Analysis and Evaluation of Three Methods for Tag Identification in OSK RFID Protocol

Violeta Tomašević and Milo Tomašević

***Abstract* — Feasibility for improving the efficiency of the tag identification phase in OSK (Ohkubo, Suzuki i Kinoshita) RFID protocol is analyzed in this paper. First, the basic identification technique is presented in general. Then, a procedure for speeding up the identification by employment of TMTO (*Time Memory Trade Off*) approach is explained. Balancing between memory and time requirements is achieved with the chaining technique and introduction of an off-line, preparatory phase. As an additional effort, a second TMTO-based level is proposed. It relies on a special look-up table which consists of the equidistant states generated during preparatory phase. The complexity of the proposed technique is discussed along with some numerical evaluation results obtained through experiments.**

***Keywords* — OSK protocol, TMTO method, tag identification, state look-up table, chaining.**

I. Introduction

MASSIVE distant identification of objects is provided by systems with three main elements: tags (embedded into objects), a reader (which enables radio-wave communication) and a back-end server (which carries out the procedures for identification of the tag and object which sent the message). Efficiency of these RFID-based systems is determined by the time needed for the tag identification [1]. Besides the identification algorithm involved, the identification time can be significantly influenced by the amount of available memory storage on the back-end server. Therefore, the identification process often implies an appropriate trade-off between identification time and memory size by employing TMTO method [2,3].

Numerous RFID protocols based on different identification techniques have been developed [4-6]. First, this paper analyzes an RFID algorithm proposed by a group of authors led by Okhubo, widely recognized in the literature as OSK protocol [7]. Then, an inefficiency of the identification phase is emphasized and an existing TMTO-based enhancement of this algorithm is presented. Finally, the main contribution of this paper is the proposal of a further improvement based on applying the second TMTO level in the process of tag identification. Since it relies on a table with precomputed states, it is referred to as the state table based method. The performance of the proposed method is briefly evaluated for some typical parameters.

Basic OSK protocol is described in section II. The subject of section III is an analysis of the problems being encountered in the tag identification and its overcoming by using two TMTO chain-based techniques. In section IV time complexities of all three techniques are determined based on counting predominant operations – hash function calculations. Comparative performance evaluation for varying representative values of relevant parameters is provided in section V. Finally, a brief summary of conducted work and its findings along with avenues for future research are brought in section VI.

Paper received May 06, 2019; accepted June 26, 2019. Date of publication July 31, 2019. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Miroslav Lutovac.

This paper is revised and expanded version of the paper presented at the 26th Telecommunications Forum TELFOR 2018 [17].

This paper is supported by Ministry of education, science and technological development of Serbia (projects III 44006 and III 44009).

Violeta Tomašević, Singidunum University, Danijelova 32, 11000 Belgrade, Serbia (telephone: 381-64-3343933, e-mail: vitomasevic@singidunum.ac.rs).

Milo Tomašević, School of Electrical Engineering, University of Belgrade, Bulevar kralja Aleksandra 73, 11120 Belgrade, Serbia (telephone: 381-64-1291800, e-mail: mvt@rs.rs).

II. OSK protocol

OSK protocol belongs to the class of RFID systems which employ the chaining technique along with the use of hash functions [8,9]. Two hash functions, G and H, are needed for execution both in an object with a tag and on the back-end server. Function G serves for generation of a message (a random value which a tag sends to the reader). After sending this message, the tag obtains a new state generated by executing H function.

OSK protocol involves three activities: initialization of all tags, communication between reader and tags/back-end server, and identification of the tag which sent the message.

Tag initialization assumes assigning a unique identifier to each tag in the system and defining an initial state for each tag. If an RFID system supports N tags, random values which represent initial tag states s_i^1, $1 \le i \le N$, are stored into memories of the corresponding tags, while pairs (ID_i, s_i^1), $1 \le i \le N$, (ID_i is identifier of the i-th tag) are stored into memory of the back-end server.

