Cryptography midterm report

Tran Van Sang, Department of Computer Science, 48-186106
Eguchi Shingo, Department of Computer Science, 48-186016

June 9, 2018

Contents

Chapter 1

Topic

We presented basic checksum algorithms such as the one used in credit card numbers. Explain what is a quasigroup. Present and analyze Damm ' s method based on quasigroups to detect all single-digit and transposition errors. Are there other quasigroups that work ?

Chapter 2

Hash function

Hash function is a class of function that takes input of arbitrary length, called message, and outputs of fixed length, called hash. Hash function has important role in cryptography and many applications due to its one-wayness. Hash function ability to resist cryptanalytic attack is mainly determined by 3 properties

- Pre-image resistance. Sometime referred as one-wayness. Given a hash value h, it should be infeasible within acceptable time to find original domain value m such that the hash function maps m into h.
- Second pre-image resistance. It should be infeasible within acceptable time to find another input m ' that has same image mapped by the hash function, given input m.
- Collision resistance. It should be infeasible within acceptable time to find 2 input messages m, m' such that the hash function maps them into same hash.

It is obvious that pre-image resistance hash function is also pre-image resistance. Similarly, functions that resist collision also have second pre-image resistance property.

Chapter 3

ISBN

3.1 Introduction

International Standard Book Number (ISBN) is an unique 13 digits long number assigned to a book. Affiliates of International ISBN Agency have ability to release a number whenever being requested by publisher. The last digit is the hash of 10 preceding digits. In other words,

$$ISBN = m_{13}, m_{12}, m_{11}, \ldots, m_1$$
$$m_1 = isbn(m_{13}, m_{12}, m_{11}, \ldots, m_2)$$

Whether

$$isbn(m_{13}, m_{12}, m_{11}, \ldots, m_2) = \sum_{i \equiv 1 \pmod 2, 2 \le i \le 13} m_i + 3 \times \sum_{i \equiv 0 \pmod ()2, 2 \le i \le 13} m_i$$

3.2 Detection ability

Using ISBN system, one can detect one digit changed or 2 consecutive digits swapped unless that those numbers' difference is 5 (i.e. swapping numbers are one of (0, 5), (1, 6), (2, 7), (3, 8), or (4, 9))

To check if there is any modification on the received ISBN number, one can check by calculating checksum $C = \sum_{i \equiv 1 \pmod 2, 2 \le i \le 13} m_i + 3 \times \sum_{i \equiv 0 \pmod 2, 2 \le i \le 13} m_i - m_1 \mod 10$ equal to 0.

3.3 Proof

3.3.1 Single digit modification

We will prove with this method 2 kinds of modification mentioned above can be detected.
Assume that ith number is changed to m'_i while other numbers keep the same.
Absolute of difference between new sum and old sum is

$$|C - C'| \mod 10 = \begin{cases} |m_i - m'_i| \mod 10 & \text{if} i \equiv 1 \pmod 2 \\ 3 \times |m_i - m'_i| \mod 10 & \text{otherwise} \end{cases}$$

Because $GCD(3, 10) = 0$, $|C - C'| \mod 10 \neq 0$

3.3.2 Consecutive swapping

If 2 digits m_i and m_{i+1} are swapped, and $|m_i - m_{i+1}| \neq 5$
Similar to previous subsection

$$\begin{aligned} |C - C'| \mod 10 &= \begin{cases} |m_i + 3 \times m_{i+1} - 3 \times m_i - \times m_{i+1}| \mod 10 & \text{if} i \equiv 1 \pmod 2 \\ |3 \times m_i + m_{i+1} - m_i - 3 \times m_{i+1}| \mod 10 & \text{otherwise} \end{cases} \\ &= 2 \times |m_i - m_{i+1}| \end{aligned} \tag{3.1}$$

Thus, $|C - C'| \mod 10 \neq 0$

3.3.3 Generalization

The ISBN checksum is one special case of the hash function $H(m) = \sum_{i \geq 2} a_i m_i \mod 10$ where $a_i (i \geq 2)$ are fixed
In the following chapter we will discuss on other form of checksum with better error detection ability.

Chapter 4

Credit card checksum

Credit card number contains n number $m_n, m_{n-1}, \dots, m_2, m_1$ where n is variate depends on credit card provider. For example, $n = 16$ for Visa, Master card, $n = 19$ for UnionPay card
The following hash function is used in credit card to detect error

$$m_1 = H(m)$$

is chosen such that

$$\sum_{iisodd} m_i + \sum_{iiseven} f(m_i) \equiv 0 \pmod{10}$$

whether

$$f(x) = 2x + \left[\frac{x}{5}\right] \mod 10$$

This $f(x)$ is called Luhn (1954) function.
By this checksum, it is able to check for single digit error or consecutive digits swap unless they are not (0, 9) or (9, 0) pairs.

Before giving proof to the error resistance, we observe some properties of the Luhn function

x	0	1	2	3	4	5	6	7	8	9
f(x)	0	2	4	6	8	1	3	5	7	9
$(x - f(x)) \mod 10$	0	9	8	7	6	4	3	2	1	0

Firstly, f maps $\{0, 1, \dots, 9\}$ to the same set $\{0, 1, \dots, 9\}$ without collision. In other words, for all $x_1 \neq x_2$ $f(x_1) \neq f(x_2)$. Functions having this property are called permutation functions.

