RFID Distance Bounding Multistate Enhancement

Gildas Avoine[1], Christian Floerkemeier[2], and Benjamin Martin[1]

[1] Université catholique de Louvain
Information Security Group
B-1348 Louvain-la-Neuve, Belgium
[2] Massachusetts Institute of Technology
Auto-ID Labs
Cambridge, MA 02139, USA

Abstract. Distance bounding protocols aim at avoiding relay attacks during an authentication process. Such protocols are especially important in RFID, where mounting a relay attack between a low-capability prover and a remote verifier is a realistic threat. Several distance bounding protocols suitable for RFID have been recently suggested, all of which aim to reduce the adversary's success probability and the number of rounds executed within the protocol. Peinado et al. introduced an efficient distance bounding protocol that uses the concept of void challenges. We present in this paper a generic technique called *MUltiState Enhancement* that is based on a more efficient use of void challenges. MUSE significantly improves the performances of the already-published distance bounding protocols and extends the void challenges to p-symbols.

Keywords: RFID, Authentication, Mafia fraud, Distance bounding.

1 Introduction

Radio Frequency IDentification (RFID) is a well-known technology that is used to identify or authenticate objects or subjects wirelessly. RFID systems consist of transponders called *tags*, and transceivers called *readers*. The proliferation of RFID technology during the last decades results from the decreasing cost and size of the tags and the increased volume in which they are deployed. Most of RFID systems deployed today are *passive*, meaning that the RFID tags do not carry a battery and harvest the power of the carrier wave generated by the RFID reader.

The capabilities of the tags are restricted and application-dependent. For example, a 10-cent tag only transmits a short unique identifier upon reception of a reader's request, while a 2-euro tag such as those used in electronic passports has an embedded microprocessor. The latter is able to perform cryptographic operations within a reasonable time period. Those in current electronic passports are typically able to compute RSA-1024 signatures. RFID tags are often also used in applications such as mass transportation, building access control, and event ticketing. In such applications, the computation capabilities of the tag rely on

B. Roy and N. Sendrier (Eds.): INDOCRYPT 2009, LNCS 5922, pp. 290–307, 2009.

wired logic only. Computations are thus restricted but, nevertheless, the tags allow for on-the-fly encryption of a few thousand bits. Strong security can thus also be achieved with tags that offer mid-range computational resources.

While identification is the primary purpose of RFID, authentication via an RFID tag is an important application. The common RFID-friendly authentication protocols implemented in practice are usually based on the ISO/IEC 9798 standard. Although these protocols are secure in a classical cryptographic model, they are susceptible to *Mafia fraud* [4]. This attack, presented by Desmedt, Goutier, and Bengio at Crypto 1987, actually defeats any authentication protocol because the adversary passes the authentication by relaying the messages between the legitimate verifier (in our case the reader) and the legitimate prover (in our case the tag). Mafia fraud is a major security issue for RFID systems, precisely because the tags answer to any query from a reader without the consent or awareness of their tag owner.

As illustrated by Avoine and Tchamkerten in [1], Mafia fraud may have a real impact in our daily lives. To illustrate the problem, the authors considered an RFID-based theater ticketing system. To buy a ticket, the customer needs to stay in the field of the ticket machine during the transaction. The presence of the customer in the vicinity of the machine is an implicit agreement of the transaction. Now, let's assume there is a line of customers waiting for a ticket. Bob and Charlie are the adversaries: Bob is at the end of the queue, close to the victim Alice, and Charlie is in front of the ticketing machine. When the machine initiates the transaction with Charlie, the latter transmits the received signal to Bob who transmits it in turn to Alice. Alice's card automatically replies to Bob and the signal is sent from Bob to the machine through Charlie. The transaction is thus transparently relayed between Alice and the machine.

Mafia fraud is not caught by the classical cryptographic models because it comprises a relay of the low-layer signal without any attempt to tamper with the carried information. Thwarting relay attacks can thus not only rely on cryptographic measures, but requires the evaluation of the distance between the prover and the verifier. This must be done without significantly increasing the required capabilities of the RFID tags, which eliminates computationally or resource intensive approaches, such as the use of global positioning systems. To decide whether the prover is in the neighborhood of the verifier, a common approach consists for the latter to measure the round trip time (RTT) of a message transmitted from the verifier to the prover, and then back from the prover to the verifier. Assuming that the signal propagation speed is known, the prover can define $\Delta t_{\max}$ that is the maximum expected RTT including propagation and processing delays. An RTT less than $\Delta t_{\max}$ demonstrates that the prover necessarily stays in the verifier's neighborhood.

The work presented in this paper is based on the 3-state approach introduced by Munilla, Ortiz, and Peinado [8,9]. Our contribution is three-fold. We show that, based on the assumptions provided in [8,9], the 3-state approach can be significantly improved, i.e., the number of rounds in the protocol can be reduced while maintaining the same security level. We then generalize the 3-state approach and introduce the generic concept of *multistate* that improves all existing distance bounding protocols. We demonstrate the effectiveness of our solution by applying it to some well-known protocols.

In Section 2, we introduce some background related to distance bounding, especially the protocols designed by Hancke and Kuhn [5] on one hand, and by Munilla and Peinado [9] on the other hand. In Section 3 our 3-state enhancement is presented, followed in Section 4 by its generalization to the p-state case. We show in Section 5 that the case $p = 4$ provides a fair trade-off between security and practicability, and analyze it when it is applied to the most common distance bounding protocols.

2 Primer on Distance Bounding

In this section, some background about RTT-based distance bounding protocols are provided. We present Hancke and Kuhn's protocol (HK) [5], then Munilla and Peinado's protocol (MP) [8,9] is detailed. The latter improves HK using the concept of *void challenges*.

As stated in [4], the measurement of the RTT should not be noised by arbitrary processing delays, including delays due to cryptographic operations. It is suggested in [4] that (a) $\Delta t_{\max}$ is computed from the speed of light, (b) each message used for the RTT measurement contains only one bit, and (c) there are no other computation processed during the measurement of the round trip time. These assumptions still apply today and are the foundations of all the published distance bounding protocols for RFID [1,2,3,5,6,7,10,11,12,13]. The protocol originally suggested by Munilla, Ortiz, and Peinado [8,9] is an exception in that it considers messages carrying three states: 0, 1, or *void*.

2.1 Hancke and Kuhn's Protocol

The HK protocol [5], depicted in Figure 1, is a key-reference protocol in terms of distance bounding devoted to RFID systems. HK is a simple and fast protocol, but it suffers from a high adversary success probability.

Initialization. The prover (P) and the verifier (V) share a secret x and agree on (a) a security parameter n, (b) a public pseudo random function H whose output size is $2n$, and (c) a given timing bound $\Delta t_{\max}$.

