Entity Authentication and Authenticated Key Exchange with Tree Parity Machines

Markus Volkmer

Hamburg University of Technology
Institute for Computer Technology
Schwarzenbergstraße 95, D-21073 Hamburg, Germany
Phone: ++49 (0)40 42878 3255, Fax: ++49 (0)40 42878 2798
markus.volkmer@tuhh.de

Abstract. This paper provides the first analytical and practical treatment of entity authentication and authenticated key exchange in the framework of Tree Parity Machines (TPMs). The interaction of TPMs has been discussed as an alternative concept for secure symmetric key exchange. Several attacks have been proposed on the non-authenticated principle. Adding and some extra entity authentication method is straightforward but outside the concept using TPMs. A simple and consequent implicit entity authentication from within the key exchange concept as an extension to the key exchange protocol is suggested. A proof for the soundness of the proposed entity authentication is given. Furthermore, next to averting a Man-In-The-Middle attack, the currently known attacks on the non-authenticated symmetric key exchange principle using TPMs can provably be averted for the authenticated variant.

1 Introduction

Symmetric key exchange based on the fast synchronization of two interacting identically structured Tree Parity Machines (TPMs) has been proposed under the name *Neural Cryptography* by Kinzel and Kanter [1]. It does not involve large numbers and principles from number theory, however, Shamir et al. conferred to this interaction over multiple rounds as *a gradual type of Diffie-Hellman* key exchange [2]. Even more related, secret key agreement based on interaction over a public insecure channel is also discussed under information theoretic aspects by Maurer and others [3–6]. Furthermore, other cryptographic principles (for authentication) based on hard learning problems (see e.g. [7]) have been discussed and have recently been proposed for application in resource-constrained environments such as RFID-Tags [8–10].

Neural Cryptography without authentication has been attacked through eavesdropping also by Shamir et al. [2] and the most recent attack using

TPMs has again been presented by Kinzel, Kanter et al. [11]. The security of the secret-key approach as well as the success of the attacks can so far only be accessed in terms of relative probabilities. No formal proofs of the achievable levels of security exist to the best of the authors knowledge and one cannot access the scheme by a reduction to number-theoretic hardness assumptions for a proof of security. Yet, the approach is often imprecisely considered broken, due to the mere existence of the attacks mentioned above for the non-authenticated variant of the symmetric principle in which a Man-In-The-Middle attack (MITM) is always possible.

Entity authentication is an important procedure still before key exchange and the en-/decryption of information with an exchanged secret key [12]. Adding usual authentication methods to Neural Cryptography is straightforward but is not embedded into the concept. It is the authors intention to formulate an entity authentication concept from within Neural Cryptography, based on the original principle and keeping the practical advantage of not operating on large numbers. The concept and early practical considerations were sketched already in [13]. A formal proof of the soundness of the authentication and the security of the authenticated key exchange is now given and experiments also demonstrate, that it averts a MITM-attack and all currently known attacks all of which also use TPMs.

1.1 The Tree Parity Machine

First, in order to discuss the proposal, the underlying principle of symmetric key exchange by interacting TPMs is briefly described. The exchange protocol is realized by an interactive adaptation process between the two interacting parties A and B. The notation A/B denotes equivalent operations for the parties A and B. A single A or B denotes an operation which is specific to one of the parties. The particular tree structure has non-overlapping binary inputs, discrete coefficients and a single binary output (Fig. 1a).

In the version using hebbian learning (cf. [1, 2]) keys are identical in synchronous TPMs, as opposed to the version using anti-hebbian learning and leading to inverted keys at the other party.

Definition 1 (Tree Parity Machine) *The TPM (see Fig. 1a) consists of K independent summation units ($1 \leq k \leq K$) with non-overlapping inputs in a tree structure and a single parity unit at the output. Each summation unit receives different N inputs ($1 \leq j \leq N$), leading to an*

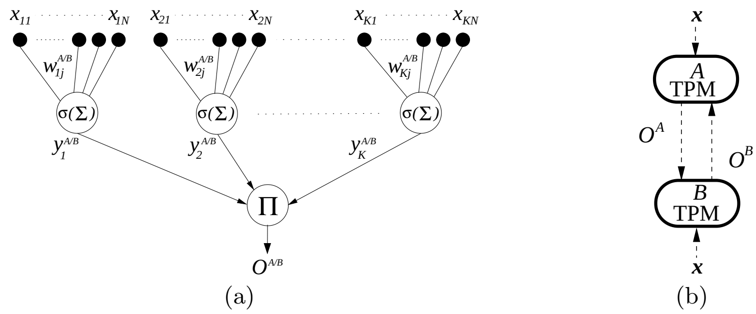

Fig. 1. (a) The Tree Parity Machine (TPM). A single output is calculated from the parity of the outputs of the summation units. (b) Outputs on commonly given inputs are exchanged between parties A and B for adaptation of their preliminary key.

input field of size $K \cdot N$. The vector-components are random variables with zero mean and unit variance.

