Bull. Korean Math. Soc. **53** (2016), No. 5, pp. 1439–1446
http://dx.doi.org/10.4134/BKMS.b150754
pISSN: 1015-8634 / eISSN: 2234-3016

PERFORMANCE COMPARISON OF CRYPTANALYTIC TIME MEMORY DATA TRADEOFF METHODS

Jin Hong and Byoung-Il Kim

Abstract. The execution complexities of the major time memory data tradeoff methods are analyzed in this paper. The multi-target tradeoffs covered are the classical Hellman, distinguished point, and fuzzy rainbow methods, both in their non-perfect and perfect table versions for the latter two methods. We show that their computational complexities are identical to those of the corresponding single-target methods executed under certain matching parameters and conclude that the perfect table fuzzy rainbow tradeoff method is most preferable.

1. Introduction

Cryptanalytic time memory tradeoff algorithms are methods for quickly inverting one-way functions and these are widely used in practice to extract passwords from password hashes. A typical tradeoff method first executes a pre-computation phase to produce large tables, and these pre-computed tables are later utilized by the online phase, which is the attempt to recover the input corresponding to each given inversion target. The classical Hellman method [11] was the first of such methods, and, according to [9], Rivest introduced the distinguished point (DP) method [7, 8] as a variant with a reduced table lookup frequency. The rainbow method [18] is currently the most widely used such method.

The subject of this paper is a closely related technique, often referred to as the time memory *data* tradeoff [5], where the objective is to recover the input corresponding to any *one* of *multiple* inversion targets. Such a setting fits naturally with attacks on streamciphers [1, 10], as can be witnessed by its practical application [6] on the A5/1 encryption algorithm for GSM phones.

Received September 15, 2015.
2010 *Mathematics Subject Classification.* Primary 68W40, 94A60.
Key words and phrases. time memory data tradeoff, multi-target tradeoff, Hellman, distinguished point, fuzzy rainbow, cryptography.
This work was supported by the Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2012R1A1B4003379). Part of this work was done while J. Hong was visiting Department of Mathematics, U. C. Davis, and he is grateful for their hospitality.

The first multi-target tradeoff methods [5, 6] were straightforward multi-target adaptations of the single-target Hellman and DP tradeoffs. A direct adaptation of the rainbow method to the multi-target setting is known [4] to be vastly inferior to the other two methods, and the fuzzy rainbow tradeoff [2, 3] was designed to be a multi-target rainbow tradeoff variant of comparable performance. This method was also independently invented to be the core component of a fully functional attack on GSM phones [16, 17].

Both the DP and fuzzy rainbow tradeoff methods have non-perfect table and perfect table sub-versions. These four tradeoff methods and the classical Hellman method may be taken as the major multi-target tradeoff methods. Heuristic arguments can be used to show that the five tradeoff methods, viewed as multi-target tradeoff algorithms, perform comparably in the asymptotic sense. In this work, we provide the first theoretical treatment of the execution complexities of the multi-target tradeoffs that does not hide any small constant factors. This information is crucial in practical comparisons of the methods and allows for educated decisions in choosing one method over another.

This work relies heavily on the recent analyses [12, 13, 14, 15] of the five methods that were carried out in the single-target setting. Based on an in-depth understanding of these previous results, we extract just the core arguments and cleverly adjust them to be applicable to the multi-target setting, treating all five methods simultaneously. We come to the conclusion that, when placed in the right perspective, the previous performance comparisons made of these methods in the single-target setting can be understood to be valid even in the multi-target setting.

The remainder of this paper is organized as follows. In Section 2, we show how the Hellman and DP methods may roughly be seen as degenerate cases of the fuzzy rainbow method, so that all methods can be treated in a uniform manner. The complexity analysis of the multi-target tradeoffs is given in Section 3, and the work is summarized in Section 4.

2. Preliminaries

In this section, we will present the five tradeoff methods in a manner that hides all the complicated details. This will allow us to see that it suffices to treat just one algorithm in order to cover all five tradeoff methods. We assume that the reader is familiar with the basic algorithms, at least in their single-target versions. In particular, we assume that the reader is aware of the pre-computation matrix structures of the classical Hellman, distinguished point (DP), and fuzzy rainbow tradeoff methods in their non-perfect and perfect table versions. However, we will put in effort to clarify the more obscure aspects of the online phase algorithms.

Standard notation for algorithm parameters, such as m, t, ℓ, and s, will be used. The reader can refer to the beginning sections of [12, 13, 14, 15] for a streamlined review of the notation and details of the single-target tradeoff

methods. We will use D to denote the number of targets given to a multi-target tradeoff method.

