Variant Constructions for TMTO based on Random Mapping Statistics

Nurdan SARAN, Ali DOĞANAKSOY

***Abstract*—The time memory trade-off (TMTO) is a generic method, introduced by Hellman, for quickly inverting a one-way function. In this paper we present some properties of random mappings in terms of TMTO attacks. We also propose new variants of the technique. Firstly, we present a technique to construct Perfect Hellman tables by using random mapping properties. Finally, we give a variant distinguished method (DP) method which has a high success rate for random permutations.**

***Keywords* —Cryptanalysis, Time Memory Trade Off Method, Random Mappings Statistics.**

I. Introduction

One-way functions exist in almost all areas of cryptography. Securities of encryption schemes, authentication mechanisms, and various other cryptographic protocols depend on the hardness of inverting one-way functions which they are based on. By inversion of a one-way function, we mean a process which finds a pre-image of a given point, if there exists any. Exhaustive search is an obvious method to find a pre-image of a given point which is trying all possible inputs and checking whether they yield the given value. If it is the case that the search operation is expected to be done for different points in the future -a more likely case to happen in real life-, the time required by exhaustive search may turn out to be excessive. Constructing a lookup table containing pre-images of all points may be another solution. In this situation, it requires extreme amounts of memory for the functions acting on large sets. Balancing the gap between the solution time and the required memory, in other words trading memory for time, is important. Hellman[1] proposes a probabilistic method which suggests a tradeoff between time and memory by storing some pre-computed data in the memory. Time Memory Trade Off (TMTO) has a lower time complexity (in online phase) than exhaustive search and a lower memory complexity than lookup table. If the attacker has a large memory, the computation time will be less.

There are mainly two improvements on TMTOs. Following Hellman's work on DES, Rivest introduces the concept of distinguished points(DP) which reduces the number of table lookups. In his dissertation [2], Borst gives expressions on success probability and complexity for distinguished points. [3] Standaert et al claim that they corrected the probability of success for distinguished points to correspond with practical implementations. Efficient mask functions are proposed and experimentally confirmed and used to build a FPGA design to perform realistic tradeoffs against the block cipher DES. The second improvement is called as Rainbow Tables. Oechslin [4] proposes Rainbow Tables in which a successive reduction function is used for each column in the chain, rather than constructing r tables with different reduction functions. There are t reduction functions; one for each column. In [5], Barkan et al. give the upper bound for coverage and lower bound for worst case time complexity when the structures of f is not used and describe some variants of rainbow tables. In [6] Hong and Sarkar apply TMTO on various block cipher modes of operations and they show how to apply multiple data TMTO to both CBC and CFB modes of operations. In addition, they show that hash function's preimage resistance is not equal to its digest length. Hong et al. [7] propose a variant of the DP technique, named variable DP (VDP), and show how to combine distinguished points with rainbow approach.

Babbage [8] and Golic [9] independently apply TMTOs on stream cipher. Application of TMTO on stream ciphers is different than the application of TMTO on block ciphers. When a stream cipher is taken into account, there are two types of attack, in terms of the one-way function the attacker tries to invert. The aim of the attacker might be inverting the function from secret key(k bit) to output prefix(k bit) [6] or inverting the function from internal state(s bit) to output prefix(s bit)[8]-[10]. If an attacker can find any of the actual states then he can find the rest of the keystream. Thus, he does not have to find the initial state (or the key). Moreover, he may find the initial state by using reverse engineering methods in many cases. Biryukov and Shamir [10] generalize TMTOs to multiple data and call Time/Memory/Data Trade Offs(TMDT) for stream ciphers. When D bits of keystream is given (there is no difference between known plaintext and chosen plaintext attack when a stream cipher is taken into account), $D-log(N)+1$ overlapping prefixes can be obtained. In the online stage, when D bits are given, it is aimed to find the pre-image of any one of these prefixes. Total memory is reduced from mt to mt/D. Since Hellman's table for block ciphers is constructed for a particular plaintext block, the multiple prefix is useless. On the other hand, precomputed table can be used on multiple prefixes for stream ciphers. [11]-[12] introduce time/memory/key trade-off(TMK) attack on block ciphers, Hellman's algorithm is generalized for the case of multiple data. Distinguished points on stream ciphers are studied by [10]-[13]. Distinguished points method is applied on stream ciphers, and it is called BSW sampling. Biryukov, Shamir and Wagner [13] improve Golic's attack in which the key is computed in about one second during the first two