Protocol. HK consists of two phases: a slow one followed by a fast one. During the slow phase V generates a random nonce N_V and sends it to P. Reciprocally, P generates N_P and sends it to V. V and P then both compute $H^{2n} := H(x, N_P, N_V)$. In what follows, H_i ($1 \leq i \leq 2n$) denotes the ith bit of H^{2n}, and $H_i \dots H_j$ ($1 \leq i < j \leq 2n$) denotes the concatenation of the bits from H_i to H_j. Then V and P split H^{2n} into two registers of length n: $R^0 := H_1 \dots H_n$ and $R^1 := H_{n+1} \dots H_{2n}$. The fast phase then consists of n rounds. In each of the rounds, V picks a random bit c_i (the challenge) and sends it to P. The latter immediately answers $r_i := R_i^{c_i}$ that is the ith bit of the register R^{c_i}.

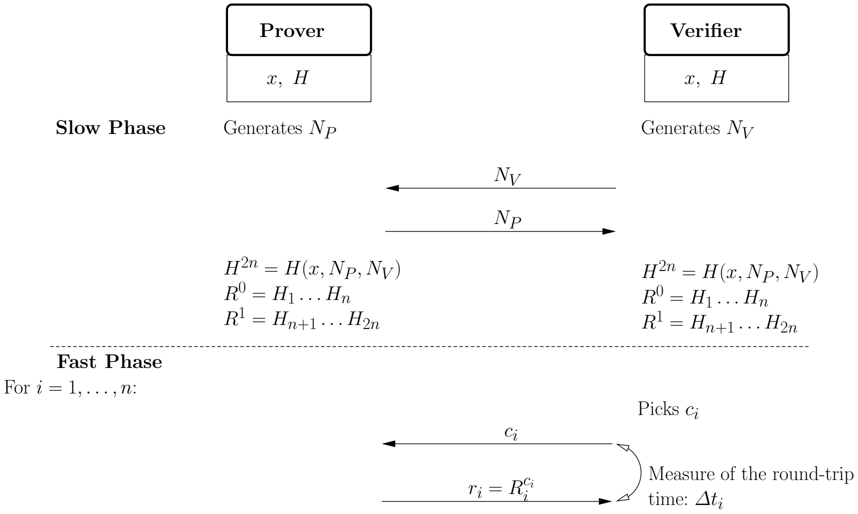


Fig. 1. Hancke and Kuhn's protocol

Verification. At the end of the fast phase, the verifier checks that the answers received from the prover are correct and that: $\forall\, i, 1 \le i \le n$, $\Delta t_i \le \Delta t_{\max}$..

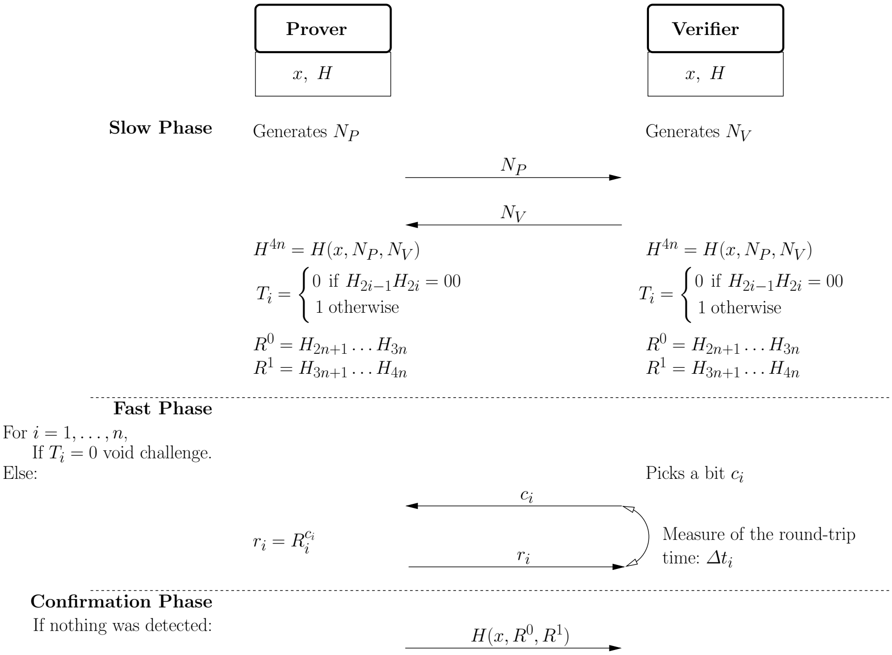
Computation of the adversary success probability. The best known attack is based on querying the tag with n 1-bit challenges between the slow and fast phases in order to obtain a full register. Without loss of generality, we can assume that the adversary obtains R^0 sending only 1-bit challenges equal to zero. Afterwards, when she tries to trick the reader, two cases occurs: (a) if $c_i = 0$ she definitely knows the right answer, (b) if $c_i = 1$, she has no clue about the right answer but she can try to guess it with probability $\frac{1}{2}$. Thereby, the adversary success probability, as explained in [5], is:

$$P_{\mathrm{HK}} = \left(\frac{3}{4}\right)^n . \tag{1}$$

2.2 Munilla and Peinado's Protocol

In order to decrease the adversary success probability of HK, Munilla and Peinado [8,9] introduce the concept of void challenges. The basic idea is that challenges can be 0, 1, or *void*, where *void* means that no challenge is sent. Prover and verifier agree on which challenges should be *void*. Upon reception of 0 or 1 while a *void* challenge was expected, the prover detects the attack and gives up the protocol. Figure 2 describes MP.

Initialization. The prover (P) and the verifier (V) share a secret x and agree on (a) a security parameter n, (b) a public pseudo random function H whose output size is $4n$, and (c) a given timing bound $\Delta t_{\max}$.



Prover
$x,\ H$
Verifier
$x,\ H$
Slow Phase
Generates N_P
Generates N_V
N_P
N_V
$H^{4n} = H(x, N_P, N_V)$
$T_i = \begin{cases} 0 \text{ if } H_{2i-1}H_{2i} = 00 \\ 1 \text{ otherwise} \end{cases}$
$R^0 = H_{2n+1} \dots H_{3n}$
$R^1 = H_{3n+1} \dots H_{4n}$
Fast Phase
For $i = 1, \dots, n$,
If $T_i = 0$ void challenge.
Else:
Picks a bit c_i
c_i
$r_i = R_i^{c_i}$
r_i
Measure of the round-trip time: Δt_i
Confirmation Phase
If nothing was detected:
$H(x, R^0, R^1)$

Fig. 2. Munilla and Peinado's protocol

Protocol. As with HK, V and P exchange nonces N_V and N_P. From these values, they compute $H^{4n} = H(x, N_P, N_V)$. $2n$ bits are used to generate a n-bit register T as follows: if $H_{2i-1}H_{2i} = 00$, 01, or 10 then $T_i = 1$, otherwise $T_i = 0$. Each T_i decides whether c_i is a void challenge ($T_i = 0$) or not ($T_i = 1$). In the latter case, c_i will be either 0 or 1, and will be called a *full* challenge. The $2n$ remaining bits are used to generate the two registers $R^0 = H_{2n+1} \dots H_{3n}$ and $R^1 = H_{3n+1} \dots H_{4n}$ as done by HK. Upon termination of the fast phase, if the prover did not detect any attack, he sends $H(x, R^0, R^1)$ to the verifier.