Definition 2 (Output) *The output $O^{A/B}(t) \in \{-1, 1\}$ of a TPM, given bounded coefficients $w_{kj}^{A/B}(t) \in [-L, L] \subseteq \mathbb{Z}$ (from input unit j to summation unit k) and common random inputs $x_{kj}(t) \in \{-1, 1\}$, is calculated by a parity function of the signs of summations (σ denotes the sign-function):*

$$O^{A/B}(t) = \prod_{k=1}^{K} y_k^{A/B}(t) = \prod_{k=1}^{K} \sigma\left(\sum_{j=1}^{N} w_{kj}^{A/B}(t)\ x_{kj}(t)\right) \quad . \qquad (1)$$

The outputs of the summation units $y_k^{A/B}(t)$ cannot be uniquely determined from the communicated output $O^{A/B}(t)$, as there are multiple combinations for a signed or unsigned output depending on the number of summation units K. They also do not allow to infer the coefficients $w_{kj}^{A/B}(t)$ or the inputs $x_{kj}(t)$. The inputs of each summation unit are mixed with the coefficients in the sum and are non-linearly mapped to a single bit. The final output is again a reduction to a single parity of K bits.

1.2 Key Agreement by Interaction of Tree Parity Machines

The so-called *bit package* variant (cf. [1]) reduces (physical) output exchanges by an order of magnitude down to around a few packages. It is thus advantageous for practical communication channels with a certain protocol overhead.

Definition 3 (Key Agreement Protocol) *Parties A and B start with an individual randomly generated secret initial vector $w_{kj}^{A/B}(t_0)$. These initially uncorrelated random variables become correlated and identical over time through the influence of the common inputs and the interactive adaptation as follows. After a set of $b > 1$ presented inputs, where b denotes the size of the bit package, the corresponding calculated b TPM outputs (bits) $O^{A/B}(t)$ are exchanged over the public channel in one package (see Fig. 1b). Note that $b = 1$ results in the single-bit variant as a special case. The b sequences of signs of the summation units $y_k^{A/B}(t) \in \{-1, 1\}$ are stored for the subsequent adaptation process.*

Definition 4 (Hebbian Learning Rule) *A hebbian learning rule adapts the coefficients (the preliminary key), using the b outputs and b sequences of signs. They are changed only on equal output bits $O^A(t) = O^B(t)$ at both parties. Furthermore, only coefficients of those summation units are changed, that agree with this common output:*

$$w_{kj}^{A/B}(t+1) := \begin{cases} w_{kj}^{A/B}(t) + O^{A/B}(t)\ x_{kj}(t) & , O^{A/B}(t) = y_k^{A/B}(t) \\ w_{kj}^{A/B}(t) & , \textit{otherwise.} \end{cases} \tag{2}$$

Definition 5 (Bounding Operation) *Coefficients are always bound to remain in the maximum range $[-L, L] \subseteq \mathbb{Z}$ by reflection onto the boundary values*

$$w_{kj}^{A/B}(t+1) := \begin{cases} \sigma\left(w_{kj}^{A/B}(t+1)\right) L & , |w_{kj}^{A/B}(t+1)| > L \\ w_{kj}^{A/B}(t+1) & , \textit{otherwise.} \end{cases} \tag{3}$$

Iterating the above procedure in as an interactive protocol, each component of the preliminary key performs a random walk with reflecting boundaries [14]. The resulting key space is of size $(2L+1)^{KN}$. Two corresponding components in $w_{kj}^A(t)$ and $w_{kj}^B(t)$ receive the same random component of the common input vector $x_{kj}(t)$. After each bounding operation (Eq. 3), the distance between the two components is successively reduced to zero (so-called attractive steps). Thus the non-linear mapping from the inputs to the output bit (Eq. 1) is also changed in each learning step depending on the interaction of the two parties and their secret random initial coefficients.

When both parties adapted to produce each others outputs, common coefficients are present in both TPMs in each of the following iterations. They remain synchronous (see learning rule Eq. 2) and continue to produce the same outputs on every commonly given input. Parties with identical inputs always converge to a dynamic common trajectory, whereas parties with differing inputs always diverge as will also be shown later in Section 2. The preliminary key in the form of the coefficients has never been communicated between the two parties, only depends on the random initial coefficients of both parties and can be used as a common time-dependent final key (possibly after applying privacy amplification [15]).