N. Saran is with the Department of Computer Engineering, Cankaya University, Ankara, e-mail: buz@cankaya.edu.tr

A.Doğanaksoy is with the Department of Mathematics, METU, email: aldoks@metu.edu.tr.

minutes of the conversation on a single PC. Dunkelman and Keller [14] show the treatment of IV in TMTOs and claim that setting the IV length to be equal to the key length, k, does not ensure a k-bit security against TMTO attacks. For each chosen IV, the attacker prepares a Hellman table (or Rainbow table) to invert the function from key to the output prefix. Therefore, the resistance to TMTO attacks is considered to be an important criteria when designing a stream cipher.

Random mappings are functions from a finite set of n elements onto m elements. They are widely used in combinatorial problems. Various characteristics of random mappings have been studied in [15], [16], [17], [18]. Random mapping statistics have a great importance place when constructing TMTOs table. Our main question in this study is "Is it possible to have a high success rate of TMTO by using the properties of random mappings?".

In the following section, we briefly summarize Hellman's construction, and the two main improvements of the attack. Then we calculate the expected values of two of the random mapping properties, and summarize the results in the literature. Finally, we give new constructions for TMTOs based on random mapping statistics.

II. Preliminaries

A. Hellman's Construction

Hellman introduced a TMTO attack and applied on the block cipher, DES. A TMTO application is composed of two phases; an offline (precomputation) phase and online phase.

Offline Phase: Starting at random m points, chains of length t are generated by iteratively evaluating $f = E \circ R$, E represents for encryption function and R represents for reduction function. $m \times (t+1)$ matrix is formed which is called Hellman table. By this way, k tables are computed in which each table has different R so if two entries in two tables are the same, usage of different functions will result in different elements in the next chain item. To save memory we store only the first and last element of a chain. By knowing the starting point, the successive elements can be recalculated in the chain. This process is expected to be equivalent to the exhaustive search.

Online Phase: For a preimage of a given point c_0, it is tried to find out if the key that is used to generate c_0 is among the one used in the generated table. To do so, starting by checking if c_0 is equal to any points of the last column (any endpoints),it is tested to find the preimage by computing the previous entry. If c_0 is not equal to any points of the last column, $f(c_0)$ is calculated and checked for a match in the $(t-1)^{st}$ column. f is applied repeatedly until c_0 is in the second column of the matrix. It should be noted that finding a match doesn't imply that the desired key is found, it may be a false alarm.

B. Rainbow Tables

Oechslin suggested to use Rainbow tables of size $mt \times t$ in which different reduction functions are applied for each iteration, instead of having t different tables. Generating a table without merges is another trade because it will increase the precomputation time. Perfect tables are tables in which there is no merges. Firstly more chains are generated since merging chains will have the identical endpoints, they can be easily discarded (in both rainbow and DP tables). In [19] Avoine . stated that to construct a perfect Hellman table is not efficient since all chains have to be looked up. We give in Section IV-A a new construction technique that constructs Perfect Hellman Tables .

C. DP Tables

Rivest suggested to use distinguished points as endpoints for the chains to reduce the number of memory accesses. In distinguished points, the chain is terminated whenever a distinguished property is obtained, instead of generating fixed length chains. Only endpoints which encountered a distinguished point in less than t iterations will be stored otherwise they will be discarded. Chains are generated until a distinguished point is reached and the triple $(SP, EP, lengthofchain)$ is stored. Otherwise until t_{max} iterations if a distinguished point is not reached , that chain is discarded. Moreover if the chain length is less than some t, say t_{min} that chain is also discarded. If same distinguished point occurs in different chains, the triple with maximum chain length is stored (since merging chains will have the same endpoint). It is clear that the chain lengths are variable. This improvement reduces the memory access in the online phase, since the ciphertext is compared to the end points only if it is a distinguished point.