Verification. The verifier checks that the received $H(x, R^0, R^1)$ is correct, i.e, the prover did not detect an attack. Then the verifier checks the Δt_is and all the values r_is as it is done in HK.

Computation of the adversary success probability. In what follows, p_f denotes the probability that $T_i = 1$ ($1 \leq i \leq n$), i.e., a full challenge is expected in the ith round of the protocol. The adversary success probability clearly depends on p_f. It is shown in [9] that the optimal adversary success probability is obtained when $p_f = \frac{3}{5}$. However, obtaining such a probability is not trivial because T is obtained from the output of the random function H. Consequently, Munilla and Peinado suggest to take $p_f = \frac{3}{4}$, which is close to $\frac{3}{5}$ and easier to generate from H.

To compute the success probability of an adversary, one must know that an adversary may consider two strategies. The first strategy relies on the adversary querying the prover before the fast phase starts. In the second one, the adversary does not query the prover at all. In HK, it was clear that the best adversary's strategy was to query the prover in advance. In MP, the problem is more difficult because the prover aborts the protocol when he receives a challenge while expecting a void one.

In the case where the adversary does not ask in advance, the adversary succeeds if no void challenge appears and if he guesses the challenge. The probability is equal to $p_{\text{ask}} = \left(p_f \cdot \frac{3}{4}\right)^n$.

On the other side, without asking the tag in advance, $p_{noask} = \sum_{i=0}^{i=n} p(i) \cdot \left(\frac{1}{2}\right)^i$, where $p(i)$ is the probability that exactly i full challenge appears. This latter is equal to $p(i) = \binom{n}{i} \cdot p_f^i \cdot (1-p_f)^{(n-i)}$. At last $p_{noask} = \left(1 - \frac{p_f}{2}\right)^n$. When $p_f = \frac{3}{4}$ the adversary chooses the no-asking strategy and his success probability is:

$$P_{\text{MP}} = \left(\frac{5}{8}\right)^n. \tag{2}$$

3 MUSE-3 HK

MP is designed such that the prover always sends a void answer upon reception of a void challenge. We prove below that this approach does not exploit the full potential of the 3-state message approach. We also introduce an improvement which decreases the adversary success probability while the number of exchanged messages remains unchanged. This new protocol, called MUSE-3 HK, is an improvement over the 3-state message approach of HK. Throughout this paper, given a protocol $\mathcal{P}$, we denote by MUSE-p $\mathcal{P}$, the enhancement of $\mathcal{P}$ with p-state messages, where MUSE stands for MUltiState Enhancement.

In MUSE-3 HK, the initialization and verification steps do not differ from HK. We restrict our description of the core step which differs from the HK approach.

Protocol. Our enhancement is just like in MP [9] based on the introduction of 3-state messages rather than binary messages. However, in our approach the three states are not used in the same way. 0,1 and the void state are no longer treated differently and we simply refer to them as a 3-*symbol*. We denote these 3-symbols by $\{0, 1, 2\}$. Throughout this paper, given a protocol MUSE-p $\mathcal{P}$, we refer to a p-*symbol* as one of the p different p-states. We denote, by $\{0, \ldots, p-1\}$, the set of these p-symbols.

As with HK and MP, V and P exchange nonces N_V and N_P. From these values, they compute a bit string $H(x, N_P, N_V)$. We will discuss the length of $H(x, N_P, N_V)$ in a following section. With this bit string $H(x, N_P, N_V)$, they generate $3n$ 3-symbols $\{S_1, \ldots, S_{3n}\}$. These 3-symbols are used to fill up three registers R^0, R^1 and R^2. In fact, each one of the three registers R^j contains n 3-symbols $\{S_{j+1}, \ldots, S_{j+n}\}$.

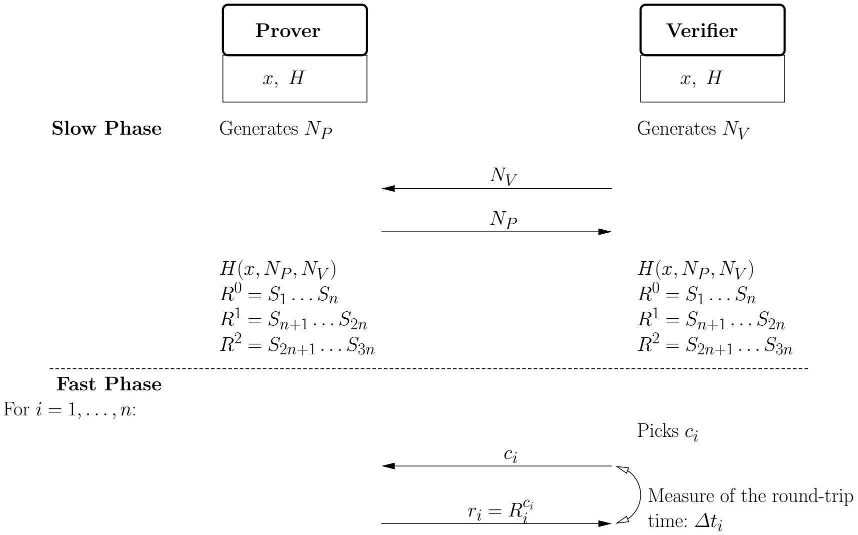


Fig. 3. Hancke and Kuhn's protocol with MUSE-3

After the fast phase begins, the verifier picks at random c_i from $\{0, 1, 2\}$ and sends it to the prover. The prover immediately answers $r_i = R_i^{c_i}$. Figure 3 illustrates this protocol.