Algorithm 1: Online phase of Hellman and DP methods

```
for j = 1 to ℓ do
    for k = 1 to D do
        generate the online chain associated with the k-th target for the
        j-th table;
        resolve alarm whenever encountered;
        terminate whenever answer is found;
    end
end
```

Algorithm 2: Online phase of fuzzy rainbow method

```
for i = s to 1 do
    for j = 1 to ℓ do
        for k = 1 to D do
            generate the online chain associated with the k-th target for
            the j-th table that starts from the i-color;
            resolve alarm if encountered;
            terminate if answer is found;
        end
    end
end
```

The only difference between the pre-computation phases of a single-target tradeoff method and its multi-target version is in the rough order of ℓ, the number of tables. The online phases of the classical Hellman method and the non-perfect and perfect table versions of the DP method may roughly be presented as Algorithm 1, and the same for the non-perfect and perfect fuzzy rainbow methods are given by Algorithm 2. The explicit operations done within the inner-most loops of Algorithm 1 and Algorithm 2 are actually more structured than seen here and varies among the different tradeoff methods, but these details will not be important for this paper. The deliberate obscuring of these inner details is crucial in making the uniform approach of this paper possible.

Setting $D = 1$ in Algorithm 1 and Algorithm 2 essentially removes the inner-most loop and reduces these algorithms to the single-target versions of the various tradeoff methods' online phase algorithms. Some parts of this paper may be easier to understand if each algorithm is viewed as a family of algorithms, with each value of integer parameter D corresponding to a different

algorithm. Each single-target tradeoff algorithm is then just a specific instance of the corresponding family of tradeoff algorithms.

Notice that, in Algorithm 2, the j-loop, corresponding to different pre-computation tables, is placed inside the i-loop, corresponding to the starting color for the online chain. As with the original rainbow tradeoff [18], this choice of nesting is logical, as it is expected to reduce the computational complexity of the online phase.

There is no such generally accepted practice for the placement of the k-loop, the loop associated with the inversion targets, appearing in Algorithm 1 and Algorithm 2. However, the choice given here is reasonable in that, when the pre-computation tables are too large to be fully loaded into fast memory, it is much more practical to handle frequent changes of the targets than to handle frequent changes of the tables one is accessing.

Note that, when we set $s = 1$ in Algorithm 2, the outer loop is removed and Algorithm 2 reduces essentially to Algorithm 1. Hence, in the remainder of this paper, we will deal only with Algorithm 2, and any argument made for Algorithm 2 may be understood to be an argument that is also applicable to Algorithm 1 by taking $s = 1$.

We will use the usual notation for search space size N, online computational time complexity T, storage complexity M, and pre-computation time complexity P. For a tradeoff method of D targets, its *tradeoff coefficient* is defined to be TM^2D^2/N^2 and its *pre-computation coefficient* is defined to be PD/N. These two definitions reduce to their corresponding single-target definitions given by [12, 13, 14, 15] when $D = 1$.

3. Analysis

In the previous section, we explained that it suffices to work with just Algorithm 2 in order to cover all of the classical Hellman, non-perfect DP, perfect DP, non-perfect fuzzy rainbow and perfect fuzzy rainbow tradeoff methods. When the Hellman or DP tradeoff methods are mentioned below, we are implicitly assuming the dummy parameter $s = 1$.

Our main interest lies with the computational complexity of Algorithm 2. Let us write $|\mathtt{M}_i|$ to denote the expected number of distinct entries contained in the i-th colored DP sub-matrix for the j-th pre-computation matrix. Note that we have suppressed the index j in the notation $|\mathtt{M}_i|$, because the expected numbers are independent of the specific tables. More precisely, this value will be a function of the tradeoff algorithm parameters m, t, and s, in addition to the color index i.

Using this notation, the probability for the operations inside the inner-most loop corresponding to a specific $i = x$, $j = y$, $k = z$ index triple to be executed during the online phase may be written as

$$P_{x,y,z} := \prod_{i=x+1}^{s} \left(1 - \frac{|\mathtt{M}_i|}{N}\right)^{\ell D} \cdot \left(1 - \frac{|\mathtt{M}_x|}{N}\right)^{(y-1)D} \cdot \left(1 - \frac{|\mathtt{M}_x|}{N}\right)^{z-1}. \tag{1}$$

Here, the first product term is the probability for none of the answers corresponding to the D inversion targets to exist among the DP sub-matrices of all colors appearing strictly after the x-th color for all ℓ pre-computation tables. The second term is the probability for none of the D answers to be among the x-th color DP sub-matrices corresponding to the first $y-1$ tables. The final term is the probability for the answers to the first $z-1$ inversion targets to be not present in the x-th color DP sub-matrix for the y-th table.

