Proc. 7th ACM Int. Conf. on Embedded Networked Sensor Syst. (SenSys). pp. 43–56. ACM (2009)
29. Sun, Y., Gurewitz, O., Johnson, D.B.: RI-MAC: A Receiver-Initiated Asynchronous Duty Cycle MAC Protocol for Dynamic Traffic Loads in Wireless Sensor Networks. In: Proc. 6th ACM Int. Conf. on Embedded Networked Sensor Syst. (SenSys). pp. 1–14. ACM (2008)
30. Tang, L., Sun, Y., Gurewitz, O., Johnson, D.B.: EM-MAC: A Dynamic Multichannel Energy-Efficient MAC Protocol for Wireless Sensor Networks. In: Proc. of ACM MobiHoc'11. p. 23 (2011)
31. Tang, L., Sun, Y., Gurewitz, O., Johnson, D.B.: PW-MAC: An Energy-Efficient Predictive-Wakeup MAC Protocol for Wireless Sensor Networks. In: Proc. of INFOCOM'11. pp. 1305–1313. IEEE (2011)
32. Texas Instruments: CC250: Low-cost low-power 2.4 ghz rf transceiver (2011), available at `http://www.ti.com/lit/ds/symlink/cc2500.pdf`
33. Weidenbach, C., Schmidt, R.A., Hillenbrand, T., Rusev, R., Topic, D.: System description: Spass version 3.0. In: CADE. pp. 514–520 (2007)
34. Yadav, P., McCann, J.A.: YA-MAC: Handling Unified Unicast and Broadcast Traffic in Multi-hop Wireless Sensor Networks. In: Proc. 7th IEEE Int. Conf. on Distributed Computing in Sensor Systems (DCOSS). pp. 1–9. IEEE (2011)
35. Yong, Y.T., Chow, C.O., Kanesan, J., Ishii, H.: EE-RI-MAC: An energy-efficient receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks. Journal of Physical Sciences 6(11), 2633–2643 (2011)