Communication between the reader and a tag starts when, after obtaining the request from the reader, tag T_i applies G hash function on the current tag state and sends this value to the reader. After that, the tag executes H hash function on its current state and generates a new state which is stored into tag's memory. In this way, during communication tag T_i passes through states $s_i^1, s_i^2, \ldots, s_i^k, \ldots$ (where $s_i^k = H(s_i^{k-1})$), and sends values $G(s_i^1)$, $G(s_i^2), \ldots G(s_i^k), \ldots$ to the reader. The reader forwards the received messages to the back-end server which carries out

the identification of the appropriate tag. The procedure of identification is illustrated in Fig. 1.

Assume that X is a message received by the back-end server. The process of identification of the tag which has sent this message starts with an initial state of the first tag s_1^1 and generates the hash chain with states that the first tag sequentially traverses. A next node in the chain is obtained by applying H function on the previous node state. After each chain node is generated, G function is applied on it and such a value is compared with value X using comparator CMP. If these two values match, the tag which sent value X is identified as the first tag. If they are not equal, a next node in the chain is generated. Maximum length of the chain is m, where m is the number of possible accesses to each tag and it is predefined for a RFID system. If the first tag is not identified as the sender, the procedure is repeated for the second, third, etc. until the correct tag is identified. Success of identification is guaranteed.

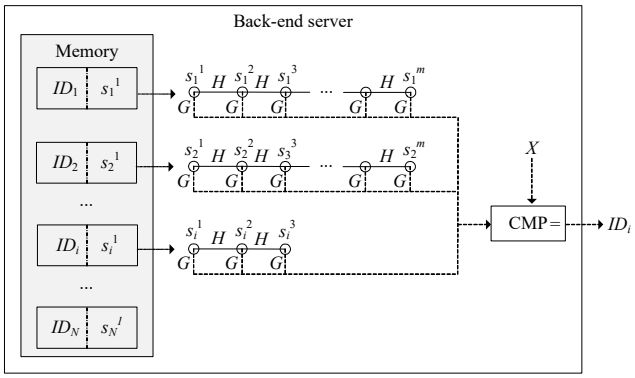

Fig. 1. Tag identification in OSK protocol.

Obviously, the identification algorithm is based on an exhaustive search and requires Nm calculations of G and H functions, what results in an enormous complexity.

III. Methods for Faster Identification

High complexity of tag identification prevents a wider use of OSK protocol. Consequently, it is predominantly used in RFID systems with a small number of possible tag accesses (e.g., $N = 2^{20}$ and $m = 128$). In order to overcome this problem, some methods are developed for improving the efficiency of identification, and they will be presented in the following.

A. TMTO-based method

This method is proposed by Avoine, Dysli and Oechslin in [10,11]. TMTO approach is a general method for improving the efficiency of some algorithms by balancing time and memory requirements. The basic idea was taken from Hellman who applied TMTO approach in cryptanalysis of DES block cipher system [12]. The essence of Hellman's idea is to chain keys/blocks in a random order using some encryption function in the preparatory phase. It ensures that in the attack phase the chains can be reconstructed from pairs of preserved starting and ending nodes of each chain. After the message block is received it is possible to find the chain to which the block belongs, and to identify the previous node in the chain which represents the key used to generate the message block. A detailed analysis of Hellman's chains can be found in [13].

A similar idea is applied in speeding up the tag identification in OSK protocol. The preparatory phase builds the chains with nodes representing the messages (resulted from applying G hash function on the current tag states) which tags send to the reader. The order of the messages in the chains is random, i.e., a chain can consist of messages of different tags. Only starting and ending nodes of the chains are preserved in memory of the back-end server. The process of identification relies on finding the received message in a chain. It is sufficient to determine the previous node which leads to the tag which sent the message.

In order to implement this idea it is necessary to define the relation between successive nodes, i.e., how a next node is derived from the previous one. To this end, R and F functions are adopted. Reduction function R uses the value from the previous node to obtain a pair of random values (p, q), where p is interpreted as tag's ordinal number, and q is current number of accesses to p-th tag. Using pair (p, q) F function calculates value $G(H^{q-1}(s_p^1))$, which represents one of the messages that the p-th tag can send and it is associated with a next node. The process of the next node generation is illustrated in Fig. 2.

