

# Chapter 2

## Cryptanalysis of Multilanguage Encryption Techniques

Prasanna Raghaw Mishra, Indivar Gupta and Navneet Gaba

**Abstract** We present an analysis of an encryption scheme MUlti-Language Encryption Technique (MULET) proposed by G. Praveen Kumar et al. in the Seventh International Conference on Information Technology, ITNG 2010. Using our analysis, we have successfully recovered 80% of the plaintext from the MULET ciphertext. We also give quantitative results in support of our findings.

### 1 Introduction

MUlti-Language Encryption Technique (MULET) proposed by Praveen Kumar et al. [11] is an encryption scheme designed to facilitate encryption/decryption for a range of languages supported by Unicode [14]. The authors have shown that the scheme is secure against brute-force attack only and have not discussed its security against cryptanalytic attacks. However, the scheme escaped the attention of cryptanalysts. Although Anoop Kumar et al. [1] indicated some of the flaws in the technique, no comprehensive cryptanalysis was presented. This motivated us to go for an in-depth cryptanalysis of the scheme. We have launched a ciphertext only attack [13]

---

The authors are grateful to Dr. P. K. Saxena, Director SAG for his support and encouragement. The authors would also like to express their sincere thanks to Dr. S. S. Bedi, Associate Director SAG for the valuable suggestions given by him during the course of the work.

---

P. R. Mishra (✉) · I. Gupta · N. Gaba  
Scientific Analysis Group, Defence Research and Development Organisation,  
Delhi 110054, India  
e-mail: prasanna.r.mishra@gmail.com

I. Gupta  
e-mail: indivar\_gupta@yahoo.com; indivargupta@sag.drdo.in

N. Gaba  
e-mail: navneetgaba2000@yahoo.com

**Table 1** Notations

$M$	Mapping constant/modulus
$ch\_map$	A set of $M$ -characters from the universal character set is considered as a mapping array
$chno$	A set of characters for universal character set is considered as a substitution array
$Quo$	Quotients required for decryption (key)
$Enc$	Ciphered text
$Dec$	Deciphered text

on MULET ciphertext and successfully recovered more than 80 % of the plaintext out of it. The organization of this paper is as follows.

In the Sect. 2, we describe MULET algorithm in brief. In Sect. 3, we describe our technique to recover plaintext out of MULET ciphertext. MULET is a double-layer encryption scheme and we have tried to remove layers in the reverse order to get back the plaintext. In Sect. 4, we give a step-by-step complexity analysis of the technique. In Sect. 5, we give the results of our attack applied on a ciphertext of English encrypted with MULET.

## 2 Description of MULET

In this section, we describe MULET encryption and decryption algorithm in brief [11]. Before describing the scheme, we first discuss the notations used in the scheme (see Table 1).

### The Scheme

**Key:** *Secret Keys*-  $M$ ,  $ch\_map$ ,  $chno$ , *Publicly Known*- Unicode

#### Encryption Algorithm

Input: Plaintext, Arrays  $ch\_map$  and  $chno$ , Modulus  $M$

Output: Ciphertext, Array  $Quo$

```

while (! End of plaintext) do
    Read a character from the original file and store the Unicode value in a variable  $n$  ;
     $R := n \% M$ 
     $Quo[i] := n / M$ 
     $Enc[i] := ch\_map[R]$ 
    Increment  $i$  ;
end while
while (! end of Enc) do
    while ( $Enc[i] == Enc[i + 1]$ ) do
        Increment count;
        Increment  $i$  ;
    end while
    if (count  $\geq 2$ ) then
        Replace the repetitions with  $chno[count]$  in enc
        Reset count to zero
    end if
end while

```

**Decryption Algorithm:**  
Input: Ciphertext, Array *Quo*  
Output: Plaintext  
  
**while** (! end of enc) **do**  
  **if** (character is *chno*[*i*]) **then**  
    Remove the character from enc and the character preceding *chno*[*i*] in the cipher text is repeated '*i*' number of times and store in dec  
  **end if**  
**end while**  
**while** (!end of dec) **do**  
  Compare the character with the mapping array *ch\_map*; Position of the character in *ch\_map* is the required remainder *R*;  
   $U := Quo[i] * M + R$ ;  
  Convert *U* to the corresponding character;  
**end while**