Computation of the adversary success probability. In MUSE-3 HK, the prover is not able to abort the protocol because he has no means to detect an attack. Consequently, the success probability of the adversary is always higher when she queries the prover before the fast phase. In such a case, the adversary obtains one of the three registers, say R^0 without loss of generality. During the fast phase, when the adversary is challenged in a given round by the verifier with the challenge 0, she can definitely provide the right response; otherwise she answers randomly. Thereby, her success probability is $\frac{1}{3} \cdot 1 + \frac{2}{3} \cdot \frac{1}{3} = \frac{5}{9}$, for each round, and the overall probability is so:

$$P_{\text{MUSE-3 HK}} = \left(\frac{5}{9}\right)^n. \tag{3}$$

From equations (2) and (3), we deduce that MUSE-3 HK performs better than MP: it provides a smaller adversary success probability with neither increasing the number of exchanges, nor adding new assumptions compared to [9]. This behavior can be explained by the fact that in MP the adversary earns some information with the use of void challenges. This leak of information is no longer an issue with MUSE-3, because the three different 3-states are used in an identical way. One may nevertheless stress that MUSE-3 HK needs more memory than MP since 3 registers are required in MUSE-3 HK while 2 registers are enough in MP. In the next section, we generalize the three-state enhancement to the p-state enhancement and analyze both success probability and memory consumption.

4 MUSE-p HK

4.1 Hancke and Kuhn's Protocol with MUSE-p

In this section, we consider MUSE-p HK, where $p \geq 2$. The new protocol, which generalizes MUSE-3 HK, is similar to it except that it uses:

- p-symbols from $\{0, 1, 2, ..., p-1\}$,
- p registers containing n p-symbols,
- challenges and answers are p-symbols.

There is a perfect match between MUSE-2 HK and HK, when $p = 2$, .

4.2 Computation of the Adversary Success Probability

In the case of HK, two different attacks strategies exist: to either query or not query the tag in advance. If the adversary chooses to not ask in advance, at each round she tries a response at random, so her success probability is $\left(\frac{1}{p}\right)^n$. Therefore, the best strategy for the adversary to perform a Mafia fraud when HK is used consists in querying the prover with some arbitrary bits before the fast phase starts. In MUSE-p HK, the same strategy is used, i.e., the adversary queries the prover with some random p-symbols obtaining thus one register of size n of a total of p registers. Without loss of generality, we assume that the adversary obtains R^0. When the adversary is challenged with 0, she is definitely able to provide the right answer, otherwise she correctly answers with probability $1/p$. We deduce the overall success probability of the adversary (depicted in Figure 4) as being:

$$P_{\text{MUSE-}p\text{ HK}} = \left(\frac{2p-1}{p^2}\right)^n . \tag{4}$$

4.3 Generation of the Registers

We assume that the prover – in our framework, an RFID tag – is not able to directly generate and store p-symbols. Consequently, he must generate the symbols using a hash function that outputs bits only. The same problem occurs for the storage: the memory needed to store one p-symbol is $\lceil \log_2(p) \rceil$ bits. In other words, real p-symbols exist in practice only during their transit on the channel.

In order to generate the p-symbols, an arbitrary set $\mathcal{A} \subseteq \mathbb{F}_{2^{\lceil \log_2(p) \rceil}}$ is defined, such that the cardinality of $\mathcal{A}$ is equal to p i.e., $\mathcal{A}$ consists of p combinations of $\lceil \log_2(p) \rceil$ bits. Then the prover uses the following deterministic technique: firstly, he defines a bijection between the set $\{0, \ldots, p-1\}$ of p-symbols and $\mathcal{A}$. Secondly, he generates a stream of bits and regroups them by blocks of $\lceil \log_2(p) \rceil$ bits. A block belonging to $\mathcal{A}$ supplies one p-symbol; otherwise it is dropped.

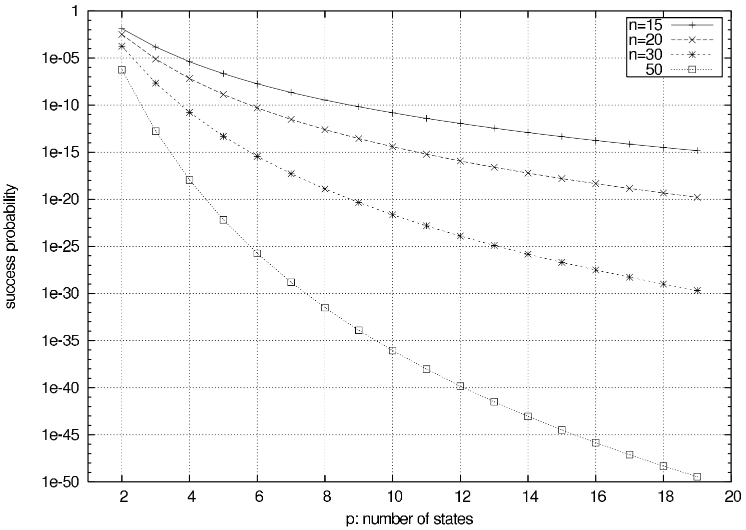


Fig. 4. Success probability for the Mafia fraud depending on the number of states

Let q be the probability of picking a given element of $\mathcal{A}$. Clearly, $\mathcal{A}$ is included in the set of all possible combinations of $\lceil \log_2(p) \rceil$ bits. Given that there is $2^{\lceil \log_2(p) \rceil}$ such combinations, we conclude that q is equal to $\frac{p}{2^{\lceil \log_2(p) \rceil}}$.

We now define A_i the event of picking an element of $\mathcal{A}$ at the ith draw. If this event happens, it means that the $(i-1)$th first draws failed (no element of $\mathcal{A}$ has been picked), and the ith succeeded (an element of $\mathcal{A}$ has been picked). As the draws are independent, we had shown that $P(A_i) = (1-q)^{i-1}q$. According to these observations, we deduce that $P(A_i)$ follows a geometric distribution. So the expectation of $P(A_i)$ (i.e., the average number of bit blocks of length $\lceil \log_2(p) \rceil$ needed to pick an element of $\mathcal{A}$) is $\frac{1}{q}$.