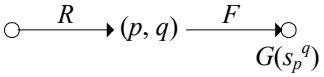

Fig. 2. Relation between two successive nodes in the chain.

Since the output of the reduction function is a random pair, the same message can appear in the chains more than once which can lead to an incorrect tag identification. This is a well-known problem of the false alarms in Hellman's approach which can be alleviated by the method of rainbow tables [14,15].

There follows a detailed description of the TMTO-based OSK protocol functioning. With respect to the basic variant, the tag initialization and tag identification phases are modified.

The initialization phase is augmented with some activities which provide some benefits during identification phase. It includes the following activities:

- generating n_c chains whose nodes correspond to messages which can be sent by the tags to the reader; the number of chains is $n_c = Nm/m_c$, where m_c is the maximum chain length (parameters N and m are predefined for a RFID system, and this length can be adjusted by varying parameter n_c); a new node in the chain is derived by successive application of the reduction function R and function F on the value from the previous node.
- saving the chains by storing pairs (SP_l, EP_l), $1 \le l \le n_c$, (SP_l represents a starting node, while EP_l is an ending node of the l-th chain) in memory of the back-end server; these pairs are the only output of the preparatory phase.

After initiatization, the RFID system components communicate between themselves in the same way as in the basic variant of OSK protocol. However, the procedure of

tag identification is now completely different and consists of the following activities:

- generating an online chain whose starting node corresponds to the received message (of the sender to be identified); subsequent nodes in the chain are generated in the same way as in the initialization phase (successively applying R and F functions)
- checking whether the current node is equal to the ending node of some chain stored in the initialization on the back-end server; this checking is carried out for every new node in the online chain after its creation; if the new node is not equal to the ending node of any chain, a next node of the online chain is created; if the check is successful the appropriate chain is found and it should be reconstructed now.
- reconstruction of the chain starts with the starting node from the found pair; then, a chain is generated until the node which is equal to the received message is found; applying the reduction function R on the previous node a pair of values (i, k) is obtained where parameter i represents the ordinal number of the tag which has sent the message, and its identifier is easily retrieved from memory.

It can be concluded that only one chain is traversed during tag identification, which means that identification is faster when the chains are shorter. However, in this case the number of chains is higher which requires more memory space for storing their starting and ending nodes. Feasibility of this technique depends on balancing between opposing time and memory requirements.

In this method, complexity of the preparatory phase is approximately $Nm^2/2$ (chains contain Nm nodes with $m/2$ hash calculations on the average). An additional memory on the back-end server for storing the starting and ending nodes is needed. However, complexity of identification is much lower than in the basic version of OSK protocol and can be calculated as $Nm^2/2n_c$ (Nm/n_c nodes in an online chain with $m/2$ hash calculations in average). Therefore, high complexity in the preparation is amortized if the tag accessing is quite intensive.

B. State table based method

The main drawback of the previous method is attributed to the adopted way of chaining, i.e., choice of R and F functions. The advantage of reduction function R is an ability to chain the states in a random order (one chain can contain states from different tags in different accesses). However, at the same time it is also a disadvantage since F function always starts calculation of $G(H^{j-1}(s_i^1))$ from the initial state of the tag selected by output of R function (parameter i in the appropriate pair). Therefore, R function makes chaining possible, but there is no dependency between the successive nodes in the chain. Demanding calculation of H function in this expression greatly influences the complexity of the entire method. An adversary fact is that both initialization and tag identification phases are affected.

More efficient calculation of expression $G(H^{j-1}(s_i^1))$ could improve the efficiency of the algorithm. If we know the state s_i^j during calculation of this expression (or if we can find it more easily), calculation of the expression would be much faster since $G(H^{j-1}(s_i^1)) = G(s_i^j)$, which implies only one calculation of G function. It guided us to propose employing a new look-up table T_s where equidistant tag states can be held. Now, state s_i^j can be found from the nearest state stored in T_s.