III. Random Mapping Statistics

Let $\mathcal{X}$ denote the finite domain of size n and $\mathcal{F}_n$ denote the collection of all functions (f) from domain $\mathcal{X}$ into range $\mathcal{X}$. We will consider only random mappings model where every function from $\mathcal{F}_n$ is equally likely to be chosen. So the sample space consists of n^n random mappings in other words the probability that a particular function(f) from $\mathcal{F}_n$ is chosen will be $\frac{1}{n^n}$.

Starting from a point $x_0 \in \mathcal{X}$ and iteratively applying f, the following sequence is obtained;

$$\{x_0, f(x_0), f^2(x_0), \ldots\}. \tag{1}$$

The k-th iteration of f on $\mathcal{X}$, where $0 \leq k \leq n$ will be $f^k(x_0) = f(f^{k-1}(x_0))$ where $f^0(x_0) = x_0$. For some $k \geq 0$ if $f^k(x_0) = y$ then we will call y to be a k th image of x_0 in f. For some $k \leq 0$ if $f^k(x_0) = y$ then we will call y to be a k th inverse of x_0 in f. For $k < 0, f^k(x_0)$ may not exist, which we will call *terminal nodes*, (in other words, a node may have no inverse image) or may not be uniquely determined (more than one inverse which will cause a *false alarm* from a view point of TMTO). Each random mapping, f can be represented by a functional graph.

A. Graph Representation of Functions

A functional graph of a function $f : \mathcal{X} \to \mathcal{X}$ is a directed graph whose nodes are the elements of $\mathcal{X}$ and whose edges are the ordered pairs $(x, f(x))$, for all $x \in \mathcal{X}$.

In Figure 1 , the typical behavior of an iteration operation is given. Since the set $\mathcal{X}$ is finite, after some iterations, we

Fig. 1: Functional Graph

will encounter a point that has occurred before. Let $f^m(x_0)$, $0 \leq m \leq n-1$ be the point that the iteration enters a loop. Then $f^m(x_0) = f^k(f^m(x_0))$, k is the smallest positive integer which we call the *cycle length*. The path between x_0 and $f^m(x_0)$ is called the *tail length*. The sum of the tail length and cycle length is defined as the *ρ-length*.

The statistical behaviors of random mappings (See [17], [20], [16], [18], [21], [15]) are summarized as in the followings:

Theorem 1: The probability distributions for random functional graph of size n are:

(i) Number of Components

$$Pr(\#components = k) = \frac{n!}{k!n^n} \sum_{\mu=k}^{n} \mathcal{S}_\mu^k \sum_{k_1,\ldots k_n} \frac{1}{k_1!\ldots k_n!} \left(\frac{1}{1!}\right)^{k_1} \cdots \left(\frac{n^n}{n!}\right)^{k_n} \quad (2)$$

where $\mathcal{S}_\mu^k$ are Stirling's Numbers of the First Kind and sum over all choices of $k_1, \ldots, k_n$ with $k_i > 0$ $(i = 1, \ldots, \mu)$ and $\sum_{i=1}^{n} k_i = \mu, \sum_{i=1}^{n} ik_i = n$

(ii) ρ-length

$$Pr(\rho - length = k) = \frac{(n-1)!}{(n-k)!}\frac{k}{n^k} \quad (3)$$

(iii) Cycle-length

$$Pr(cycle - length = k) = \sum_{j=k}^{n} \frac{(n-1)!}{(n-j)!n^j}. \quad (4)$$

Although the expected values of random mappings are widely studied in literature, we calculate the expected values for ρ length and cycle length and resulting tail length with a different approach.