Secondly, there are only 2 fixed points (0 and 9), i.e. $x = f(x)$ and for all x, $(x - f(x)) \mod 10$ are almost different except 0 and 9.

Now we move on proving the error resistance.

If only one digit is changed, m_i is changed to m'_i,

$$|C - C'| \mod 10 = \begin{cases} |m_i - m'_i| \mod 10 & \text{if} i \equiv 1 \pmod 2 \\ f(m_i) - f(m'_i) \mod 10 & \text{otherwise} \end{cases}$$

Because of permutation property, this value never be zero. Thus, the error can be detected

If 2 consecutive number m_i, m_{i+1} are swapped.

$$\begin{aligned} |C - C'| \mod 10 &= \begin{cases} |m_i + f(m_{i+1}) - m_{i+1} - f(m_i)| \mod 10 & \text{if} i \equiv 1 \pmod 2 \\ |f(m_i) - m_{i+1} - f(m_{i+1}) - m_i| \mod 10 & \text{otherwise} \end{cases} \\ &= |(m_i - f(m_i) - (m_{i+1} - f(m_{i+1}))| \mod 10 \end{aligned} \tag{4.1}$$

If $m_i = m_{i+1}$, there is no need for detection. If m_i, m_{i+1} are not (0, 9) or (9, 0) pair, from the above table, $m_i - f(m_i) \neq m_{i+1} - f(m_{i+1})$. Thus, $|C - C'| \mod 10 \neq 0$

Chapter 5

Damm's method

Damm's method is check digit algorithm based on *quasigroup*. It can detect all single-digit errors and all adjacent transposition errors.

5.1 Quasigroup

Quasigroup $(Q, *)$ is a set Q with a binary operation "$*$" such that *Latin square property* holds: for every $a, b \in Q$, there exists unique x and y satisfy the following.

$$a * x = b$$
$$y * a = b$$

Let us consider a *Cayley table* of the operation "$*$". Latin square property can be characterize by whether the table is *Latin square*: each element occurs exactly once in each row and exactly once in each column. For example, the operation expressed by the following table satisfies Latin square property.

$*$	0	1	2
0	0	1	2
1	1	2	0
1	2	0	1

The following lemma is derived immediately, which is significant for error detection of damm's method.

Lemma 1 *Suppose $(Q, *)$ is quasigroup. For arbitrary x, x' and a in Q, the following hold.*

$$x \neq x' \ \Rightarrow a * x \neq a * x' \tag{5.1}$$
$$x \neq x' \ \Rightarrow x * a \neq x' * a \tag{5.2}$$

5.2 Damm's method

This algorithm is based on quasigroup $(Q, *)$ of 10-order with a special property: for all $c, x, y \in Q$ the followings hold.

$$(c * x) * y = (c * y) * x \ \Rightarrow x = y \tag{5.3}$$

$$x * x = 0 \tag{5.4}$$

The example of such a quasigroup is shown in Figure 5.1.

*	0	1	2	3	4	5	6	7	8	9
0	0	3	1	7	5	9	8	6	4	2
1	7	0	9	2	1	5	4	8	6	3
2	4	2	0	6	8	7	1	3	5	9
3	1	7	5	0	9	8	3	4	2	6
4	6	1	2	3	0	4	5	9	7	8
5	3	6	7	4	2	0	9	5	8	1
6	5	8	6	9	7	2	0	1	3	4
7	8	9	4	5	3	6	2	0	1	7
8	9	4	3	8	6	1	7	2	0	5
9	2	5	8	1	4	3	6	7	9	0

Figure 5.1: Example:Quasigroup for damm's method

The validity of a digit sequence $m_1 m_2 ... m_n$ is judged by whether $(...((0 * m_1) * m_2) * ... * m_n) = 0$ holds.

This method can detect all single-digit errors and all adjacent transposition errors. The proof is as follow. We denote k_i^m as $(...(0 * m_1) * ... * m_i)$.

- single-digit errors

  Suppose digit sequences m and m' are same except for i'th element: $m_i \neq m'_i$.

  $$k_i^m = k_{i-1}^m * m_i$$
  $$k_i^{m'} = k_{i-1}^m * m'_i$$

  From the implication 5.1,

  $$k_i^m \neq k_i^{m'}$$

  is derived.

  $$k_{i+1}^m = k_i^m * m_i$$
  $$k_{i+1}^{m'} = k_i^m * m'_i$$

  From the implication 5.2,

  $$k_{i+1}^m \neq k_{i+1}^{m'}$$

Therefore, inductively,

$$k_n^{m'} \neq k_n^m = 0$$

- adjacent transposition errors

  Suppose a digit sequence m' is a result of swapping m_i and m_{i+1} in a sequence m. Here, we assume $m_i \neq m_{i+1}$.

  $$k_{i+1}^m = (k_{i-1}^m * m_i) * m_{i+1}$$
  $$k_{i+1}^{m'} = (k_{i-1}^m * m_{i+1}) * m_i$$

  From (5.3),

  $$k_{i+1}^m \neq k_{i+1}^{m'}$$

  is derived. Hence, same as the case single-digit errors ,

  $$k_n^{m'} \neq k_n^m = 0$$

Chapter 6

Summary

Content