2.1 Example of MULET

Mapping Constant/Modulus *M* = 16  
*ch\_map*:

0	1	2	3	4	5	6	7
अ	आ	इ	ई	उ	ऊ	ऋ	ॠ
8	9	10	11	12	13	14	15
ऐ	ए	ए	ऐ	ऑ	ओ	ओ	औ

*chno*:

0	1	2	3	4	5	6	7	8	9
०	१	२	३	४	५	६	७	८	९

Plaintext:

Cryptography is the science of secret writing

Key generated (in Hex):  
724 707 6f7 726 706 796 692 207 206 637 656  
636 206 666 732 636 657 207 727 746 6e6 a6

**Ciphertext:**

ईऐउलआँअईआईऐओऊऔअऊइउलऐरल

### 3 Cryptanalysis

MULET encryption is done in two layers. The first layer makes use of secret data *ch\_map* and the modulus  $M$  and the second layer uses secret data *chno*. We notice that because of the two layers of encryption, the existing attacks on classical ciphers [2, 13] are not applicable directly on MULET. We have devised a ciphertext only attack [13] on the scheme to recover various secret parameters and finally the plaintext. The only assumption we make is that the plaintext language is known. There are two main parameters whose knowledge leads to recovery of plaintext, viz., the modulus  $M$  and the useful portion of array *chno*. To start with, we first guess the modulus  $M$ .

#### 3.1 Guessing the Modulus

A MULET ciphertext contains characters from both the arrays, viz. *ch\_map* and *chno*. Let the number of distinct characters occurring in the ciphertext be  $d$  and the number of characters from *chno* occurring in the ciphertext be  $b_1$ . We observe that the index 0 and 1 of *chno* is never accessed. Similarly, an index higher than 5 corresponds to 6-graph or higher. For a reasonable size of modulus ( $M \geq \text{size of plaintext alphabet}/2$ ) occurrence of 6-graph or higher is extremely rare, therefore, index higher than 5 is rarely accessed. Thus, the wise choice of  $b_1$  to start with is 4.

The maximum number of characters from *ch\_map* that can occur in ciphertext is  $M$ . Thus, the bound on the number of distinct characters occurring in ciphertext is  $M + b_1$ . There is a possibility that all the 4 characters from *chno* may not be occurring in the ciphertext. Similarly, some characters from *ch\_map* may also escape ciphertext. Let the maximum number of characters from the two arrays not occurring in the ciphertext be  $b_2$ . Now we have the following relation:

$$M - b_2 + b_1 \leq d \leq M + b_1$$

which implies that  $M$  can be found by trying  $b_2$  values precisely  $d - b_1, d - b_1 + 1, \dots, d - b_1 + b_2$ . The value of  $b_2$  is relatively small (say  $\leq 10$ , as it is less likely that more than 10 characters from *ch\_map* are skipped) in most cases. Based on experimentation, the value of  $b_2$  is taken as 6 in our case. As the value of  $d$  is determined from the ciphertext, modulus  $M$  can be guessed in much smaller number of trials.

### 3.2 Segregating the *ch\_map* and *chno* Characters in the Ciphertext

Let us assume that the set of unicode values of the plaintext alphabets are  $\mathcal{A} = \{v_i, i = 1, 2, \dots, l\}$ . and the expected frequency of  $v \in \mathcal{A}$  in a meaningful text is  $f_v^e$ . The first layer of encryption process uses a function  $\phi_M : \mathcal{A} \rightarrow \{0, 1, \dots, M-1\}$  given as  $\phi_M(v) = r, r \equiv v \pmod{M}, 0 \leq r < M$ . Clearly,  $\phi_M$  is many-one if  $M < l$ .  $M \geq l$  makes the cipher merely a simple substitution [2, 8], and it can be analyzed using existing methods [2, 3, 5, 10].