1) ρ-length:

$$\begin{aligned}
&E(\rho - length = k)\\
&= \sum_{k=0}^{n} \frac{(n-1)!}{(n-k)!}\frac{k^2}{n^k} \text{ by writing } k \to n-k\\
&= \frac{(n-1)!}{n^n} \sum_{k=0}^{n} \frac{(n-k)^2}{k!} n^k\\
&= \frac{(n-1)!}{n^n}\left[n^2 \sum_{k=0}^{n} \frac{n^k}{k!} - 2n \sum_{k=0}^{n} \frac{n^k k}{k!} + \sum_{k=0}^{n} \frac{k^2 n^k}{k!}\right]
\end{aligned}$$

Let $Y_1 = \sum_{k=0}^{n+1} \frac{n^k}{k!}$, $Y_2 = \sum_{k=0}^{n+1} \frac{n^k k}{k!}$, $Y_3 = \sum_{k=0}^{n+1} \frac{n^k k^2}{k!}$

and define

$F(x) = \frac{e^{nx}}{2}$ thus $F(x) = \sum_{k=0}^{\infty} \frac{n^k x^k}{k!}$ then $Y_1 \approx F(1)$

$F'(x) = \frac{ne^{nx}}{2}$ thus $F'(x) = \sum_{k=0}^{\infty} \frac{kn^k x^{k-1}}{k!}$ then $Y_2 \approx F'(1)$

$F''(x) = \frac{n^2 e^{nx}}{2}$ thus $F''(x) = \sum_{k=0}^{\infty} \frac{k^2 n^k x^{k-2}}{k!} - \frac{kn^k}{k!}$ then $Y_3 \approx F''(1) + F'(1)$

$$\begin{aligned}
E(k) &= \frac{n!}{n^n}\left[\left(n^2\right)e^n - 2n(ne^n) + n^2 e^n + ne^n\right]\\
&\cong \frac{n!}{n^n}\frac{e^n}{2} = \sqrt{\frac{\pi n}{2}}
\end{aligned}$$

2) cycle-length :

$$\begin{aligned}
&E(cycle - length = k)\\
&= \sum_{k=0}^{n}\sum_{j=k}^{n} \frac{(n-1)!}{(n-j)!n^j} k\\
&= \sum_{j=1}^{n} \frac{(n-1)!}{(n-j)!n^j} + 2\sum_{j=2}^{n} \frac{(n-1)!}{(n-j)!n^j} + \ldots + \frac{(n-1)!}{n^n}\\
&= \frac{1}{2}\sum_{j=0}^{n} \frac{(n-1)!(j+1)j}{(n-j)!n^j}\\
&= \frac{(n-1)!}{2}\sum_{j=0}^{n} \frac{(j+1)j}{(n-j)!n^j} \text{ by writing } j \to n-j\\
&= \frac{(n-1)!}{2n^n}\sum_{j=0}^{n} \frac{(n-j)(n-j+1)}{j!n^{-j}}\\
&= \frac{(n-1)!}{2n^n}\left[(n^2+n)\sum_{j=0}^{n} \frac{n^j}{j!} - (2n+1)\sum_{j=0}^{n} \frac{n^j j}{j!} + \sum_{j=0}^{n} \frac{j^2 n^j}{j!}\right]\\
&= \frac{(n)!}{2n^n}\frac{e^n}{2} = \sqrt{\frac{\Pi n}{8}}
\end{aligned}$$

Since $E(\rho - length) = E(cycle - length) + E(tail - length)$, then $E(cycle - length = k) = \sqrt{\frac{\Pi n}{8}}$

Theorem 2: The expected values for a random mapping are [20]-[17]

(i) Number of components $= \frac{ln(n)}{2}$

(i) Number of terminal nodes $= e^{-1}n \approx 0,3679n$

(ii) Number of image nodes $= (1 - e^{-1})n \approx 0,6321n$

(iii) Average cycle length $= \sqrt{\frac{\pi n}{8}} \approx 0,6267\sqrt{n}$

(iv) Average tail length $= \sqrt{\frac{\pi n}{8}} \approx 0,6267\sqrt{n}$

(v) Averageρlength $= \sqrt{\frac{\pi n}{2}} \approx 1,2533\sqrt{n}$

(vi) Average component size $= \frac{2n}{3}$

(vii) Maximum cycle length $= 0.78248\sqrt{n}$
(viii) Maximum tail length $= 1.73746\sqrt{n}$

B. Random Permutations

Permutations have the unary functional graphs that the sum of in-degrees must be the same as the sum of the out-degrees. Each node must have exactly one inverse. There are no terminal nodes and tail nodes. This also shows that every node is cyclic.