Thanks to the previous results we know that we need $\frac{1}{q} \cdot \lceil \log_2(p) \rceil$ bits to create a p-symbol. In order to filling up a register, n p-symbols have to be picked. At last the average number of bits to be generated in order to obtain a full register of n p-symbols is not $n \cdot \lceil \log_2(p) \rceil$ but:

$$n \cdot \lceil \log_2(p) \rceil \cdot \frac{2^{\lceil \log_2(p) \rceil}}{p}.$$

4.4 Memory Consumption

From a theoretical point of view, using MUSE allows to decrease the adversary success probability towards zero by increasing the value p. However the increase of p is bounded by the memory. So, if n is the number of rounds, HK needs to store $2n$ bits, MP $4n$ bits, and MUSE-p HK $np\lceil \log_2(p) \rceil$ bits. Figure 5 depicts the memory consumption given n and p. We see that memory grows quickly with the number of states. Large values of p are thus not realistic in practice. Moreover, in

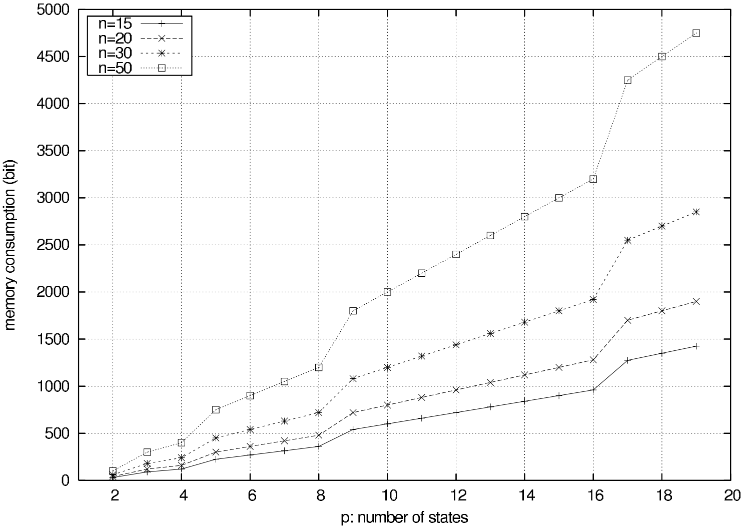


Fig. 5. Number of bits depending on the number of states

order to optimize the memory consumption, and the generation of the registers, and so ease the implementation, p has to be a power of two. Consequently, in Section 5 we provide an analysis of the performance of MUSE when $p = 4$. This is a good candidate because it is a small power of two which allows us to have a good trade-off between memory consumption, number of rounds and adversary success probability. Such a choice for p avoids the problem of generating symbols for the registers, because all of the 2-bits combinations represent a 4-symbol.

5 MUSE-4 Applied to Some Protocols

In this section, we apply MUSE-4 to some well-known distance bounding protocols and compare their performances with their original form.

5.1 Hancke and Kuhn

We begin by analyzing the performance of MUSE-4 HK with respect to memory and adversary's success probability. We then compare the results with the performances of the original HK, MUSE-3 HK, and MP.

Since MUSE-4 HK is a special case of MUSE-p HK analyzed in Section 4, the evaluation of the performances is trivial. From Section 4.4, we know that MUSE-4 HK requires the tag to store $4n$ bits where n is the number of rounds and from formula (4), we get:

$$P_{\text{MUSE-4 HK}} = \left(\frac{7}{16}\right)^n . \qquad (5)$$

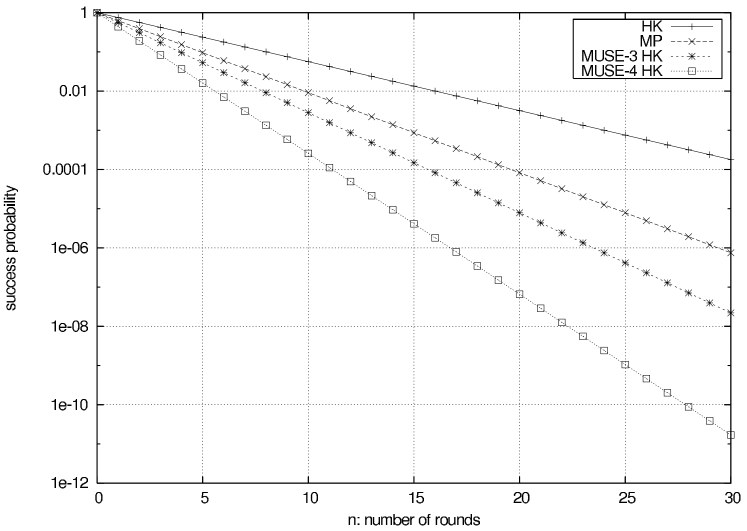


Fig. 6. Adversary's success probability

Table 1. Memory consumption, number of rounds, and adversary success probability

	HK		MP		MUSE-3 HK		MUSE-4 HK	
Probability	Memory	Rounds	Memory	Rounds	Memory	Rounds	Memory	Rounds
10^{-2}	32	16	36	9	42	7	40	5
10^{-4}	64	32	76	19	90	15	88	11
10^{-6}	94	47	116	29	138	23	128	16
10^{-8}	128	64	156	39	186	31	176	22
10^{-10}	160	80	192	48	234	39	216	27

The adversary success probabilities of HK, MP, MUSE-3 HK and MUSE-4 HK are respectively given in equations (1), (2),(3), and (5). Figure 6 shows that MUSE-4 HK clearly decreases the number of rounds of the protocol compared to HK, MUSE-3 HK, and MP, for any fixed adversary's success probability.

Figure 7 depicts the memory consumption according to the number of rounds. It shows that MUSE consumes more memory than the original versions of the protocols, but the loss is partly compensated by the reduced number of rounds. In particular, one may notice that MUSE-4 uses the memory optimally compared to MUSE-3 because 3 is not a power of 2. Table 1 summarizes our analysis for different values of adversary's success probability. It points out that MUSE-4 HK performs better than all the other candidates. For example, HK needs 32 fast phase rounds to get an adversary's success probability as low as 10^{-4}, while MUSE-4 HK needs only 11 rounds to reach the same security level.

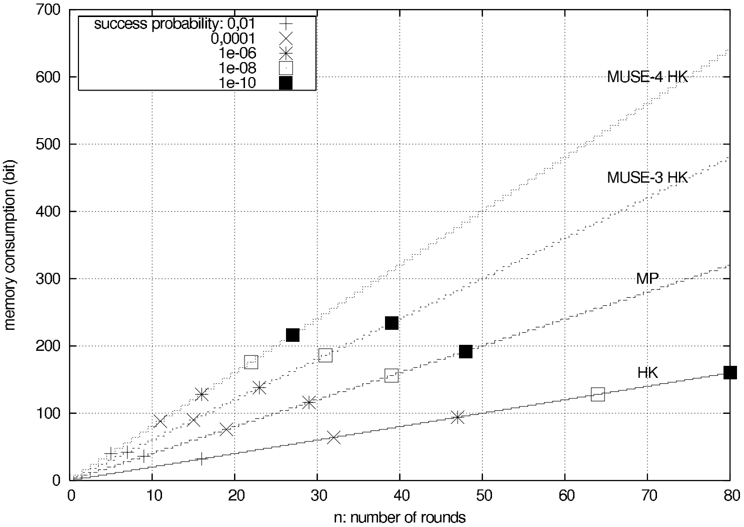


Fig. 7. Memory consumption and success probability according to the number of rounds

5.2 Kim and Avoine

Kim and Avoine's protocol (KA) [6] basically relies on *predefined* challenges. Predefined challenges allow the prover to detect that an attack occurs. However, contrarily to MP, the prover does not abort the protocol upon detection of an attack, but sends random responses to the adversary. The concept of the predefined challenges works as follows: the prover and the verifier agree on some predefined 1-bit challenges; if the adversary sends in advance a challenge to the prover that is different from the expected predefined challenge, then the prover detects the attack. The complete description of KA is provided below.