Let us next briefly consider the expected number of one-way function iterations required of carrying out the operations appearing in the inner-most loop of Algorithm 2. Note that this number is a function of algorithm parameters m, t, and s, but is independent of the specific table and also of the specific inversion target. Hence, we can use notation W_i to denote the work factor associated with the operations corresponding to indices i, j, and k. That is, as with our discussion concerning $|\mathtt{M}_i|$, the indices j and k need not be associated with this expected number.

Now, using the expression (1) for the execution probabilities, the expected online computational complexity

$$T = \sum_{i=1}^{s} \sum_{j=1}^{\ell} \sum_{k=1}^{D} P_{i,j,k} \cdot W_i \tag{2}$$

can be written in the form

$$T = \sum_{i=1}^{s} W_i \cdot \frac{N}{|\mathtt{M}_i|} \Big\{ 1 - \Big(1 - \frac{|\mathtt{M}_i|}{N} \Big)^{D\ell} \Big\} \prod_{h=i+1}^{s} \Big(1 - \frac{|\mathtt{M}_h|}{N} \Big)^{D\ell} \tag{3}$$

after some easy simplifications.

This is certainly a function of the algorithm parameters m, t, ℓ, s, and D, but we can make the crucial observation that this expression could also be understood as a function of m, t, s, and $D\ell$. That is, if D and ℓ were changed in such a way that the product $D\ell$ remains the same, then the online time complexity T remains the same.

To summarize this finding, we introduce the notion of *matching* parameter sets. Let us fix any one of the classical Hellman, non-perfect DP, perfect DP, non-perfect fuzzy rainbow, and perfect fuzzy rainbow tradeoff methods. Consider a set of parameters m, t, s, and ℓ for the single-target version of this tradeoff method. Next, consider the multi-target version of the same tradeoff method that aims to invert just one of D targets, and let us associate the set of parameters m, t, s, and $\frac{\ell}{D}$ to this method. In the remainder of this paper, we will refer to these two sets of parameters as *matching* parameter sets for the single-target and multi-target versions of the same tradeoff method. In other words, the matching parameter sets associated with a single-target tradeoff method and the corresponding multi-target tradeoff method differ only in their numbers of tables, with the table count for the single-target version being equal

to the product of the table count and the target count for the multi-target version.

As was observed above, expression (3) for the computational complexity of Algorithm 2 implies the following.

Proposition 3.1. *The online time complexity of a single-target tradeoff method and that of the corresponding multi-target tradeoff method, executed under matching parameter sets, are equal.*

One consequence of this result is that the explicit formulas for time complexities associated with any set of parameters that were obtained in [12, 13, 14, 15] for the single-target tradeoff methods can easily be understood to be those for the multi-target tradeoff methods. It suffices to replace every table count ℓ appearing in those formulas with the product of table count and target count being used by the multi-target tradeoff method.

We have tested the correctness of the above claim with computer program experiments for the non-perfect DP and the perfect fuzzy rainbow tradeoff methods. For each of the two tradeoff methods, we experimented with both small and large target sets, comparing the experimentally obtained online times with the theoretical time complexities that were stated by [12] and [14] for the single-target setting.

It is rather straightforward to argue that the success rates of single-target and multi-target tradeoff methods executed under matching parameter sets are equal, both being

$$1 - \prod_{i=1}^{s} \left(1 - \frac{|\mathtt{M}_i|}{N}\right)^{\ell}, \tag{4}$$

where ℓ is the number of tables for the single-target tradeoff method. It is also true that, when executed under matching parameter sets, the pre-computation complexities and storage complexities of a single-target tradeoff method is larger by a factor of D, the number of targets, than the multi-target tradeoff method. To see this, it suffices to note that both complexities are linear in the number of tables. We have thus arrived at the following claim, where the two coefficients are as they were defined at the end of Section 2.

Theorem 3.2. *The success rate of a single-target tradeoff method and that of a corresponding multi-target tradeoff method, executed with matching parameter sets, are equal. Corresponding statements concerning the pre-computation coefficients and the tradeoff coefficients are also true.*

Recall that the performances of different single-target tradeoff methods could be compared [12, 15] based on the range of pre-computation coefficient and tradeoff coefficient pairs that are made available by each method through various parameter choices achieving a common success rate. Since any storage optimization technique that reduces the number of bits required to record each pre-computation table entry does not depend on the number of targets, we

can apply the same performance comparison strategy to multi-target tradeoff methods. The above theorem implies that, for each of the five tradeoff methods under consideration, the range of these pairs is the same regardless of the number of inversion targets. Thus, the performance comparisons carried out by [12, 13, 14, 15] remain valid without change for the multi-target versions of the five methods.