Implementation of this method with a table of equidistant states requires two modifications in the initialization phase:

- generation of the T_s state table
- modified process of chain generation.

When the T_s table is introduced, the basic problem is its size. This is where the second TMTO level is applied (the first one is balancing between length/number of chains and memory needed for their storing). Namely, speed-up of tag identification is higher if more states are stored in the table, but it requires a larger memory space.

In order to prepare a detailed description of the technique, some details necessary for its implementation should be explained.

In an RFID system with N tags and m tag accesses, the number of possible tag states is Nm, so $b = \log_2(Nm)$ bits are needed for encoding the state. Assume that a tag sends the message of d bits to the reader, where d must be large enough to ensure that searching the space of 2^d elements be physically infeasible (e.g., 128 bits or more).

Let T be a minimal perfect hash function which implements mapping T: $G(H^{j-1}(s_i^1)) \rightarrow p_k$, $0 \le p_k \le 2^b\text{-}1$ (T function maps d-bit values into b-bit values). Evidently, T function substitutes reduction function R in the previous algorithm. Consequently, F function has to be modified so it becomes F: $p_k \rightarrow G(H(s_k))$, where s_k is tag state that corresponds to value p_k. It can be regarded that value p_k consists of two components, p_e and p_w, of e and w bits, respectively, where $e+w=b$. Parameters e and w depend on the memory size available for storing T_s table. T_s table has 2^e entries, and each entry stores one tag state (d bits). Component p_w represents an offset which defines how many times hash function H has to be applied on the tag state which is stored in the p_e–th entry of the T_s table. Subsequent applying of H function results in state s_k which corresponds to p_k. According to the definition of function F, a new node in the chain is created after G function is applied on this state (Fig. 3). It can be seen that the number of calculations needed for creation of a new node is significantly lower than in the previous method.

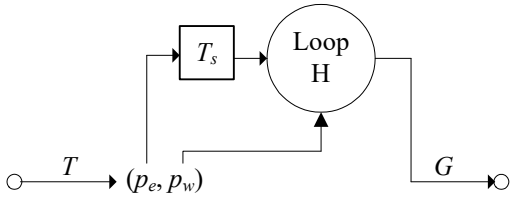

Fig. 3. Creation of the next node in the chain.

With respect to above mentioned, configuring an RFID system assumes setting the following parameters: N (the number of tags), m (the number of accesses to each tag), d (width of state/hash value in bits), e (parameter which determines the size of T_s table) and n_c (the number of chains). Parameter w is calculated as $\log_2(Nm)-e$.

The preparatory phase starts with setting the tag identifiers and randomly chosen d-bit values which represent the initial tag states. A pair of these values for each tag is kept in T_{ID} table. Hash function H is applied m times on an initial state of each tag. Every 2^w-th value is stored successively in the T_s table entries. After this table is completed, n_c chains are generated. A current node in the chain is created as follows:

- b-bit value p is obtained by applying T function on the value from a previous node
- e most significant bits (component p_e) are extracted from p, this value addresses a T_s entry to fetch state s
- hash function H is applied p_w times on s to obtain value s'
- $G(s')$ is calculated and the obtained result is used to set hash value in the created node.

For each chain, only a pair of hash values from starting and ending nodes are kept and stored in memory.

Tag identification proceeds as follows. When message x arrives at the back-end server, an online chain is generated whose starting node has the value x. Creation of each subsequent node begins with applying T function on the value from a previous node. Then, from the obtained result fields with e and w bits are extracted. Table T_s is addressed with e bits to read the state on which H function is applied p_w times. Finally, G function is applied. The resulted value is matched against all chain ending nodes. If there is no matching, the next node in the online chain is created. If the value is equal to the ending node of some chain, its corresponding starting node is fetched, and the chain is reconstructed until the node with value x is reached. T function is applied on the hash value from a node that preceeds x to obtain value p and to extract component p_e from it. Identifier of the tag which has sent value x is found in the entry $\lfloor 2^w p_e/m \rfloor$ of T_{ID} table.

