Encrypting Messages using the Merkle-Hellman Knapsack Cryptosystem

Ashish Agarwal†

Simon Fraser University

Summary
The Merkle-Hellman was invented in 1978 and is based on the superincreasing subset problem sum. This paper demonstrates how to use the cryptosystem to encrypt messages so that only the intended recipient of the message is able to decipher the message.
Key words:
Security, cryptography, cryptosystem, knapsack problem.

1. Introduction

The knapsack problem is an NP complete problem in combinatorial optimization. The knapsack problem selects the most useful items from a number of items given that the knapsack or the rucksack has a certain capacity. Knapsack problems are widely used to model solutions industrial problems such as public-key cryptography.
The 0-1 knapsack problem states that if there is a knapsack with a given capacity and a certain number of items that need to be put in the knapsack. Each item has a value and a weight associated with it. The knapsack problem selects the items that can be put in the knapsack so that the value of all the items is maximized and the weight does not increase the total capacity of the knapsack.
This can be denoted as –

Maximize $\sum_{i=1}^{n} p_i x_i$ (1)

Subject to $\sum_{i=1}^{n} w_i x_i \leq W$ (2)

$$x_i = \begin{cases} 1, & \text{if the item is included in the knapsack} \\ 0, & \text{if the item is not in the knapsack} \end{cases} \quad (3)$$

where,
p is the value associated with each item i
w is the weight associated with each item i
W is the maximum capacity of the knapsack
n is the number of items
The subset sum problem is a special case of the knapsack problem [5]. This problem finds a group of integers from a list vector V, where V = (v1, v2, v3, … vn), the subset of elements in the vector V which have a given sum S. It also determines if a vector X = (x_1, x_2, x_3… x_n) exists where xi element of {0,1} so that V*X = S [5].

Ralph Merkle and Martin Hellman used the subset problem to create a cryptosystem to encrypt data. A superincreasing knapsack vector s is created and the superincreasing property is hidden by creating a second vector M by modular multiplication and permutation. The vector M is the public key of the cryptosystem and s is used to decrypt the message [2].

2. Encrypting Messages

The basic idea behind the Merkle-Hellman encryption scheme is to create a subset problem which can be solved easily and then to hide the superincreasing nature by modular multiplication and permutation. The transformed vector forms the encrypted message and the original superincreasing vector forms the private key and is used to decipher the message.

2.1 Mathematical Explanation

The first step is to choose a superincreasing sequence of numbers of positive integers. A superincreasing sequence is one where every number is greater than the sum of all preceding numbers.
$s = (s_1, s_2, s_3, \ldots. s_n)$ (4)
The second step is to convert all the characters of the message into binary. The binary sequence is represented by the variable b.
The third step is to choose two numbers – an integer (a) which is greater than the sum of all numbers in the sequence s and its co-prime (r).
The sequence s and the numbers a and r collectively form the private key of the cryptosystem.
All the elements – $s_1, s_2, s_3, \ldots. s_n$ of the sequence s are multiplied with the number r and the modulus of the multiple is taken by dividing with the number a.
Therefore, $p_i = r*s_i \bmod(a)$.

All elements $p_1, p_2, p_3, \ldots. p_n$ of the sequence p are multiplied with with the corresponding elements of the

Manuscript received May 5, 2011
Manuscript revised May 20, 2011

binary sequence b. The numbers are then added to create the encrypted message M_i.

$$\therefore M_i = \sum_{i=1}^{n} p_i * b_i \quad (5)$$

The sequence M = (M_1, M_2, M_3… M_n) forms the public key of the cryptosystem.

2.2 Example – Encrypting the string "Hello"

The first step is to choose a superincreasing sequence is chosen. In this case the sequence is –
s = 3, 5, 15, 25, 54, 110, 225

All the characters in the string are then converted into binary –
H = 1001000
e = 1100101
l = 1101100
l = 1101100
o = 1101111
The binary sequence is b = (b_1, b_2, b_3 ….. b_n)
The two numbers chosen are – a is chosen as 439 and r is 10.
The sequence p = p_1, p_2,… p_n
Where $p_i = r*s_i \bmod a$
The message is encrypted by multiplying all the elements of sequence p with the corresponding elements of sequence b and adding the resulting sum.
Therefore, the encrypted message

$$M = \sum_{i=1}^{n} p_i * b_i \quad (6)$$

$p_1 = 3 * 10 \bmod 439 = 30$
$p_2 = 5 * 10 \bmod 439 = 50$
$p_3 = 15 * 10 \bmod 439 = 150$
$p_4 = 25 * 10 \bmod 439 = 250$
$p_5 = 54 * 10 \bmod 439 = 101$
$p_6 = 110 * 10 \bmod 439 = 222$
$p_7 = 225 * 10 \bmod 439 = 55$