Let $\alpha(n)$ be the number of cycles in a permutation. Then we have

$$Pr(\alpha(n) = j) = \frac{S_n^j}{n!}$$

where $S_n^j, 1 \leq j \leq n$ is a Stirling number of first kind [18].

Let L_n be the expected length of the longest cycle in a random permutation then

$$lim_{n\to\infty}(L_n/n) = 0.62432965$$

Theorem 3: The expected values for a random permutation are

(i) Number of components $= \sum_{i=1}^{n} \frac{1}{i}$
(ii) Number of terminal nodes $= 0$
(iii) Number of image nodes $= n$
(iv) Average cycle length $= \frac{n+1}{2}$
(v) Average tail length $= 0$
(vi) Average ρ length $= \frac{n+1}{2}$
(vii) Maximum cycle length $= 0.62432965n$

It should be noted that the average cycle length as seen from a random node, u, is as in the above theorem.

However the average cycle length of a mapping is

$$\text{Average cycle length} = \frac{n}{\#components} = \frac{n}{lnn}.$$

Since #components $\approx lnn$

IV. Variant Constructions

A. Perfect Hellman

Assuming the encryption function is modeled as a random mapping the set into itself, if we consider the graph representation of a random mapping, we choose our start points as terminal nodes (we mean the points which have no preimage). It is clear that to find the terminal nodes, it is necessary to increase precomputation from N to $2N$. Since there are $N' = N * e^{-1}$ terminal nodes, starting from N' by iteratively applying f we construct a Hellman table. In [19], starting from m_0 random points, maximum number of perfect chains of length t is given as

$$m(t, m_0) = \frac{m_0}{1 + \frac{tm_0}{2N}}$$

and maximum number of chains of length t by starting with m_0 equal to N is given as

$$m_{max} = \frac{2N}{t+2}$$

Starting from N' points, maximum number of chains of length t of Perfect Hellman table will be $m_{max} \approx \frac{2N}{t+6}$.

This shows us that the complexity of Perfect Hellman table will approximately be equal to the complexity of Perfect Rainbow table.

B. Variant Distinguished Point Table

Our main aim is to construct a table from distinguished point to distinguished point. Let D be the number of distinguished points. We construct a table of size, D. To do this, we first list distinguished points as starting points. Then, we iteratively apply f until a distinguished point is reached. By constructing a table in this manner, we have $\approx$ %99 success rate for a random permutation. Average cycle length for a random permutation is $\approx \frac{n}{lnn}$. The memory complexity of the attack is equal to the number of distinguished points.

For a random mapping, we can construct the same table. If there is no distinguished point in the chain, then the chain will enter in an infinite loop. In order to prevent this undesired situation, we iterate until a predefined time, t, then the chain is discarded. (The reader should note that t will be smaller than the average cycle length of a random mapping).

V. Conclusion

Resistance against TMTO attacks is an important criteria for designing a cipher. In this paper, we summarize the random mapping statistics in terms of TMTO attacks. We also give the expected values of cycle length, ρ length and $tail$ length of a random mapping. Then some properties of random permutations are given. We present some variant constructions of TMTOs depending on the random mapping statistics. Perfect Hellman tables are constructed by choosing the starting points from the terminal nodes of a random mapping. A table from distinguished point to distinguished point is constructed without restrictions of time, t for random permutations. Variant distinguished point method can also be applied on random mappings.

References

[1] M. E. Hellman, "A cryptanalytic time-memory trade-off." *IEEE Transactions on Information Theory*, vol. 26, no. 4, pp. 401–406, 1980.
[2] J. Borst, "Block ciphers: Design, analysis and side-channel analysis," Ph.D. dissertation, Katholieke Universiteit Leuven, 2001.
[3] F.-X. Standaert, G. Rouvroy, J.-J. Quisquater, and J.-D. Legat, "A time-memory tradeoff using distinguished points: New analysis & fpga results," in *CHES*, ser. Lecture Notes in Computer Science, B. S. K. Jr., Çetin Kaya Koç, and C. Paar, Eds., vol. 2523. Springer, 2002, pp. 593–609.
[4] P. Oechslin, "Making a faster cryptanalytic time-memory trade-off," in *CRYPTO*, ser. Lecture Notes in Computer Science, D. Boneh, Ed., vol. 2729. Springer, 2003, pp. 617–630.
[5] E. Barkan, E. Biham, and A. Shamir, "Rigorous bounds on cryptanalytic time/memory tradeoffs," in *CRYPTO*, ser. Lecture Notes in Computer Science, C. Dwork, Ed., vol. 4117. Springer, 2006, pp. 1–21.

