- Cryptology
  - Merkle-Hellman knapsack cryptosystem
    - Merkle-Hellman additive knapsack cryptosystem
    - Merkle-Hellman multiplicative knapsack cryptosystem
    - Merkle-Hellman multipy-iterated knapsack cryptosystem
  - Advanced knapsack cryptosystems

Additional Research Topics

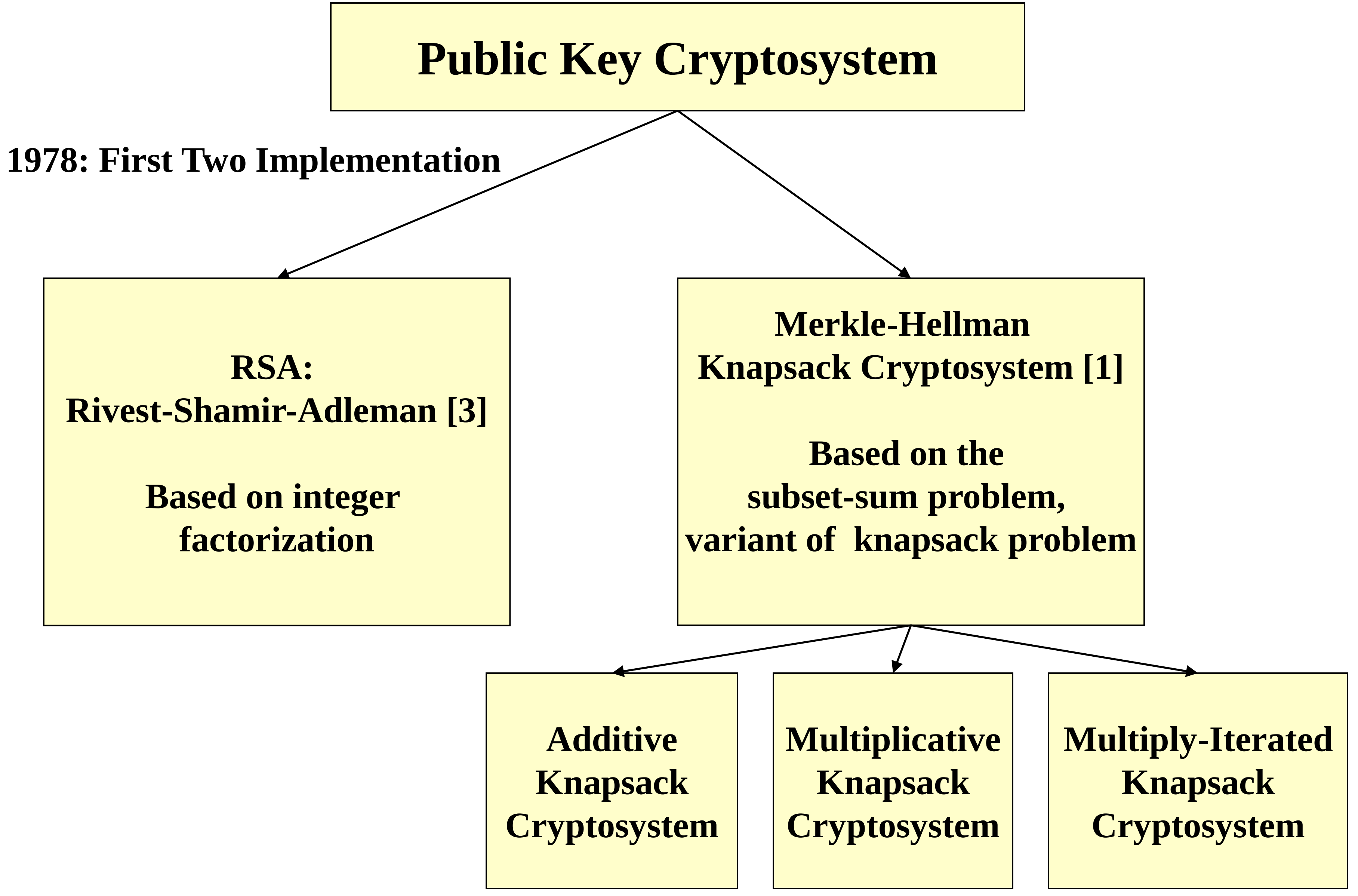
- Data Structures and Algorithms
  - Dynamic Programming Technique
    - Bioinformatics Algorithms.
    - Visualization.
  - Visualization of the Advanced Data Structures and Graph Algorithms
  - Exploring Advanced Sorting Algorithms.
    - Visualization