Encrypting the character H –
p = (30, 50, 150, 250, 101, 222, 55) and b = (1 0 0 1 0 0 0)
$M_H = 30 + 250 = 280$
Encrypting the character e –
p = (30, 50, 150, 250, 101, 222, 55) and b = (1 1 0 0 1 0 1)
$M_e = 30 + 50 + 101 + 55 = 236$
Encrypting the character l –
p = (30, 50, 150, 250, 101, 222, 55) and b = (1 1 0 1 1 0 0)
$M_l = 30 + 50 + 250 + 101 = 431$
Encrypting the character o –
p = (30, 50, 150, 250, 101, 222, 55) and b = (1 1 0 1 1 1 1)
$M_o = 30 + 50 + 250 + 101 + 222 + 55 = 708$

Therefore, the encrypted message M = (280, 236, 431, 431, 708).

3. Decrypting messages

3.1 Mathematical Explanation

To decrypt the message M, the recipient of the message would have to find the bitstream which satisfies the equation [1]–

$$M = \sum_{i=1}^{n} p_i * b_i \quad (7)$$

To solve the equation (7), the user would need the private key (s, a, r). The first step is to calculate the modular multiplicative inverse of r in r mod a [4]. This is calculated using the Extended Euclidean algorithm. This is denoted by r^{-1}.
The second step is to multiply each element of the encrypted message (M) with r^{-1} mod a.

The largest number in the private key which is smaller than the resulting number is subtracted from the number. This result continues until the number is reduced to zero [1].

3.2 Example: Decrypting the message – 280, 236, 431, 431, 708

Decrypting the first character –

The modular inverse of 10 in 10 mod 439 is calculated using the extended Euclidean algorithms and was calculated to be 44.
The encrypted message M_H is 280 and s = 3, 5, 15, 25, 54, 110, 225.
280 * 44 mod 439 = 28
The largest number in the sequence s, which is smaller than 28 is 25.
28 – 25 = 3
3 – 3 = 0
Therefore, the binary sequence becomes 1 0 0 1 0 0 0. This binary sequence represents the character H.
Decrypting the second character –
The encrypted message M_e is 236 and the sequence s = 3, 5, 15, 25, 54, 110, 225
236 * 44 mod 439 = 287
287 – 225 = 62
62 – 54 = 8
8 – 5 = 3
3 – 3 = 0

Therefore, the binary sequence becomes 1 1 0 0 1 0 1. This binary sequence represents the character e.
Decrypting the third and fourth characters –
The encrypted message is M_l 431 and the sequence s = 3, 5, 15, 25, 54, 110, 225
431 * 44 mod 439 = 87
87 – 54 = 33
33 – 25 = 8
8 – 5 = 3
3 – 3 = 0
Therefore, the binary sequence becomes 1 1 0 1 1 0 0. This binary sequence represents the character l.

Decrypting the fifth character –
The encrypted message M_o is 708 and the sequence s = 3, 5, 15, 25, 54, 110, 225.
708 * 44 mod 439 = 422
422 – 225 = 197
197 – 110 = 87
87 – 54 = 33
33 – 25 = 8
8 – 5 = 3
3 – 3 = 0
Therefore, the binary sequence becomes 1 1 0 1 1 1 1. This binary sequence represents the character o.

Conclusion

This paper explained how to encrypt and decrypt data using the Martin-Hellman knapsack cryptosystem. The cryptosystem was demonstrated by encrypted a string “Hello” and then decrypting it. The decrypted string matched the original string.

References

[1] A.Menezes, P.vanOorschot and S.Vanstone, “Handbook of Applied Cryptography”, CRC Press, 1996
[2] R.Merkle and M.Hellman, “Hiding information and signatures in trapdoor knapsacks”, IEEE Transactions on Information Theory – 24, 5, pp 525 – 530.
[3] W.Diffie and M.Hellman, “New directions in cryptography”, IEEE Transactions on Information Theory – 22, 6, pp 644 – 654.
[4]http://www.mast.queensu.ca/~math418/m418oh/m418oh04.pdf
[5]http://mathworld.wolfram.com/SubsetSumProblem.html

Ashish Agarwal was born in Kanpur, India in 1989. He is currently pursuing is BASc in Computer Engineering from Simon Fraser University in Vancouver, Canada.
He has had a passion for computers since he was a child and his research interests include image processing, cryptography and computer networks.