A practical test for an accomplished key exchange is to test on successive equal outputs in a sufficiently large number of iterations t_{min}, such that equal outputs by chance are excluded:

$$\forall t \in [t', \cdots, t' + t_{min}] : \; O^A(t) = O^B(t) \; . \tag{4}$$

The time to accomplish key exchange is almost independent on N and scales with $\ln N$ for very large N. Furthermore, it is proportional to L^2 [16] and to $\ln K$. Own investigations confirmed the time to be distributed and peaked around 400 (i.e. thirteen 32-bit packages) for the parameters given in [1].

1.3 Attacks using Tree Parity Machines

For the key exchange protocol without authentication, eavesdropping attacks have concurrently been proposed by Shamir et al. [2] and Kanter, Kinzel et al. [16, 11]. The attacks in [2, 16, 11] can all be made arbitrarily costly and thus can be defeated by simply increasing the parameter L. The security increases proportional to L^2 while the probability of a successful attack decreases exponentially with L [16]. The approach is thus regarded computationally secure with respect to these attacks for sufficiently large L [17, 11]. The latest attack, which does not seem to be affected by an increase of L (but still by an increase of K) uses 100 coordinated and communicating TPMs [11].

A drawback of the theoretically scalable security levels is their impracticality for large values especially of N and L. They affect the average synchronization time and also the memory required for an implementation. A recent variant that tries to improve the security against TPM attackers is relatively costly in comparison with the original proposal [18].

It is important to note, though, that all of the existing attacks refer to a non-authenticated key exchange, in which a MITM-attack on the symmetric principle is possible as well.

2 Entity Authentication with Tree Parity Machines

Generally, for mutual authentication the two parties engage in a conversation to increase their confidence that it is a specific other party with whom they communicate. Additionally exchanging a new secret (session) key leads to authenticated key exchange.

In the following, assume that all communication among interacting parties is under the adversary's control. In particular, the adversary can read the bit packages produced by the parties, provide her own bit packages to them, modify bit packages before they reach their destination, and delay bit packages as well as replay them.

A simple but effective suggestion for the incorporation of entity authentication into Neural Cryptography is discussed in the following. The scheme cannot be reduced to number-theoretic hardness assumptions. Yet, a proof that the proposed authentication is sound is given, as well as a proof of its security with regard to eavesdropping-attacks that also use TPMs.

2.1 Implicit Entity Authentication

The structure of the network, the involved computations producing the output $O^{A/B}(t)$ (Eq. 1), the adaptation-rule (Eq. 2) and especially the common inputs $x_{kj}(t)$ are public in the original protocol. The different initial preliminary keys $w_{kj}^{A/B}(t_0)$ of the two parties are the only secret information. If they were public, the resulting final keys could simply be calculated (by an adversary), because all further computations are completely deterministic.

An implicit solution to include authentication into the neural key exchange protocol bases on the simple but strong fact, that two interacting parties A and B which have different input vectors

$$x^A(t) \neq x^B(t); \quad x^A(t), x^B(t) \in \{0,1\}^{KN} \tag{5}$$

cannot become synchronous. In the following we Consider two TPMs A and B (and an attacking TPM E) with identical structure in which $k = 1, \cdots, K$ denotes the number of summation units y_k and $j = 1, \cdots, N$ denotes the j-th component of a summation k. Let x_{kj} denote the the input to component j of summation unit k and let w_{kj} denote the j-th weight of summation unit k.

Definition 6 (Synchronous TPMs) *Two TPMs A and B are synchronous at iteration t_s when all their weights are identical:*

$$w_{kj}^A(t_s) = w_{kj}^B(t_s) \quad \forall k, j \ . \tag{6}$$

Definition 7 (Synchronous Summation Units) *Two corresponding summation units k of two TPMs A and B are synchronous at iteration t_s when all their weights (components) are identical:*

$$w_{kj}^A(t_s) = w_{kj}^B(t_s) \quad \forall j \ (k \textit{ fixed}) \ . \tag{7}$$

A lemma is proven first which shows that once a summation unit k is synchronous (i.e. it is in an identical state with the other party) it remains synchronous for all subsequent iterations.