Initialization. The prover (P) and the verifier (V) share a secret x and agree on (a) a security parameter n, (b) a public pseudo random function H whose output size is $4n$, and (c) a given timing bound $\Delta t_{\max}$ and summarized in Figure 8.

Protocol. As previously, V and P exchange nonces N_P and N_V. From these values they compute $H^{4n} = H(x, N_P, N_V)$, and split it in four registers: $R^0 := H_1 \dots H_n$ and $R^1 := H_{n+1} \dots H_{2n}$ are the potential responses; the register $D := H_{3n+1} \dots H_{4n}$ constitutes the potential predefined challenges; finally, the register $T := H_{2n+1} \dots H_{3n}$ allows the verifier (resp. prover) to decide whether a predefined challenge should be sent (resp. received): in round i, if $T_i = 1$ then a random challenge is sent; if $T_i = 0$ then the predefined challenge D_i is sent instead of a random one.

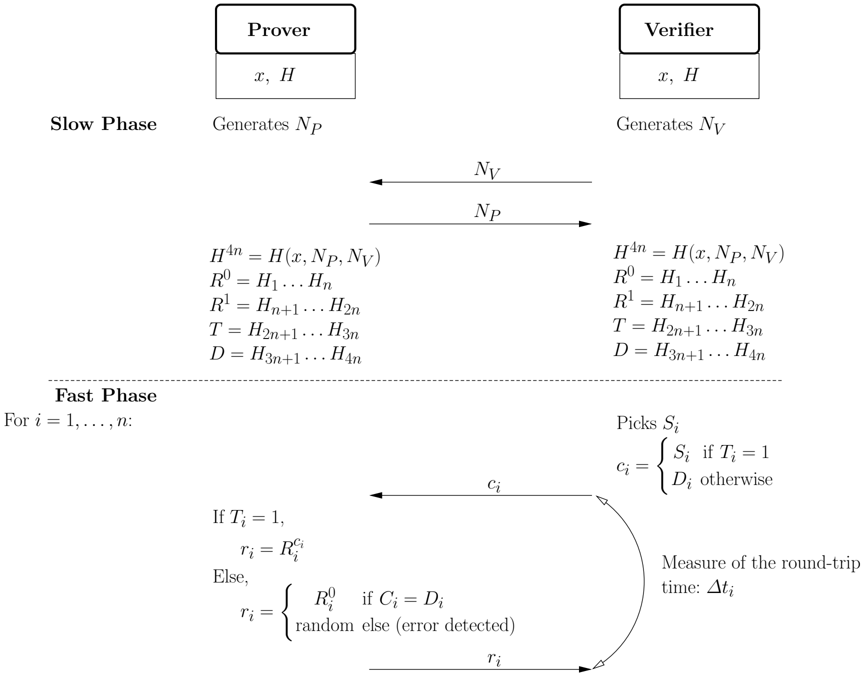

Prover
$x,\ H$
Verifier
$x,\ H$
Slow Phase
Generates N_P
Generates N_V
N_V
N_P
$H^{4n} = H(x, N_P, N_V)$
$R^0 = H_1 \dots H_n$
$R^1 = H_{n+1} \dots H_{2n}$
$T = H_{2n+1} \dots H_{3n}$
$D = H_{3n+1} \dots H_{4n}$
$H^{4n} = H(x, N_P, N_V)$
$R^0 = H_1 \dots H_n$
$R^1 = H_{n+1} \dots H_{2n}$
$T = H_{2n+1} \dots H_{3n}$
$D = H_{3n+1} \dots H_{4n}$
Fast Phase
For $i = 1, \dots, n$:
Picks S_i
$c_i = \begin{cases} S_i & \text{if } T_i = 1 \\ D_i & \text{otherwise} \end{cases}$
c_i
If $T_i = 1$,
$r_i = R_i^{c_i}$
Else,
$r_i = \begin{cases} R_i^0 & \text{if } C_i = D_i \\ \text{random} & \text{else (error detected)} \end{cases}$
Measure of the round-trip time: Δt_i
r_i

Fig. 8. Kim and Avoine's protocol

Verification. At the end of the fast phase, the verifier checks that the answers received from the prover are correct and that: $\forall\, i, i \in \{1, \dots, n\}$, $\Delta t_i \leq \Delta t_{\max}$.

MUSE-4 KA. Applying MUSE to KA does not significantly modify the protocol. Except that R^0, R^1, two additional registers R^2, R^3, and D contain 4-symbols instead of bits, the basic principle of KA remains unchanged. In order to create these registers, $11n$ random bits must be generated instead of $4n$ for KA: $R^j = H_{2jn+1} \dots H_{2n(j+1)}$ for $j \in \{0, 1, 2, 3\}$, $T = H_{8n+1} \dots H_{9n}$, and $D = H_{9n+1} \dots H_{11n}$. During the fast phase, the only difference between KA and MUSE-4 KA is that in the latter case the verifier (resp. prover) sends 4-symbol challenges (resp. responses) instead of binary challenges (resp. responses).

Comparison. In what follows, p_r is the probability that $T_i = 1$, i.e., a random challenge is expected in the ith rounds of the protocol. In the original paper [6], an adversary can choose to ask or not to ask the tag in advance. If she does not ask in advance, her success probability is $\left(\frac{1}{2}\right)^n$ for the original protocol, and it is $\left(\frac{1}{4}\right)^n$ with MUSE-4. If the adversary queries the tag in advance, the cumulated probability of not being detected by the reader in the ith round is

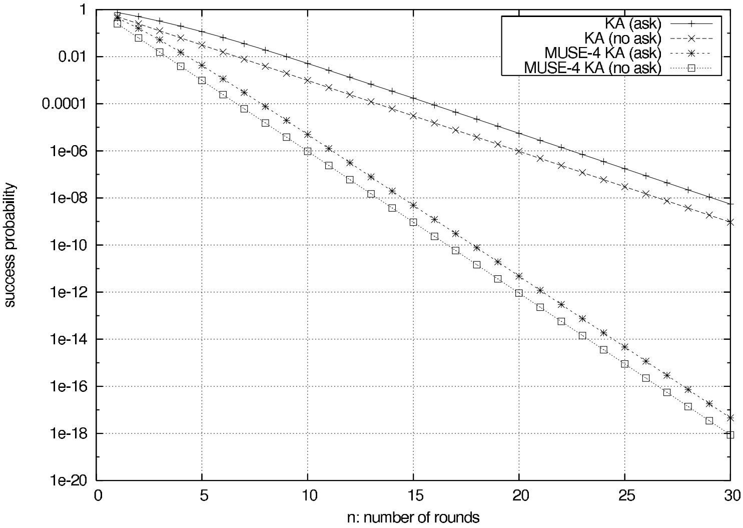


Fig. 9. Adversary's success probability when $p_r = 1/2$

Table 2. Memory consumption, number of rounds, and adversary success probability, when $p_r = 1/2$