The complexity of the preparatory phase depends on two processes: building of T_s table and chain generation. Nm calculations of the hash function H are needed to build the table. After that, this table can be exploited for more efficient chain generation, since only 2^{w-1} (instead of $m/2$) calculations of H on the average are needed for the creation of each new node in the chain. Now, overall complexity of the preparatory phase is $Nm\cdot(1+2^{w-1})$ calculations. Obtained speed-up is achieved at the expense of an additional memory on the back-end server for storing the table. This additional memory implies two opposing trends. On one side, complexity of the preparatory phase increases because of the calculation of all tag states in order to build T_s table. However, the complexity decreases because of more efficient generation of chains. Some experimental experience from section V shows that, for a small table size, the first trend prevails and the complexity increases, while, for a larger table size, the second trend is dominant and the overall complexity improves.

In the tag identification phase, Nm/n_c nodes are created in an online chain, where, as in the preparatory phase, on the average 2^{w-1} hash calculations are required per each node. Therefore, the overall complexity of the tag identification is $Nm\cdot 2^{w-1}/n_c$ calculations.

IV. Complexity of Methods

This section comparatively discusses the complexity of all three methods elaborated in this paper: basic OSK algorithm, method with TMTO approach (denoted further as TMTO-1) and method with state table (denoted further as TMTO-2).

As mentioned earlier, OSK algorithm doesn't require any preparatory phase for tag identification. However, TMTO-1 and TMTO-2 methods rely on the preparatory phase for producing data structures which are later used in tag identification. The practical feasibility of these methods is mainly determined by feasibility of the preparatory phase since the prepared data structures enable quite an acceptable time of tag identification.

If we assume that the back-end server can execute c calculations per second, then the durations of the preparatory phases in TMTO-2 and TMTO-1 methods can be calculated as $Nm\cdot(1+2^{w-1})/c$ and $Nm^2/(2c)$, respectively. Ratio of these values is $2\cdot(1+2^{w-1})/m \approx 2^w/m$. It follows that TMTO-2 method is better if the condition $2^w < m$ holds. Otherwise, TMTO-1 method is more efficient.

Time needed for the tag identification in OSK algorithm is Nm/c, in TMTO-1 method it is $Nm^2/(2cn_c)$, and in TMTO-2 method it is $Nm\cdot 2^{w-1}/(cn_c)$. It can be seen that, with respect to the basic OSK method, TMTO-1 method achieves identification speed-up of $2n_c/m$ times. Since in an ideal case the chains contain all messages that can be sent by tags distributed into n_c chains of mN/n_c length, the obtained speed-up is directly proportional to the number of tags. With more available memory, the chains become shorter so the efficiency of TMTO-1 method increases.

Speed-up of tag identification in TMTO-2 method with respect to the OSK algorithm is $2n_c/2^w$. This ratio is primarily determined by the size of the state table T_s. Since $Nm = 2^e 2^w$, the length of the chains in this case is $2^e 2^w/n_c$, speed-up is directly proportional to the chain length, and inversely proportional to the number of entries in T_s table.

The ratio of time durations of the tag identification phases in TMTO-2 and TMTO-1 methods is $2^w/m$. Evidently, this ratio is quite similar as in the preparatory phase. Therefore, the method which is more efficient in the preparatory phase for the given parameters of an RFID system has less complexity later during tag identification. The number of tag accesses m, as the basic parameter of a RFID system, is constant. Therefore, with an increase of the available memory size, parameter e also increases, so w decreases which implies a lower complexity, reducing the time of tag identification in TMTO-2 method.

V. Numerical Experiments

The performance of described methods will be estimated by experiments with some specific values of relevant parameters. These values are chosen to reflect contemporary trends in processor and memory technologies.

In all experiments, OSK-based RFID system is configured in such a way that parameters d and c have given values, while the values of other parameters (N, m, n_c and e) are varied in ranges which correspond to real conditions.