Lemma 1 (Hidden Unit Lemma) *Two corresponding summation units y_k^A and y_k^B of two TPMs A and B that have identical internal states $w_{kj}^A(t) = w_{kj}^B(t)$, $\forall j$ (k fixed) at an arbitrary iteration t_s (that are synchronous at an arbitrary iteration t_s) remain synchronous for all $t > t_s$ when having the same inputs and applying the same learning rule (Eq. 2) and bounding operation (Eq. 3).*

Proof. Consider the subsequent iteration at time $t_s + 1$. Formally, two cases can be distinguished:

1. *No Adaptation at iteration $t_s + 1$.* If $O^A(t_s + 1) \neq O^B(t_s + 1)$, no adaptation is performed and in this trivial case

$$w_{kj}^A(t_s + 1) = w_{kj}^A(t) = w_{kj}^B(t) = w_{kj}^B(t_s + 1), \ k \text{ fixed} \ , \tag{8}$$

   i.e. the summation unit remains synchronous.
2. *Adaptation at iteration $t_s + 1$.* If $O^A(t_s + 1) = O^B(t_s + 1) = y_k^A(t_s + 1) = y_k^B(t_s + 1)$, an adaptation is performed and each component j of the weight vector of hidden unit k of both parties will be changed according to the same learning rule in Eq. 2. Hence,

$$w_{kj}^{A/B}(t_s + 1) = w_{kj}^{A/B}(t_s) + O^{A/B}(t_s + 1) \ x_{kj}^{A/B}(t_s + 1), \ k \text{ fixed} \tag{9}$$

   for the parties A and B. The adaptation is performed in the same direction as $O^A(t_s+1) = O^B(t_s+1)$ and $x_{kj}^A(t_s+1) = x_{kj}^B(t_s+1) \ \forall k, j$. Thus $w_{kj}^A(t_s + 1) = w_{kj}^B(t_s + 1)$, $\forall j$ (k fixed) – the summation units remain synchronous.

Theorem 1 (Authentication: Soundness). *A Tree Parity Machine (TPM) B having an identical structure to TPM A (as defined by the parameters K, N and L), as well as with identical output generation as defined by Eq. 1 cannot become synchronous by updating its weights according to the learning rule Eq. 2 and the bounding operation Eq. 3, when having different inputs from party A.*

Proof. Consider two TPMs A and B with identical structure. Obviously, two TPMs can only become synchronous, when all their corresponding K summation units become synchronous. W.l.o.g. consider just one summation unit k' that is not yet synchronous. As known from the Hidden Unit Lemma above, the synchronous summation units $k \neq k'$ remain synchronous when applying the same inputs in A and B. These hidden units will always generate the same hidden unit outputs, i.e. $y_k^A(t) = y_k^B(t),\ k \neq k'$. Consequently, if an adaptation of the remaining non-synchronous summation unit k' takes place at an arbitrary iteration, then $O^A = O^B = O^{A/B} = y_{k'}$. Thus in the following it suffices to restrict the considerations to the one remaining non-synchronous summation unit k'. Whenever a summation unit is adapted, all of its components are adapted by definition of the TPM Learning algorithm. Furthermore the adaptation is performed in the same direction and is the same for both TPMs by definition of the TPM learning rule Eq. 2. W.l.o.g. lets further assume only one component remains that is not identical at iteration t_s, i.e. $w_{k'j}^A(t_s) \neq w_{k'j}^B(t_s)$ for a particular component j. If for a subsequent iteration t_s+1, $x_{k'j}^A(t_s+1) \neq x_{k'j}^B(t_s+1)$ for at least this one component j in summation unit k' the following two situations can occur:

1. *No Adaptation at iteration* t_s+1. If no adaptation is performed in iteration t_s+1 (as $O^A \neq O^B$ or $O^{A/B} \neq y_{k'}$), summation unit k' remains unchanged and one still has $w_{k'j}^A(t_s+1) \neq w_{k'j}^B(t_s+1)$ regardless of the applied input.
2. *Adaptation at iteration* t_s+1. If an adaptation is performed in iteration t_s+1, as $O^A = O^B = O^{A/B} = y_{k'}$, all components j of summation unit k' of TPM A and B will be adapted according to the learning rule Eq. 2. For TPM A this yields

$$w_{k'j}^A(t_s+1) = w_{k'j}^A(t_s) + O^A(t_s)\ x_{k'j}^A(t_s+1)\ . \tag{10}$$

Party B uses the same learning rule but here with a differing input $x_{k'j}^B(t_s+1) \neq x_{k'j}^A(t_s+1)$ for at least one summation unit k' and at least one component j. This yields

$$w_{k'j}^B(t_s+1) = w_{k'j}^B(t_s) + O^A(t_s)\ x_{k'j}^B(t_s+1) \tag{11}$$

for the components j of summation unit k' of party B. Consequently, $w_{k'j}^A(t_s+1) \neq w_{k'j}^B(t_s+1)$ for at least one summation unit k' and at least one component j. Thus the parties are not synchronous in iteration t_s+1.

