**Experiment No. 5**

**Title:** Knapsack Cipher

**Batch: Roll No.: Experiment No.:**

**Title:** To implement Knapsack Cipher.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Resources needed:** Windows/Linux OS

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Theory:**

The Merkle-Hellman knapsack cryptosystem was proposed by Merkle and Hellman. Merkle was also one of the founders of public key cryptography. The Merkle-Hellman knapsack cryptosystem is based on a problem that is known to be NP-complete. This seems to make it an ideal candidate for a secure public key cryptosystem. The knapsack problem can be stated as follows.

Given a set of n weights W0,W1, . . . , Wn−1 and sum S, find a0, a1, . . . , an−1, where each

ai є {0, 1}, so that S = a0W0 + a1W1 +…+an−1Wn−1 will form the general knapsack.

A super increasing knapsack, is similar to the general knapsack except that when the weights are arranged from least to greatest, each weight is greater than sum of all previous weights

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Algorithm :**

1. **Key pair generation :**
2. Generate a super increasing knapsack S=(S1,S2,…,Sr).
3. Convert the super increasing knapsack into a general knapsack G=(G1,G2,…,Gr).
4. Choose a multiplier *m* and a modulus *n* so that *m* and *n* are relatively prime and *n* is greater than the sum of all elements in the super increasing knapsack.
5. Then the general knapsack is computed from the super increasing knapsack by modular multiplication: Gi=Si\*m mod n
6. The public key is the general knapsack.
7. The private key is the super increasing knapsack together with the conversion factors
8. **Encryption:**
9. First convert plain text message M into binary equivalent (M1,M2,…,Mn)2.
10. Calculate C=M1G1+M2G2+…….GnMn
11. Display C
12. **Decryption:**
13. Calculate m-1 (multiplicative inverse of m under modulus n)
14. Calculate P’=C m-1 mod n
15. Select weights from super increasing knapsack starting from largest weight so that it will sum up to P’.
16. This will give the binary equivalent of the plaintext, convert it to decimal equivalent and display.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Example :**

1. We’ll choose the superincreasing knapsack

(2*,* 3*,* 7*,* 14*,* 30*,* 57*,* 120*,* 251).

1. To convert the superincreasing knapsack into a general knapsack, we choose a multiplier *m* and a modulus *n* so that *m* and *n* are relatively prime and *n* is greater than the sum of all elements in the superincreasing knapsack. For this example, we choose *m* = 41 and *n* = 491. Then the general knapsack is computed from the superincreasing knapsack by modular multiplication:

2*m* = 2 \* 41 = 82 mod 491

3*m* = 3 \* 41 = 123 mod 491

7*m* = 7 \* 41 = 287 mod 491

14*m* = 14 \* 41 = 83 mod 491

30*m* = 30 \* 41 = 248 mod 491

57*m* = 57 \* 41 = 373 mod 491

120*m* = 120 \* 41 = 10 mod 491

251*m* = 251 \* 41 = 471 mod 491

The resulting general knapsack is *(*82*,* 123*,* 287*,* 83*,* 248*,* 373*,* 10*,* 471*)*, which appears to be a general (non-super increasing) knapsack.

1. The public key is the general knapsack

Public key: *(*82*,* 123*,* 287*,* 83*,* 248*,* 373*,* 10*,* 471*).*

1. The private key is the super increasing knapsack together with the modular inverse of the conversion factor *m*, that is,

Private key: *(*2*,* 3*,* 7*,* 14*,* 30*,* 57*,* 120*,* 251*)* and

*m*−1 mod *n* = 41−1 mod 491 = 12*.*

Suppose Bob’s public and private key pair are given in *step* 3 and *step* 4, above. Then if Alice wants to encrypt the message *M* = 150 for Bob,

1. To encrypt, Alice first converts 150 to binary, that is, 10010110. Then she uses the 1 bits to select the elements of the general knapsack that are summed to give the ciphertext. In this case, Alice finds

*C* = 82 + 83 + 373 + 10 = 548*.*

1. To decrypt this ciphertext, Bob uses his private key to find

*Cm*−1 mod *n* = 548 \* 12 mod 491 = 193

1. Bob then solves the super increasing knapsack for 193. This is an easy (linear time) problem from which Bob recovers the message in binary 10010110 or, in decimal,

*M* = 150.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Questions:**

1. Explain Digital signature using Knapsack with suitable example.
2. Explain any one attack on Knapsack.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Outcomes:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Conclusion :**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of faculty in-charge with date**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**References:**

**Books/ Journals/ Websites:**

* Mark Stamp, “Information security Principles and Practice” Wiley