We have only to consider the other case, i.e.,  $M < l$ . The remainders are replaced by the corresponding *ch\_map* array. Once the first layer encryption is over, an  $n$  consecutive occurrence of a character  $v$  is replaced by the couple of characters *ch\_map*( $r$ )*chno*( $n$ ). Here, we make an assumption that the second layer changes do not alter the relative frequency distribution of *ch\_map* characters in the ciphertext. We find expected frequency distribution of  $\phi_M$  after the first layer. The expected frequency  $R_r$  of a value  $r$  of  $\phi_M$  can be given as  $R_r = \sum_{v \in \phi_M^{-1}(r)} f_v^e$ . Expected frequency of a *chno* character at index  $n$  is the same as frequency of  $n$ -ets ( $n$  consecutive occurrences of any character) in the first layer text. Let these frequencies lie in the interval  $[0, b_3]$ . We consider the set

$$S = \{c \text{ is a ciphertext character} : \text{frequency of } c \leq b_3\}$$

We select  $b_1$  characters from  $S$  and arrange them in decreasing order of their frequencies. The highest frequent character corresponds to index 2, the next to 3, and so on. With this information, we remove the second layer changes and compute the frequency distribution of the changed ciphertext. We measure how close the resulting frequency distribution is to the expected one. To measure the closeness we use a metric similar as  $\ell_1$  metric. The distance between the guessed and the expected frequency distributions  $d$  with respect to our metric is given as  $\frac{\sum_{i=1}^M |R_{r_i} - f_{c_i}^e|}{M}$  (meanings of the symbols used are described in the next section). We carry out trials on all possible values of  $b_1$ . There will be  $\binom{o(S)}{b_1}$  trials for a given value of  $b_1$ . The selection giving the minimum distance will reveal the part of *chno* used in the second layer (it should be noted that minimum is calculated over all possible values of  $b_1$  and  $M$ ).

### 3.3 Making Final Substitution

We assume that upto this stage we have successfully undone the second layer changes. We rewrite the expected frequency distribution of  $\phi_M$  as  $(r_1, R_{r_1}), (r_2, R_{r_2}), \dots, (r_M, R_{r_M})$  such that  $1 \leq i < j \leq M \implies R_{r_i} \geq R_{r_j}$  (where  $\{r_1, r_2, \dots, r_M\} = \{0, 1, \dots, k-1\}$ ). Let the frequency of ciphertext character  $c$  be  $f_c^o$ . The frequency distribution is  $(c_1, f_{c_1}^o), (c_2, f_{c_2}^o), \dots, (c_M, f_{c_M}^o)$  such that  $1 \leq i < j \leq M \implies$

$f_{c_i}^o \geq f_{c_j}^o$  (where  $c_1, c_2, \dots, c_M$  are distinct letters in the intermediate ciphertext). From here we guess the map  $u : \{c_1, c_2, \dots, c_M\} \rightarrow \{0, 1, \dots, M-1\}$  which establishes the relation between remainders and ciphertext characters as  $u(c_i) = r_i$ ,  $i = 1, 2, \dots, M$ . We finally replace the ciphertext character  $c$  by  $v_c^{mp}$  where  $v_c^{mp}$  is chosen such that  $f_{v_c^{mp}}^e = \max_{v \in \phi_M^{-1}(u(c))} \{f_v^e\}$ .  $v_c^{mp}$  is the most probable replacement for the ciphertext character  $c$ .

## 4 Complexity Analysis

The first step of attack, i.e., guessing the modulus requires at most  $d$  trials. While the removal of the second layer encryption requires  $\binom{o(S)}{b_1}$  calculations of frequency distributions and distance calculations. For a ciphertext of length  $L$ , the total number of operations required to find frequency distribution will roughly take  $2L$  operations. The Euclidean distance calculation will take at most  $M \log_2 L$  operations. Therefore, the total number of trials are bounded above by the expression  $\sum_{i=1}^{b_1} \sum_{M=1}^d \binom{o(S)}{i} (2L + M \log_2 L) \leq L \left( \sum_{i=1}^{b_1} \sum_{M=1}^d \binom{o(S)}{i} (M+2) \right)$ . This shows that our attack is linear in the size of ciphertext.