Public Key Cryptosystem

- In Symmetric or Private Key cryptosystems the encryption and decryption keys are either the same or can be easily found from each other.
- Public Key Cryptosystem (PKC) was introduced in 1976 by Diffie and Hellman [2]. In PKC different keys are used for encryption and decryption.

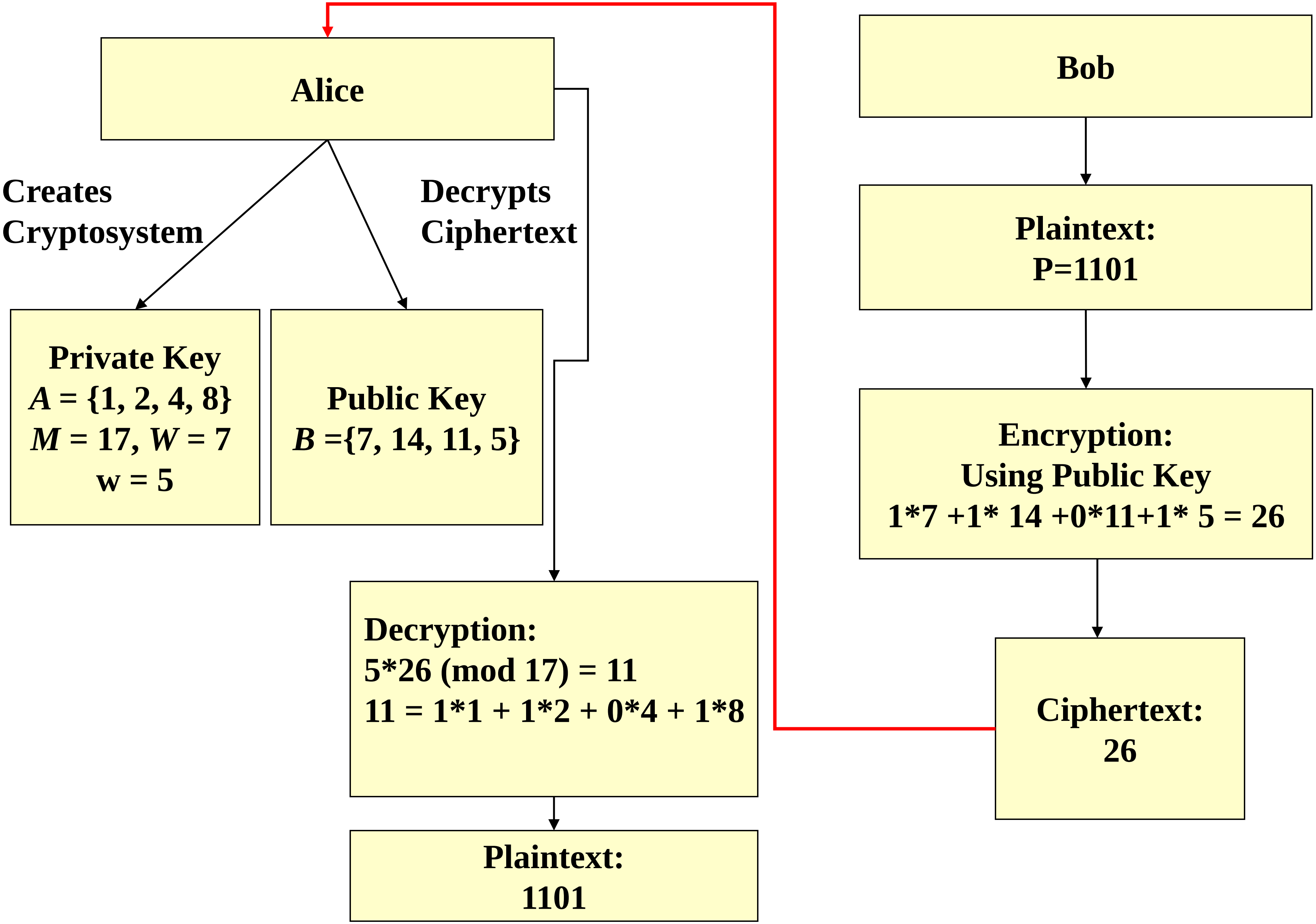
Alice:
1. **Chooses secret (private) key**
2. **Create and publishes public key**
3. **Receives ciphertext**
4. **Decrypts ciphertext using secret key to recover the plaintext – original message**