As inputs are considered to be different for any subsequent iteration for at least one arbitrary component j in each summation unit k, A and B cannot remain synchronized.

Note that in practice, the inputs are generated from a pseudo-random number generator such as a LFSR and the subsequent inputs are evenly distributed such that a different initialization of the LFSR yields unequal inputs in each iteration.

The absence of synchronization in the case of no common inputs enables us to incorporate (symmetric) authentication by keeping the common (pseudo-random) inputs $x^{A/B}(t)$ secret between the two parties in addition to their individual secret (random) initial vector $w^{A/B}(t_0)$. There are $2^{KN} - 1$ possible common inputs as second initial secrets, which is a large enough practical amount for the parameters as chosen in [1] that makes brute force attacks computationally very expensive.

Even more, a MITM-attack and all other currently known attacks [2, 11] using TPMs are averted on principal by such an authentication, as will be shown in the proof to the next theorem. Especially the MITM-attacker would have to be able to synchronize with respect to two sides (both parties), which he can't if he does not even produce the same inputs. An attack by learning cannot be successful if the inputs are different. It is important to note, that such a second secret does not represent any principal disadvantage to the symmetric approach, because a basic common information is always necessary for some identification.

Theorem 2 (Security vs. TPM Attacks). *An attacker E using a TPM with identical structure to the TPMs of parties A and B (as defined by the parameters K, N and L), as well as with identical output generation as defined by Eq. 1, can never remain synchronous with A or B having different inputs from the synchronizing parties A and B.*

Proof. Consider the two TPMs A and B and a third TPM of Attacker E all with identical structure. Suppose parties A and B are not synchronous at iteration t_s, i.e. $w^A_{kj}(t_s) \neq w^B_{kj}(t_s)$ for at least one component j in an arbitrary summation unit k. W.l.o.g. let the attacker E already be synchronous to A (or B) at iteration t_s, i.e. $w^A_{kj}(t_s) = w^E_{kj}(t_s)\ \forall k, j$. Note that if the attacker was synchronous to A and B, the two parties themselves would be synchronous already.

Again, two TPMs can only become synchronous, when all their corresponding K summation units become synchronous. W.l.o.g. consider just one summation unit k' that is not yet synchronous. As known from the

Hidden Unit Lemma above, the synchronous summation units $k \neq k'$ remain synchronous when applying the same inputs in A and B. These hidden units will always generate the same hidden unit outputs, i.e. $y_k^A(t) = y_k^B(t),\ \ k \neq k'$. Consequently, if an adaptation of the remaining non-synchronous summation unit k' takes place at an arbitrary iteration, then $O^A = O^B = O^{A/B} = y_{k'}$.

Thus in the following it suffices to restrict the considerations to the one remaining non-synchronous summation unit k'. Whenever a summation unit is adapted, all of its components are adapted by definition of the TPM Learning algorithm. Furthermore the adaptation is performed in the same direction and is the same for both TPMs by definition of the TPM learning rule Eq. 2. W.l.o.g. lets further assume only one component remains that is not identical at iteration t_s, i.e. $w_{k'j}^A(t_s) \neq w_{k'j}^B(t_s)$ for a particular component j. For any subsequent iteration $t > t_s$, in which A and B continue to synchronize, let $x_{k'j}^{A/B} \neq x_{k'j}^E$ for at least this one component j in summation unit k', the following two situations can occur:

1. *No Adaptation at iteration* $t_s + 1$ If no adaptation is performed in iteration $t_s + 1$ (as $O^A \neq O^B$ or $O^{A/B} \neq y_{k'}$), all components of all parties remain unchanged and still $w_{kj}^A(t_s + 1) \neq w_{kj}^B(t_s + 1)\ \forall k, j$, as well as $w_{kj}^A(t_s) = w_{kj}^E(t_s)\ \forall k, j$.
2. *Adaptation at iteration* $t_s{+}1$ If an adaptation is performed in iteration $t_s + 1$, as $O^A = O^B$, all components j of summation unit k' of TPM A, B and E will be adapted according to the learning rule Eq. 2. This yields
$$w_{k'j}^{A/B}(t_s + 1) = w_{k'j}^{A/B}(t_s) + O^{A/B}(t_s)\ x_{k'j}^{A/B}(t_s + 1) \tag{12}$$
for the parties A and B.
The attacker performs uses the same learning rule but with a differing input $x_{k'j}^E(t_s + 1) \neq x_{k'j}^{A/B}(t_s + 1)$ for at least one j, yielding
$$w_{k'j}^E(t_s + 1) = w_{k'j}^E(t_s) + O^{A/B}(t_s)\ x_{k'j}^E(t_s + 1) \tag{13}$$
for the attacker.
Consequently, $w_{k'j}^A(t_s + 1) \neq w_{k'j}^E(t_s + 1)$ for at least one summation unit k' and at least one component j and the attacker is not synchronous in iteration $t_s + 1$.