## 5 Experimental Results

To verify our strategy, we encrypted an English text of 1,000 characters with MULET. The two arrays and the modulus we took were the same as taken in example 1 in [11]. After making trials on  $M$  and removing the first layer, we found that the modulus  $M$  is 16. We calculate the frequency of characters of the ciphertext. Table 2 gives the ciphertext characters and their percentage occurrence in the ciphertext in descending order.

We calculated the remainder  $r$  for each of the 52 English unicode characters. The characters giving the same remainder were grouped together. Table 3 lists the remainder  $r$ , its expected frequency ( $R_r$ ) corresponding English character ( $v$ ), and the most probable among them ( $v^{mp}$ ) in descending order of  $R_r$ .

A comparison of Tables 2 and 3 suggests the possible mapping of ciphertext characters and remainders. We replace a ciphertext character with the most probable character corresponding to the possible remainder. In Table 4 we show the ciphertext characters, their possible remainders, and the possible replacement (the most probable character corresponding to the possible remainder).

Carrying out these replacements gives us the recovered plaintext. A comparison of the recovered plaintext from the original one reveals that the attack successfully recovers 83.36% of the plaintext in our case. The point to note is that this recovered text may further be fed to text mining technique including pattern recognition and dictionary-based techniques [4, 6, 7, 9, 12] to further enhance the success rate.

**Table 2** Frequency distribution of ciphertext character

S.N.	Ciphertext character	Percentage occurrence
1	0x090a	15.729
2	0x0909	13.022
3	0x090e	9.635
4	0x0908	9.461
5	0x0914	7.909
6	0x0906	7.798
7	0x0907	7.001
8	0x0913	6.833
9	0x090d	5.209
10	0x090c	4.456
11	0x0911	3.917
12	0x090b	3.276
13	0x0912	2.384
14	0x0905	2.291
15	0x0910	0.870
16	0x090f	0.207

**Table 3** Pre-computed table for determination of most probable character corresponding to a given remainder

S.N.	Remainder ( $r$ )	Expected frequency of $r(R_r)$	Possible plaintext characters ( $v$ )	Most probable character ( $v^{mp}$ )
1	5	15.6289	E U e u	e
2	4	13.1218	D T d t	t
3	9	9.6341	I Y i y	i
4	3	9.4623	C S c s	s
5	15	7.9191	O o	o
6	1	7.7885	A Q a q	a
7	2	7.0108	B R b r	r
8	14	6.8234	N n	n
9	8	5.2195	H X h x	h
10	7	4.4463	G W g w	g
11	12	3.9274	L l	l
12	6	3.2665	F V f v	f
13	13	2.3832	M m	m
14	0	2.2916	P p	p
15	11	0.8688	K k	k
16	10	0.2084	J Z j z	j

**Table 4** Possible plaintext character corresponding to a ciphertext character

S.N.	Ciphertext character	Remainder	Possible replacement
1	0x090a	5	e
2	0x0909	4	t
3	0x090e	9	i
4	0x0908	3	s
5	0x0914	15	o
6	0x0906	1	a
7	0x0907	2	r
8	0x0913	14	n
9	0x090d	8	h
10	0x090c	7	g
11	0x0911	12	l
12	0x090b	6	f
13	0x0912	13	m
14	0x0905	0	p
15	0x0910	11	k
16	0x090f	10	j

The bounds we set were  $b_1 = 4, b_2 = 6, b_3 = 5$  and  $b_4 = 10$ . For our case, the plaintext language was English. For  $b_1 = 4, b_3 = 5$  we have  $o(S) \leq 11$  (see Table 2). So, the worst case complexity will be  $L \left( \sum_{i=1}^{b_1} \sum_{M=1}^{20} \binom{11}{i} (M+2) \right) = L \left( \sum_{i=1}^4 \binom{11}{i} \sum_{M=1}^{20} (M+2) \right) = L \times 550 \times 250 = 137500L$ . For  $L = 1000$  the complexity is of order  $2^{27}$ . It is to be noted that the calculations are made for the worst case and we have finished our attack in a much lesser time. Below are given the parts of plaintext, ciphertext, and the recovered.