Bob
1. **Uses Public Key to encrypt the message**
2. **Sends ciphertext – encrypted message to Alice**


Public Key Cryptosystem
1978: First Two Implementation
RSA:
Rivest-Shamir-Adleman [3]
Based on integer factorization
Merkle-Hellman
Knapsack Cryptosystem [1]
Based on the subset-sum problem, variant of knapsack problem
Additive Knapsack Cryptosystem
Multiplicative Knapsack Cryptosystem
Multiply-Iterated Knapsack Cryptosystem

Merkle-Hellman Knapsack Cryptosystem Example

- Alice: Private Key
  - Private Key: $A = \{1, 2, 4, 8\}$, $M = 17$, $W = 7$, w = 5
  - Public Key: $B = \{7, 14, 11, 5\}$
- Bob: Encryption
  - Plaintext 1101
  - Ciphertext = 7 + 14 + 5 = 26
- Alice: Decryption
  - 5*26 (mod 17) = 11
  - 11 = 1*1 + 1*2 +0*4 + 1*8
  - Plaintext: 1101


Alice
Bob
Creates Cryptosystem
Decrypts Ciphertext
Private Key
A = {1, 2, 4, 8}
M = 17, W = 7
w = 5
Public Key
B ={7, 14, 11, 5}
Plaintext:
P=1101
Encryption:
Using Public Key
1*7 +1* 14 +0*11+1* 5 = 26
Ciphertext:
26
Decryption:
5*26 (mod 17) = 11
11 = 1*1 + 1*2 + 0*4 + 1*8
Plaintext:
1101

Merkle-Hellman Knapsack Cryptosystem

- 1982: Single iteration Merkle - Hellman Knapsack Cryptosystem was broken by Adi Shamir [4,5,6]

- 1983: At the CRYPTO ’83 , Adleman used an Apple II computer to demonstrate Shamir’s method [8]

- 1985: Multiple iteration Merkle-Hellman knapsack was broken by Brickell [9], a system of 40 iterations was breaking in about an hour of Cray-1 time

Merkle-Hellman Knapsack Cryptosystem

- History has not been kind to knapsack schemes [11] Lecture Notes on Cryptography, S. Goldwasser, M. Bellare
- Merkle offered $100 award for breaking singly - iterated knapsack
- Singly-iterated Merkle - Hellman KC was broken by Adi Shamir in 1982 [4,5,6] using Hendrik W. Lenstra’s polynomial time algorithm [7] for the integer programming problem when the number of variables is fixed.
- At the CRYPTO ’83 conference, Adleman used an Apple II computer to demonstrate Shamir’s method [8]
- Merkle offered $1000 award for breaking multiply-iterated knapsack
- Multiply-iterated Merkle-Hellman knapsack was broken by Brickell in 1985 [9]

Classical Knapsack Problem

- General 0-1 knapsack problem: given *n* items of different values v_i and weights w_i, find the most valuable subset of the items while the overall weight does not exceed a given capacity W
- The knapsack problem is NP-hard [10]
- The knapsack problem could be solved in pseudo-polynomial time through dynamic programming