In order to disable finding the tag which has sent the message by exhaustive searching, a large value $d = 128$ bits was adopted. Also, it was assumed that parameter c has the value of 2^{24} calculations per second.

The number of RFID tags in the considered examples varies in the range from a thousand to a million, which is quite sufficient for common cases. Each tag can be accessed from 128 times (it is not much, but this case is frequently analyzed in the related work) to about million (it enables a very long period of system functioning). Since the total number of messages that tags can send is Nm, according to Hellman's recommendations from [12], in the experiments it was supposed that the number of chains is approximately $(Nm)^{1/2}$, optimizing the balance between the number of chains and their length. The value of parameter e is calculated from earlier expression $e = \log_2(Nm) - w$, where the offset between two equidistant states in T_s table is set to 16, which implies $w = 4$. This offset value is chosen to be relatively small (so as to enable full effects of TMTO-2 method), but not too small to prevent physical feasibility of this method by available amounts of memory.

Table 1 shows the durations of the preparatory phases for TMTO-1 and TMTO-2 methods (there is no such a phase in basic OSK method). It can be seen that TMTO-2 method attains significant speed-ups in all considered cases. Cases when the preparation time is excessively long so the preparation phase is not really feasible (a number of months or years) are denoted as "-" in the table. It is also evident that sometimes (e.g., *Ex*.3) TMTO-1 is not feasible while TMTO-2 is feasible for the same system configuration.

TABLE 1: DURATION OF THE PREPARATION PHASES

					Preparation time	
Ex.	***N***	***m***	***n_c***	***e***	***TMTO-1***	***TMTO-2***
1	2^{10}	2^{7}	2^{8}	13	0.5 s	0.125 s
2	2^{10}	2^{10}	2^{10}	16	32 s	1 s
3	2^{10}	2^{20}	2^{15}	26	-	17 min.
4	2^{15}	2^{7}	2^{11}	18	16 s	4 s
5	2^{15}	2^{10}	2^{13}	21	17 min.	32 s
6	2^{25}	2^{20}	2^{23}	41	-	-
7	2^{20}	2^{7}	2^{14}	23	8.5 min.	2.1 min.
8	2^{20}	2^{10}	2^{15}	26	9.1 h	17 min.
9	2^{20}	2^{20}	2^{20}	36	-	12.1 days

A comparative overview of the durations of the tag identification phases for all three considered techniques is shown in Table 2. Because of exhaustive searching, basic OSK method shows the worst performance, while TMTO-1 method searches over a limited state space resulting in a faster tag identification. The best results can be evidenced in TMTO-2 method which is feasible in all considered cases (even when the number of tags and the number of tag accesses are very high, *Ex*.6 and *Ex* 9, where two other methods are infeasible).

One reason for the best comparative performance of TMTO-2 method both in the preparatory and identification phases is an additional amount of memory for storing state table. For full understanding of the requirements, an analysis of the memory complexity is also necessary for all three methods.

Depending on a specific method, in the preparatory phase the following tables are produced and stored on the back-end server: T_{ID} for keeping pairs (tag identifier, initial tag state), T_c for keeping information about the chains in form of a pair per chain (starting node, ending node) and T_s for keeping equidistant states of tags. T_{ID} table has N entries with $2d$ bits per entry, T_c table has n_c entries with $2d$ bits per entry, and T_s table has 2^e entries with d bits per entry. During tag identification, basic OSK method uses only T_{ID} table, TMTO-1 method uses T_{ID} and T_c tables, while TMTO-2 method uses all three tables. For chosen parameter values, memory requirements of these methods are estimated in the previously considered examples, and obtained results are presented in Table 3.