<b>Plaintext</b>	:	P a g e o f Y O . . .
<b>Unicode value</b>	:	0x50, 0x61, 0x67, 0x65, 0x6f, 0x66, 0x59, 0x4f, . . .
<b>MULET encryption (Unicode value)</b>	:	0x0905, 0x0906, 0x090c, 0x090a, 0x0914, 0x090b, 0x090e, 0x0914, . . .
<b>Recovered text</b>	:	p a g e o f i o . . .

## 6 Conclusion

In this paper, we have presented a cryptanalysis of an encryption scheme named MULti-Language Encryption Technique (MULET). Based on our analysis, we have launched a ciphertext-only attack and retrieved more than 80 % of the plaintext from



MULET ciphertext. We have also shown that the scheme can be broken in linear time, contrary to the claim of exponential complexity by the proposers of the scheme. There are many more schemes proposed on this philosophy. We hope that this analysis will be helpful to demonstrate the weaknesses of such schemes.

## References

1. Anoop Kumar, S., Sanjeev S., Santosh, S.: MSMET: a modified and secure multilanguage encryption technique. *Int. J. Comput. Sci. Eng.* **4**(03), 402–405 (2012)
2. Bauer, F.L.: *Decrypted Secret: Methods and Maxims of cryptology*, 4th edn. Springer, Germany (2006)
3. Churchhouse, R.: *Codes and Ciphers, Julius Caesar, the Enigma and the Internet*. Cambridge University Press, Cambridge (2002)
4. Elder, J., Miner, G., Nisbet, B.: *Practical Text Mining and Statistical Analysis for Non-structure Text Data Applications*. Academic Press, Burlington (2012)
5. Harris, F.A., Tuck, S.: *Solving Simple Substitution Ciphers*. American Cryptogram Association, New York (1959)
6. Jakobsen, T.: A fast method for cryptanalysis of substitution ciphers. *Cryptologia* **19**(3), 265–274 (1995)
7. Lucks, M.: A constraint satisfaction algorithm for the automated decryption of simple substitution ciphers. In: *Advance in Cryptology, CRYPTO'1988*, LNCS, vol. 1462, pp. 132–144. Springer, Berlin (1998)
8. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: *Handbook of Applied Cryptography*. CRC Press, New York (1997)
9. Olson, E.: Robust dictionary attack of short simple substitution ciphers. *Cryptologia* **31**(4), 332–342 (2007)
10. Peleg, S., Rosenfeld, A.: Breaking substitution ciphers using a relaxation algorithm. *Commn. ACM* **22**(11), 598–605 (1973)
11. Praveen Kumar, G., Arjun Kumar M., Parajuli, B., Choudhury, P.: MULET: a multilanguage encryption technique. In: *Proceedings of Seventh International Conference on Information Technology 2010*. IEEE Computer Society (2010)
12. Spillman, R., Janssen, M., Nelson, B., Kepner, M.: Use of a genetic algorithm in the cryptanalysis of simple substitution ciphers. *Cryptologia* **XVII**(3), 31–44 (1993)
13. Stamp, M., Low, R.M.: *Applied Cryptanalysis: Breaking Ciphers in the Real World*. John Wiley & Sons, Hoboken (2007)
14. Unicode Character form. <http://www.Unicode.org>

Mathematics and Computing 2013

International Conference in Haldia, India

Mohapatra, R.N.; Giri, D.; Saxena, P.K.; Srivastava, P.D.

(Eds.)

2014, XXV, 352 p. 81 illus., Hardcover

ISBN: 978-81-322-1951-4