As inputs are considered to be different for any subsequent iteration for at least one arbitrary component j in each summation unit k, E cannot remain synchronous even if he is synchronous (by guessing e.g.) in one iteration.

The original aim of the two-party-interaction is to adapt coefficients such that both parties produce identical outputs on commonly given inputs. Given different inputs, the two parties are trying to adapt completely different non-linear relations between (different) inputs $x^A(t) \neq x^B(t)$ and outputs $O^{A/B}(t)$. More concretely, the random walks with reflecting boundaries performed by the coefficients in the iterative process now make uncorrelated moves. Even moves in the wrong direction with regard to the aim of learning common outputs are made (cf. [19, 14]). Two corresponding components $w^A_{kj}(t)$ and $w^B_{kj}(t)$ now receive a different random component $x^A_{kj}(t) \neq x^B_{kj}(t)$ of their (differing) input vectors (cf. Eq. 1). The distance between the components is thus no longer successively reduced to zero after each bounding operation and the two parties' coefficients remain different. Parties with identical inputs always converge to the dynamic common trajectory. Parties with differing always diverge.

Consequently, when the two parties do not become synchronous, there also will not be time-dependent equal coefficients $w^{A/B}(t)$ and thus on principle no key exchange. It is exactly the service one would want to restrict to authorized parties by employing any other method for entity authentication. Legitimate partners have the advantage over an attacker in the form of the information about each others pseudo-random numbers given as identical initializations of their pseudo-random number generator. This secret common information is by no means a key as it does not influence the key exchange process itself – the final key is determined by the random secret initial coefficients $w^A_{kj}(t_0)$ and $w^B_{kj}(t_0)$ of the two parties.

2.2 Practical Evaluation

For demonstration, the development of distance between two preliminary keys is investigated over time as their normalized sum of absolute differences

$$D(w^A(t), w^B(t)) = \frac{1}{KN\ 2L} \sum_{k=1}^{K} \sum_{j=1}^{N} |w^A_{kj}(t) - w^B_{kj}(t)| \ \in [0, 1] \qquad (14)$$

for different offsets in the (pseudo-random) sequence of inputs (input vectors) and for completely different inputs:

$$\forall t\colon\ x^A(t) = x^B(t + \Delta),\ \Delta \in \mathbb{N}\ . \qquad (15)$$

The first situation represents a party (respectively an attacker), who has a different initialization of his pseudo-random number generator. The second situation is typical for a party (respectively an attacker) with incomplete (missing) inputs or even completely differently generated inputs.

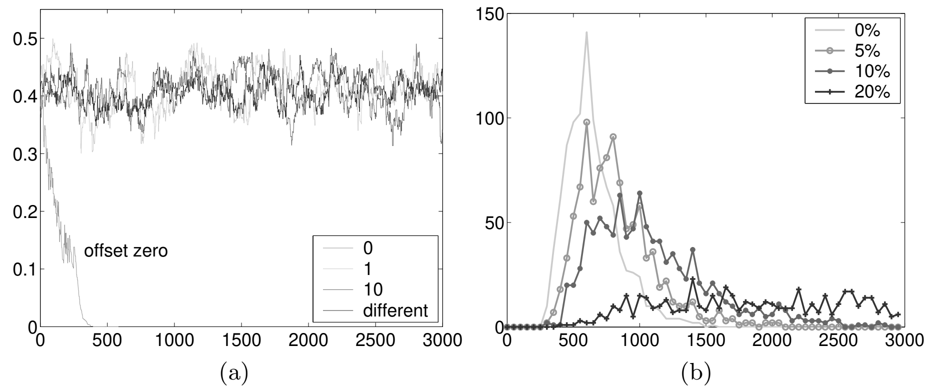

Fig. 2. (a) Distance D as defined in Eq. 14 vs. the number of exchanged bits (iterations t) for offset zero (successful authentication), offsets one and ten, as well as for completely different inputs. (b) Distribution of the iterations necessary to accomplish key exchange for different percentages of noise on the communicated output bits of one party. The curves for one and two percent noise were omitted, as they almost match with the zero percent curve. An average over 1000 runs was taken.