TABLE 2: DURATION OF THE IDENTIFICATION PHASES

					Identification time		
Ex.	***N***	***m***	***n_c***	***e***	***OSK***	***TMTO-1***	***TMTO-2***
1	2^{10}	2^{7}	2^{8}	13	7.8ms	1.9ms	0.24ms
2	2^{10}	2^{10}	2^{10}	16	62.5ms	31.3ms	0.49ms
3	2^{10}	2^{20}	2^{15}	26	64s	-	15.6ms
4	2^{15}	2^{7}	2^{11}	18	250ms	7.8ms	0.97ms
5	2^{15}	2^{10}	2^{13}	21	2s	0.13ms	1.95ms
6	2^{25}	2^{20}	2^{23}	41	-	-	2s
7	2^{20}	2^{7}	2^{14}	23	8s	31.3ms	3.9ms
8	2^{20}	2^{10}	2^{15}	26	64s	1s	15.6ms
9	2^{20}	2^{20}	2^{20}	36	-	-	0.5s

TABLE 3: MEMORY REQUIREMENTS

Ex.	***OSK***	***TMTO-1***	***TMTO-2***
1	32KB	40KB	168KB
2	32KB	64KB	1.1MB
3	32KB	1.1MB	1GB
4	1MB	1.1MB	5.1MB
5	1MB	1.25MB	33.25MB
6	1MB	257MB	32TB
7	32MB	32,5MB	160,5MB
8	32MB	33MB	1.1GB
9	32MB	64MB	1TB

The results indicate that basic OSK method requires the smallest amount of memory because it doesn't include the preparation phase. Other two methods imply more demanding memory requirements, but mainly within contemporary available memory budgets in commercial computer systems. The only exceptions are cases *Ex*.6 and *Ex*.9 where an application of TMTO-2 method requires excessive amounts of memory (order of TB).

It can finally be concluded that nowadays the methods with TMTO approach involved in the tag identification phase can be successfully applied in OSK-based RFID systems with about million tags and million accesses to each tag, ensuring the preparation time of several minutes and identification time of the order of milliseconds.

I. Conclusion

As it was proven in the evaluation, the proposed method which uses a table with equidistant tag states (TMTO-2) guarantees the speed-up of both preparatory and tag identification phases with respect to the method with ordinary TMTO approach (TMTO-1). It is a consequence of fewer hash function calculations during creation of a new node in the chain. A detailed comparative performance analysis of these methods is performed in [16].

In order to make the performance analysis more complete, two implementation cases of the method with state table should be particularly considered:

I. when an additional amount of memory ΔM is used in TMTO-2 for storing of T_s table with respect to memory M used in TMTO-1 (more memory needed in TMTO-2)

II. when T_s table is held in the part M_s of memory M available in TMTO-1 (equal memory sizes in both methods)

The cases should be considered separately since the tag identification directly depends on the chain length which varies considerably in these conditions. Namely, if T_s table is stored into an additional memory space (case *I*), the chain lengths in both TMTO-1 and TMTO-2 methods are the same. However, if a part of system memory is used for storing table T_s (case *II*), the memory available for storing of chains decreases. Therefore, the number of chains is lower and their length is higher, which makes their searching during tag identification longer.

In case *I*, two opposing trends can be noticed: first, when the complexity of the preparatory phase is increased because of computing of all tag states by H hash function calculations, and second when the complexity of the preparatory phase decreases since there are less calculations in the creation of new nodes. In case *II*, the impact of M_s size on the tag identification phase time should be examined.

Because of applying the TMTO approach, the complexity analysis must consider both the preparatory and identification phases. Hence, physical feasibility of tag identification is directly affected by acceptable duration of the preparatory phase. Since the second TMTO level is included in both phases, there are some RFID system configurations which are operational with TMTO-2 method involved, but not practically operational with TMTO-1 method.

A relatively high number of parameters for configuring a RFID system (the number of tags, the number of accesses, message length, the number of chains, state table size, etc.) opens wide possibilities for different optimizations depending on importance for a particular intended use. Because of their mutual dependence (e.g., chain length depends on the number of tags and the number of tag accesses) it is not possible to achieve that all parameters have optimal values at the same time, and some trade-off should be established in order to attain an operational and efficient RFID system.

Following some indication from related literature, some future work should investigate the application of the rainbow table method together with state table based technique in striving to achieve a better performance.