	KA		MUSE-4 KA	
Probability	Memory	Rounds	Memory	Rounds
10^{-2}	48	12	66	6
10^{-4}	76	19	99	9
10^{-6}	88	22	121	11
10^{-8}	116	29	154	14
10^{-10}	140	35	187	17

$\frac{1}{2} + \frac{1}{4} \cdot \left(\frac{1}{2} + \frac{1}{2} \cdot p_r\right)^{i-1}$. With MUSE-4, and following the same computations as done in [6], we find that the cumulated probability of not being detected by the reader in the ith round is:

$$\frac{1}{4} + \frac{3}{16} \cdot \left(\frac{1}{4} + \frac{3}{4} \cdot p_r\right)^{i-1} .$$

Figure 9 shows how the adversary's success probability evolves, depending on the number of rounds, when $p_r = 1/2$.

As previously explained with HK, MUSE-4 KA requires the tag to store more bits than KA per round (11 bits instead of 4 per round). However, this drawback is partly compensated by the fact that MUSE-4 KA reduces the number of rounds, and so the total required memory. Table 2 points out that MUSE-4 divides by 2 the number of rounds when used with KA, while keeping the same security level.

5.3 Brands and Chaum

Brands and Chaum's protocol (BC) [2] is the earliest distance bounding protocol. BC is the protocol that provides the lowest adversary success probability for a given number of rounds that is $\left(\frac{1}{2}\right)^n$. This nice property is explained by the fact that BC requires a final signature after the fast phase of the protocol, as described below and summarized in Figure 10.

Initialization. The prover and the verifier share a secret x and agree on (a) a security parameter n, (b) a commit scheme, and (c) a given timing bound $\Delta t_{\max}$.

Protocol. Both of the prover and the verifier generate n bits, c_i for the verifier and m_i for the prover. Then the prover commits on n bits m_i using a secure commitment scheme. Afterwards, the fast phase begins. The verifier sends c_i to prover. The latter immediately answers $r_i = c_i \oplus m_i$. Once the n rounds are completed, an additional phase, called the *authentication phase* is executed: the prover opens the commitment, concatenates the $2n$ bits c_i and r_i into m, and signs it with his secret x.

Verification. After the commitment opening the verifier checks that the r_is are those he expected. Then he computes m in the same way than the prover did, and verifies that the signature is valid and that no adversary have changed the challenges or the responses. He finally checks that: $\forall\, i, 1 \leq i \leq n$, $\Delta t_i \leq \Delta t_{\max}$.

MUSE-4 BC. When using MUSE-4 BC, prover and verifier must each generate n 4-symbols, that is $2n$ bits: instead of picking some bits, they pick at random the m_i for the prover, and the c_i for the verifier in $\{0, 1, 2, 3\}$. The commit is done as usual, but the m_i are no longer encoded on one bit but on two bits.

For the fast phase, r_i is still equal to $c_i + m_i$, except that the is done modulo 4 instead of modulo 2.

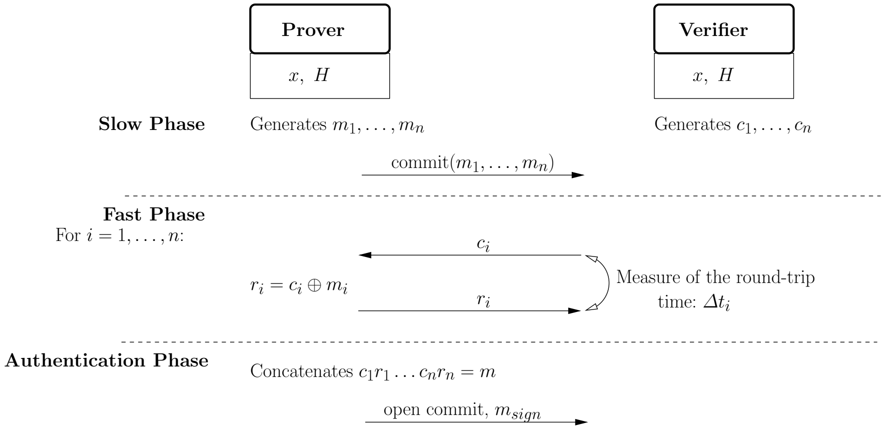

Prover
$x,\ H$
Verifier
$x,\ H$
Slow Phase
Generates $m_1, \ldots, m_n$
Generates $c_1, \ldots, c_n$
commit$(m_1, \ldots, m_n)$
Fast Phase
For $i = 1, \ldots, n$:
c_i
$r_i = c_i \oplus m_i$
r_i
Measure of the round-trip time: Δt_i
Authentication Phase
Concatenates $c_1 r_1 \ldots c_n r_n = m$
open commit, m_{sign}

Fig. 10. Brands and Chaum's protocol

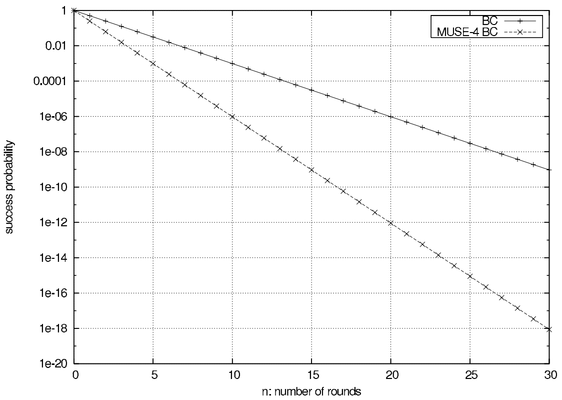


Fig. 11. Adversary success probability depending on the number of rounds for BC

For the authentication phase, the length of m is $4n$ bits: we have to map $\mathbb{F}_4$ on $\mathbb{F}_2 \times \mathbb{F}_2$. Afterwards the prover can send the signed bit-string.