Figure 2a displays that the distance between two parties with different inputs remains fluctuating does not decrease towards zero. Different offsets were investigated with the same qualitative outcome. Completely different inputs (although not realistic given a concrete and publicly known Linear Feedback Shift Register (LFSR) as pseudo random number generator) show the same qualitative behavior. One can constitute, that on average there must be as many repulsive as attractive steps for this behavior (cf. [19]). Two parties with the same inputs (offset zero) soon decrease their distance and become synchronous.

In order stress the importance of having common inputs, a second test with identical inputs but with a certain percentage of equally distributed 'noise' imposed on the communicated outputs of one party is performed. If such a noise would appear only in a certain period, the system would still become synchronous but with a delay of roughly the length of the noisy period plus the time used up for unsuccessful synchronization before the noisy period, which is thus not the interesting case. Figure 2b shows that the distribution of synchronization times is flattened and biased towards longer times for increasing noise. Surprisingly, the system can still become synchronous even with a highly noisy communication channel.

In case of a noisy bit at an arbitrary iteration two events may occur: Either originally equal outputs become unequal and a learning step is skipped, or originally unequal outputs become equal and a learning step is forced. The occurrence of a skipped learning step only leads to longer synchronization times, as weights remain unchanged in both parties. The consequence to a forced learning step is also an increase in the synchronization time as a learning step is performed as opposed to the intended learning goal to learn equal outputs on equal inputs. Obviously, the (coordinated) inputs basically determine the synchronization. Of course, the average synchronization time increases as does the probability for a late synchronization.

Furthermore, a corrupt party, who provides her own bit packages to the parties or modifies bit packages, can only destroy the synchronization and consequently the exchange of a key, as would a serious jamming of the channel. Delayed bit packages delay the synchronization. Replaying transmissions does not seem to make sense, because the content of the bit packages varies depending on the secret random initial coefficients of the two parties and their interaction. Only the existing attacks on the non-authenticated protocol by learning (e.g. on previously recorded bit package transmissions) can be successful with a certain probability.

Let us recapitulate the arguments for the properties of Completeness and Soundness (see e.g. [12]) in the context of the proposed implicit authentication:

- *Completeness – A always succeeds in convincing B if he knows the common secret:* If A knows the common secret in the form of having the same inputs, he will always synchronize within a finite time (typically around 400 iterations for the parameters used in [1]). A proof of this property is a proof of convergence and is much more difficult (and lengthy) than the proof of soundness. It is supposed to appear in a forthcoming paper.
- *Soundness – A succeeds with (arbitrary) small probability if he does not know the secret of B:* If A does not know the common secret and has different inputs, synchronization will fail. The two parties will always be driven apart again by the repulsive steps. A will thus succeed with a probability of zero. This was proven in Theorem 1.

As the only information transmitted is the parities of unknown bitstrings that were non-linearly and dynamically mapped to a single bit, no information on the common secret should be leaked while the interactive protocol is performed. An inversion of the non-linear dynamic mapping is not known. Each single output parity bit should not reveal any information on its corresponding $K \cdot N$ input bits or the current $K \cdot N \cdot L$ coefficient bits.

3 Conclusion

An implicit entity authentication from within the concept of Neural Cryptography was suggested for discussion. Using the common inputs as a second secret for authentication, naturally integrates an interactive entity authentication protocol into the already interactive symmetric key exchange concept. The soundness of the proposed protocol was proven.

The bit packaging variant of the key exchange protocol together with the extension represents a parallel interaction protocol (cf. [12]), in which a number of problems (b outputs of party A) are posed an and a number of solutions (b corresponding outputs of party B) at a time are asked. This reduces the number of interaction messages on a slow-response-time connection or low-bandwidth channel.

Using entity authentication is necessary to avert a MITM-attack and it provably also averts all currently known attacks (using TPMs), which assume the full knowledge on the inputs to the TPMs.

A proof of completeness results in a proof of convergence. It is much lengthier than the presented proof of soundness and shall appear in a forthcoming paper. Future theoretic investigations should relate this non-classical key exchange principle from outside number-theoretic foundations to information-theoretic concepts (see e.g. [3–5]), as well as to hard learning problems and statistical learning theory to advance its discussion within the cryptography community.

The general trade-off in applied cryptography between available resources and the required level of security also applies using the TPM principle. In many practical embedded security solutions e.g. [20, 21] it is often admissive to provide a system safe enough for the particular application, and given certain attack scenarios. The TPM principle (extended with the proposed entity authentication) has some advantages for such embedded applications due to its hardware-friendly basic operations. It can also be used to perform group key exchange and allows to derive a stream cipher [10, 22].