References

[1] M.Kaur, M.Sandhu, N.Mohan, and P.Sandhu, "RFID Technology Principles, Advantages, Limitations & Its Applications", *Int. J. of Computer and Electrical Engineering*, vol. 3, no. 1, pp. 151-157, 2011.

[2] V.Tomašević, and M.Tomašević, "Time-Memory Trade-Off in RFID Systems", *Int. Scientific Conf. on ICT and E-Business Related Research - Sinteza*, Belgrade, Singidunum University, Serbia, pp 124-130, 2016.

[3] J.Chang, and H.Wu, "On Constant-Time-Identification and Privacy-Preserving RFID Protocols: Trade-Off between Time and Memory", *J. of Information Science & Engineering*, vol. 32, no. 4, pp. 887-901, 2016.

[4] K.Jung, and S.Lee, "A systematic review of RFID applications and diffusion: key areas and public policy issues", *J. of Open Innovation: Technology, Market, and Complexity*, Springer Open, 2015.

[5] G.Liu, H.Zhang, F.Kong, and L.Zhang. (2018). "A Novel Authentication Management RFID Protocol Based on Elliptic Curve Cryptography", *Wireless Personal Communications*, vol. 101, no. 3, pp. 1445-1455, 2018.

[6] X.Zhuang, Y.Zhu, C.Chang, and Q.Peng, (2018). "Security Issues in Ultralightweight RFID Authentication Protocols", *Wireless Personal Communications*, vol. 98, no. 1, pp. 779-814, 2018.

[7] M.Ohkubo, K.Suzuki, and S.Kinoshita, "Cryptographic Approach to "Privacy Friendly" Tags", *RFID Privacy Workshop*, MIT, USA, 2003.

[8] I.Syamsuddina, T.Dillonb, E.Changc, and S.Han, "A Survey of RFID Authentication Protocols Based on Hash-Chain Method", *Int. Conf. on Convergence and Hybrid Information Technology*, Busan, South Korea, 2008.

[9] L.Yang, P.Yu, W.Bailing, Q.Yun, B.Xuefeng, Y.Xinling, and Y.Zelong, "Hash-based RFID Mutual Authentication Protocol", *Int. J. of Security and its Applications*, vol. 7, no. 3, pp. 183-194, 2013.

[10] G.Avoine, and P.Oechslin, "A Scalable and Provably Secure Hash-Based RFID Protocol", *IEEE Int. Conf. On Pervasive Computing and Communications*, Hawaii, USA, pp.110-114, 2005.

[11] G.Avoine, A.Bingol, X.Carpent, and S.Yalcin, "Privacy-friendly Authentication in RFID Systems: On Sub-linear Protocols based on Symmetric-key Cryptography", *IEEE Trans. on Mobile Computing*, Issue 99, 2012.

[12] M.Hellman, "A Cryptanalytic Time-Memory Trade-Off", *IEEE Trans. on Information Theory*, IT-26, no. 4, pp. 401-406, 1980.

[13] V.Tomašević, and M.Tomašević, "An Analysis of Chain Characteristics in the Cryptanalytic TMTO Method", *Theoretical Computer Science*, vol. 501, pp. 52-61, 2013.

[14] P.Oechslin, "Making a Faster Cryptanalytic Time-Memory Trade-Off. Advances in Cryptology", 2729 of LNCS, pp. 617-630, 2003.

[15] G.Avoine, A.Bourgeois, and X.Carpent, "Analysis of rainbow tables with fingerprints", *Inf. Security and Privacy*, Springer, vol. 9144 of LNCS, pp. 356-374, 2015.

[16] V.Tomašević, and M.Tomašević, "Double Time-Memory Trade-Off in OSK RFID Protocol", *Wireless Personal Communications*, Springer, https://doi.org/10.1007/s11277-019-06417-8, p.1-18, April 2019.

[17] V. Tomašević and M. Tomašević, "Speeding Up Tag Identification in OSK RFID Protocol," (in Serbian) *2018 26th Telecommunications Forum (TELFOR)*, Belgrade, 2018, pp. 1-4.