[6] J. Hong and P. Sarkar, "New applications of time memory data tradeoffs," in *ASIACRYPT*, ser. Lecture Notes in Computer Science, B. K. Roy, Ed., vol. 3788. Springer, 2005, pp. 353–372.
[7] J. Hong, K. C. Jeong, E. Y. Kwon, I.-S. Lee, and D. Ma, "Variants of the distinguished point method for cryptanalytic time memory trade-offs," in *ISPEC*, ser. Lecture Notes in Computer Science, L. Chen, Y. Mu, and W. Susilo, Eds., vol. 4991. Springer, 2008, pp. 131–145.
[8] S. Babbage, "Improved "'exhaustive search"' attacks on stream ciphers." in *ECOS 95 (European Convention on Security and Detection)*, no. 408 in IEE Conference Publication, May 1995.
[9] J. D. Golic, "Cryptanalysis of alleged a5 stream cipher." in *EUROCRYPT*, 1997, pp. 239–255.
[10] A. Biryukov and A. Shamir, "Cryptanalytic time/memory/data tradeoffs for stream ciphers," in *ASIACRYPT*, ser. Lecture Notes in Computer Science, T. Okamoto, Ed., vol. 1976. Springer, 2000, pp. 1–13.
[11] A. Biryukov, S. Mukhopadhyay, and P. Sarkar, "Improved time-memory trade-offs with multiple data," in *Selected Areas in Cryptography*, ser. Lecture Notes in Computer Science, B. Preneel and S. E. Tavares, Eds., vol. 3897. Springer, 2005, pp. 110–127.
[12] S. Mukhopadhyay and P. Sarkar, "Application of lfsrs in time/memory trade-off cryptanalysis," in *WISA*, ser. Lecture Notes in Computer Science, J. Song, T. Kwon, and M. Yung, Eds., vol. 3786. Springer, 2005, pp. 25–37.
[13] A. Biryukov, A. Shamir, and D. Wagner, "Real time cryptanalysis of a5/1 on a pc," in *FSE*, ser. Lecture Notes in Computer Science, B. Schneier, Ed., vol. 1978. Springer, 2000, pp. 1–18.
[14] O. Dunkelman and N. Keller, "Treatment of the initial value in time-memory-data tradeoff attacks on stream ciphers," *Inf. Process. Lett.*, vol. 107, no. 5, pp. 133–137, 2008.
[15] J. Arney and E. A. Bender, "Random mappings with constraints on coalescence and number of origins." *Pacific J. Math.*, vol. 103, no. 2, pp. 269–294, 1982.
[16] D. E. Knuth, *Art of Computer Programming, Volume 2: Seminumerical Algorithms (3rd Edition)*. Addison-Wesley Professional, November 1997. [Online]. Available: http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20\&path=ASIN/0201896842
[17] B. Harris, "Probability distributions related to random mappings," *The Annals of Mathematical Statistics*, vol. 31, no. 4, pp. 1045–1062, 1960.
[18] V. F. Kolchin, *Random Mappings*. Springer ,-Verlag, 1986.
[19] G. Avoine, P. Junod, and P. Oechslin, "Time-memory trade-offs: False alarm detection using checkpoints," in *INDOCRYPT*, ser. Lecture Notes in Computer Science, S. Maitra, C. E. V. Madhavan, and R. Venkatesan, Eds., vol. 3797. Springer, 2005, pp. 183–196.
[20] P. Flajolet and A. M. Odlyzko, "Random mapping statistics." in *EUROCRYPT*, 1989, pp. 329–354.
[21] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. Boca Raton, Florida: CRC Press, 1996, uRL: http://cacr.math.uwaterloo.ca/hac.