4. Conclusion

In this work, we analyzed the execution behaviors of the classical Hellman, non-perfect DP, perfect DP, non-perfect fuzzy rainbow, and perfect fuzzy rainbow tradeoff methods in their multi-target settings. We showed that the online time complexity of each multi-target tradeoff method is identical to that of the corresponding single-target tradeoff method executed under a certain matching parameter set. This implies that previous performance comparisons of tradeoff methods done under the single-target setting remain valid in the multi-target setting. In particular, referring to the claim of [14], we can conclude that the perfect table fuzzy rainbow multi-target tradeoff method performs the best among the major multi-target tradeoff methods. This is a meaningful conclusion, as the fuzzy rainbow tradeoff, in both its single-target and multi-target versions, has yet to receive the attention it deserves.

References

[1] S. H. Babbage, *Improved "exhaustive search" attacks on stream ciphers*, In European Convention on Security and Detection, IEE Conference Publication (1995), no. 408, 161–166.

[2] E. P. Barkan, *Cryptanalysis of Ciphers and Protocols*, Ph.D. Thesis, Technion—Israel Institute of Technology, March 2006.

[3] E. Barkan, E. Biham, and A. Shamir, *Rigorous bounds on cryptanalytic time/memory tradeoffs*, In Advances in Cryptology—CRYPTO 2006, 1–21, Lecture Notes in Comput. Sci., 4117, Springer, Berlin, 2006.

[4] A. Biryukov, S. Mukhopadhyay, and P. Sarkar, *Improved time-memory trade-offs with multiple data*, In SAC 2005, 110–127, Lecture Notes in Comput. Sci., 3897, Springer, Berlin, 2006.

[5] A. Biryukov and A. Shamir, *Cryptanalytic time/memory/data tradeoffs for stream ciphers*, In Advances in Cryptology—ASIACRYPT 2000, 1–13, Lecture Notes in Comput. Sci., 1976, Springer, Berlin, 2000.

[6] A. Biryukov, A. Shamir, and D. Wagner, *Real time cryptanalysis of A5/1 on a PC*, In FSE 2000, 1–18, Lecture Notes in Comput. Sci. **1978**, Springer, 2001.

[7] J. Borst, *Block Ciphers*: *Design, Analysis, and Side-Channel Analysis*, Ph.D. Thesis, Katholieke Universiteit Leuven, September 2001.

[8] J. Borst, B. Preneel, and J. Vandewalle, *On the time-memory tradeoff betweeen exhaustive key search and table precomputation*, In Proceedings of the 19th Symposium on Information Theory in the Benelux, WIC, 1998.

[9] D. E. Denning, *Cryptography and Data Security*, Addison-Wesley, 1982.

[10] J. Dj. Golić, *Cryptanalysis of alleged A5 stream cipher*, In Advances in Cryptology—EUROCRYPT '97, 239–255, Lecture Notes in Comput. Sci. **1233**, Springer, 1997.

[11] M. E. Hellman, *A cryptanalytic time-memory trade-off*, IEEE Trans. Inform. Theory **26** (1980), no. 4, 401–406.
[12] J. Hong and S. Moon, *A comparison of cryptanalytic tradeoff algorithms*, J. Cryptology **26** (2013), no. 4, 559–637.
[13] B.-I. Kim and J. Hong, *Analysis of the non-perfect table fuzzy rainbow tradeoff*, In ACISP 2013, 347–362, Lecture Notes in Comput. Sci. **7959**, Springer, 2013.
[14] ———, *Analysis of the perfect table fuzzy rainbow tradeoff*, J. Appl. Math. **2014** (2014), Article ID 765394.
[15] G. W. Lee and J. Hong, *A comparison of perfect table cryptanalytic tradeoff algorithms*, Des. Codes Cryptogr. **80** (2016), no. 3, 473–523.
[16] K. Nohl, *Attacking phone privacy*, Presented at Black Hat USA 2010, Las Vegas, July 2010.
[17] K. Nohl and C. Paget, *GSM-SRSLY?*, Presented at 26th Chaos Communication Congress (26C3), Berlin, December 2009.
[18] P. Oechslin, *Making a faster cryptanalytic time-memory trade-off*, In Advances in Cryptology—CRYPTO 2003, 617–630, Lecture Notes in Comput. Sci., 2729, Springer, Berlin, 2003.

Jin Hong
Department of Mathematical Sciences and ISaC
Seoul National University
Seoul 151-747, Korea
E-mail address: jinhong@snu.ac.kr

Byoung-Il Kim
Department of Mathematical Sciences and ISaC
Seoul National University
Seoul 151-747, Korea
E-mail address: samaria2@snu.ac.kr