Comparison. The use of MUSE-4 highly decreases the adversary success probability. For the original protocol [2], it is:

$$P_{\mathrm{BC}} = \left(\frac{1}{2}\right)^n . \tag{6}$$

This probability is explained by the fact that the adversary fails as soon as she sends a wrong challenge to the prover, due to the final signature. With MUSE-4, the adversary succeeds with probability $\frac{1}{4}$ at each round instead of $\frac{1}{2}$. We so obtain:

$$P_{\text{MUSE-4 BC}} = \left(\frac{1}{4}\right)^n . \tag{7}$$

Formulas (6) and (7), represented in Figure 11, point out the advantage of MUSE-4 BC over the original BC. Table 3 shows how the number of rounds and the memory consumption evolve for a given level of security. We can see that in the case of BC, MUSE-4 BC consumes the same memory than the original protocol, while it is twice faster.

6 On the Implementability of MUSE

Most RFID protocols in the HF and UHF frequency bands use amplitude modulation between two different states for signalling between reader and tag. For the reader-to-tag channel, the signal is typically modulated between one level that represents the carrier-wave amplitude and an attenuated level. Pulse duration coding where the duration for which the signal is attenuated is varied is used to encode two different logical symbols. For the tag-to-reader signaling, the tags

Table 3. Memory consumption, number of rounds and adversary success probability

	BC		MUSE-4 BC	
Probability	Memory	Rounds	Memory	Rounds
0.0156	12	6	12	3
0.0000153	32	16	32	8
$3.81 \cdot 10^{-6}$	36	18	36	9
$1.49 \cdot 10^{-8}$	52	26	52	13
$2.33 \cdot 10^{-10}$	64	32	64	16

either load modulate the reader signal (at LF and HF) or backscatter some of incident electromagnetic wave (UHF). Coding schemes used include Manchester, Miller, and FM0 encoding.

To accommodate MUSE-3, we need to encode three different symbol states on the reader-to-tag and tag-to-reader channel. This can be accomplished by using the existing coding schemes that define logical 0 and logical 1 and interpreting the absence of a modulated signal in the predefined timeslot for either the challenge or the response as the third symbol. This assumes that there is all other tags in the range of the reader will remain silent. Encoding MUSE-p with $p \geq 4$ using the same overall symbol periods require either more than two modulation levels, additional phase modulation or the use of higher bandwidth signals. All of which come at the expense of the signal-to-noise ratio required for reliable decoding of the signals and required complexity of the decoder.

The turn-around-times between reader and tag signaling are typically of the order of a few milliseconds in HF protocols and microseconds in UHF systems. The turn-around-times are needed to allow for the decoding of reader commands in the tag microchip and to reduce noise in the reader receiver circuitry resulting from the modulation of the reader signal. Detecting additional propagation delays resulting from relay attacks over short distances such as a few meters (corresponding to delays of the order of a tens of nanoseconds) will thus be difficult due to the large turn-around-times. In a sophisticated relay attack, the adversary can also reduce the processing delays resulting from the decoding of the original reader commands significantly by operating the tag emulator at a higher clock speed. This is possible because the emulated tag can be battery-powered and is thus not constrained by the limited power budget of an ordinary passive tag.

7 Conclusion

We introduced in this paper the concept of p-symbol that extends the void challenges suggested by Munilla, Ortiz, and Peinado, and we provided a generic p-symbol-based technique that behaves better than the original Munilla et al.'s protocol. Our solution, called MUSE, is generic in the sense that it can be used to improve any distance bounding protocols. We provided a formal analysis of MUSE in the general case, and illustrated it when $p = 3$ and $p = 4$.

We definitely believe that distance bounding protocols require further analysis since they form the only countermeasure known today against relay attacks. We are already surrounded by several billion RFID tags and so potential victims of relay attacks, especially when our tags serve as access keys or credit cards. Contactless smartcard manufacturers, and more generally RFID manufacturers, recently understood the strength of relay attacks and the threat for their business.

References

1. Avoine, G., Tchamkerten, A.: An efficient distance bounding RFID authentication protocol: balancing false-acceptance rate and memory requirement. In: Samarati, P., Yung, M., Martinelli, F., Ardagna, C.A. (eds.) ISC 2009. LNCS, vol. 5735, pp. 250–261. Springer, Heidelberg (2009)
2. Brands, S., Chaum, D.: Distance-bounding protocols. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 344–359. Springer, Heidelberg (1994)
3. Bussard, L., Bagga, W.: Distance-bounding proof of knowledge to avoid real-time attacks. In: SEC, pp. 223–238 (2005)
4. Desmedt, Y., Goutier, C., Bengio, S.: Special uses and abuses of the fiat-shamir passport protocol. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 21–39. Springer, Heidelberg (1988)
5. Hancke, G., Kuhn, M.: An RFID distance bounding protocol. In: Conference on Security and Privacy for Emerging Areas in Communication Networks – SecureComm 2005, Athens, Greece. IEEE, Los Alamitos (2005)
6. Kim, C.H., Avoine, G.: RFID distance bounding protocol with mixed challenges to prevent relay attacks. In: Cryptology ePrint Archive, Report 2009/310 (2009)
7. Kim, C.H., Avoine, G., Koeune, F., Standaert, F.-X., Pereira, O.: The Swiss-Knife RFID Distance Bounding Protocol. In: Lee, P.J., Cheon, J.H. (eds.) ICISC 2009. LNCS, vol. 5461, pp. 98–115. Springer, Heidelberg (2009)
8. Munilla, J., Ortiz, A., Peinado, A.: Distance Bounding Protocols with Void-Challenges for RFID. In: Workshop on RFID Security – RFIDSec 2006, Graz, Austria (July 2006) (Ecrypt)
9. Munilla, J., Peinado, A.: Distance Bounding Protocols for RFID Enhanced by using Void-Challenges and Analysis in Noisy Channels. Wireless Communications and Mobile Computing 8(9), 1227–1232 (2008)
10. Nikov, V., Vauclair, M.: Yet Another Secure Distance-Bounding Protocol. Cryptology ePrint Archive, Report 2008/319 (2008), `http://eprint.iacr.org/`
11. Reid, J., Neito, J.G., Tang, T., Senadji, B.: Detecting relay attacks with timing based protocols. In: Bao, F., Miller, S. (eds.) Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security – ASIACCS 2007, Singapore, Republic of Singapore, pp. 204–213. ACM, New York (2007)
12. Singelée, D., Preneel, B.: Distance bounding in noisy environments. In: Stajano, F., Meadows, C., Capkun, S., Moore, T. (eds.) ESAS 2007. LNCS, vol. 4572, pp. 101–115. Springer, Heidelberg (2007)
13. Tu, Y.-J., Piramuthu, S.: RFID Distance Bounding Protocols. In: First International EURASIP Workshop on RFID Technology, Vienna, Austria (September 2007)