Acknowledgements

The author wishes to thank André Schaumburg for experimental evaluations and discussions on the concept. He is grateful to Florian Grewe, Sebastian Wallner and Karl-Heinz Zimmermann for their comments on the proofs.

References

1. Kanter, I., Kinzel, W., Kanter, E.: Secure exchange of information by synchronization of neural networks. Europhysics Letters **57** (2002) 141–147
2. Klimov, A., Mityagin, A., Shamir, A.: Analysis of neural cryptography. In: Advances in Cryptology – ASIACRYPT 2002. Volume 2501 of LNCS., Springer Verlag (2002) 288–298
3. Maurer, U.: Protocols for secret key agreement by public discussion based on common information. In: Advances in Cryptology – CRYPTO 1992. Volume 740 of LNCS., Springer Verlag (1993) 461–470
4. Maurer, U.: Secret key agreement by public discussion. IEEE Transactions on Information Theory **39** (1993) 733–742
5. Brassard, G., Savail, L.: Secret-key reconciliation by public discussion. In: Advances in Cryptology – EUROCRYPT 1993. Volume 765 of LNCS., Springer-Verlag (1994) 410–423
6. Gander, M., Maurer, U.: On the secret-key rate of binary random variables. In: Proc. of the IEEE International Symposium on Information Theory (Abstracts). (1994)
7. Blum, A., Furst, M., Kearns, M., Lipton, R.J.: Cryptographic primitives based on hard learning problems. In: Proceedings of the 3rd Annual International Cryptology Conference, Santa Barbara, California, USA, CRYPTO'93. Volume 773 of Lecture Notes in Computer Science., Springer (1994) 278–291
8. Juels, A.: RFID: The problems of cloning and counterfeiting. In: European Network of Excellence for Cryptology Workshop on RFID and Lightweight Crypto, Graz University of Technology, Graz, Austria. (2005)
9. Juels, A., Weis, S.: Authenticating pervasive devices with human protocols. In: Proceedings of the 25th Annual International Cryptology Conference, Crypto'05, Santa Barbara, California, USA. (2005)
10. Volkmer, M., Wallner, S.: Tree parity machine rekeying architectures. IEEE Transactions on Computers **54** (2005) 421–427
11. Kanter, I., Kinzel, W., Shacham, L., Klein, E., Mislovaty, R.: Cooperating attackers in neural cryptography. Phys. Rev. E **69** (2004)
12. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. 5th edn. CRC Press (2001)
13. Schaumburg, A.: Authentication within tree parity machine rekeying. Reihe Informatik, Technical Report TR 2004-10, Universität Mannheim (2004) Stefan Lucks and Christopher Wolf eds.
14. Ruttor, A., Reents, G., Kinzel, W.: Synchronization of random walks with reflecting boundaries. J. Phys. A: Math. Gen. **37** (2004) 8609–8618
15. Maurer, U., Wolf, S.: Secret key agreement over a non-authenticated channel — part iii: Privacy amplification. IEEE Transactions on Information Theory **49** (2003) 839–851

16. Mislovaty, R., Perchenok, Y., Kanter, I., Kinzel, W.: Secure key-exchange protocol with an absence of injective functions. Phys. Rev. E **66** (2002)
17. Rosen-Zvi, M., Klein, E., Kanter, I., Kinzel, W.: Mutual learning in a tree parity machine and its application to cryptography. Phys. Rev. E. **66** (2002)
18. Ruttor, A., Kinzel, W., Kanter, I.: Neural cryptography with queries. Journal on Statistical Mechanics **P01009** (2005)
19. Ruttor, A., Kinzel, W., Shacham, L., Kanter, I.: Neural cryptography with feedback. Phys. Rev. E **69** (2004)
20. Sarma, S.E., Weis, S.A., Engels, D.W.: RFID systems and security and privacy implications. In Kaliski, B., ed.: Proc. of the Workshop on Cryptographic Hardware and Embedded Systems 2002, CHES 2002. Volume 2523 of LNCS., Springer-Verlag (2003) 454–469
21. Weis, S.A., Sarma, S.E., Rivest, R.L., Engels, D.W.: Security and privacy aspects of low-cost radio frequency identification systems. In Hutter, D., ed.: Proc. of the 1st International Conference on Security in Pervasive Computing, SPC 2003. Volume 2802 of LNCS., Springer-Verlag (2004) 201–212
22. Volkmer, M., Wallner, S.: Lightweight key exchange and stream cipher based solely on tree parity machines. In: European Network of Excellence for Cryptology Workshop on RFID and Lightweight Crypto, Graz University of Technology, Graz, Austria. (2005) 102–113