Subset-Sum Problem

- Subset – Sum problem is a special case of knapsack problem when a value of each item is equal to its weight
- Input: set of positive integers: $A = \{a_1, a_2, \ldots a_n\}$ and the positive integer S
- Output:
  - TRUE, if there is a subset of A that sums to S and the subset itself
  - FALSE otherwise.
- The subset-sum problem is NP-hard

Easy Knapsack Problem

- An easy knapsack problem is one in which set $A = \{a_1, a_2, \ldots a_n\}$ is a super-increasing sequence
- A super-increasing sequence is one in which the next term of the sequence is greater than the sum of all preceding terms:

  $a_2 > a_1, a_3 > a_1 + a_2, \ldots, a_n > a_1 + a_2 + \ldots + a_{n-1}$
- Example: A= $\{1, 2, 4, 8, \ldots 2^{n-1}\}$ is super-increasing sequence

Polynomial Time Algorithm for Easy Knapsack Problem

- Input: $A = \{a_1, \ldots a_n\}$ is super-increasing sequence, S
- Output: TRUE and P – binary array of n elements, $P[i] = 1$ means: a_i belongs to subset of A that sums to S, $P[0] = 0$ otherwise. The algorithm returns FALSE if the subset doesn't exist

for $i \leftarrow n$ **to** 1

    if $S \geq a_i$

        then $P[i] \leftarrow 1$ **and** $S \leftarrow S - a_i$

    else $P[i] \leftarrow 0$

if S != 0

    then return (FALSE – no solution)

else return ($P[1], P[2], \ldots P[n]$).

Merkle-Hellman Additive Knapsack Cryptosystem

Alice:

1. Constructs the Knapsack cryptosystem
2. Publishes the public key
3. Receives the ciphertext
4. Decrypts the ciphertext using private key

Bob:

1. **Encrypts the plaintext using public key**
2. **Sends the plaintext to Alice**

Alice
Knapsack Cryptosystem Construction

- Chooses $A = \{a_1, \ldots a_n\}$ super-increasing sequence,
  A is a private (easy) knapsack
  $a_1 + \ldots + a_n = E$
- Chooses M - the next prime larger than E.
- Chooses W that satisfies $2 \leq W < M$ and $(W, M) = 1$
- Computes Public (hard) knapsack $B = \{b_1, \ldots .b_n\}$, where $b_i = Wa_i \pmod{M}$, $1 \leq i \leq n$
- **Keeps Private Key:** A, W, M
- **Publishes Public key**: B

Bob – Encryption Process

- Binary Plaintext P breaks up into sets of n elements long: $P = \{P_1, \ldots P_k\}$
- For each set P_i compute $\sum_{j=1}^{n} P_{ij} b_j = C_i$
- $\boldsymbol{C_i}$ is the ciphertext that corresponds to plaintext P_i
- $\boldsymbol{C = \{C_1, \ldots C_k)}$ is ciphertext that corresponds to the plaintext P
- ***C*** **is sent to Alice**

Alice – Decryption Process

- Computes w, the multiplicative inverse of W mod M:
  $wW \equiv 1 \pmod{M}$
- The connection between easy and hard knapsacks:
  $Wa_i = b_i \pmod{M}$ or $wb_i = a_i \pmod{M}$ $1 \le i \le n$
- For each C_i computes: $S_i = wC_i \pmod{M}$

$$S_i = wC_i = w\sum_{j=1}^{n} P_{ij} b_j = \sum_{j=1}^{n} P_{ij} w b_j = \sum_{j=1}^{n} P_{ij} a_j$$

- Plaintext P_i could be found using polynomial time algorithm for easy knapsack

Example

- Alice Private Key:
  - $A = \{1, 2, 4, 8\}$, $M = 17$, $W = 7$, $2 \le W < 17$, $(7, 17) = 1$
- Public Key:

B={7 mod 17, 14 mod 17, 28 mod 17, 56 mod 17}={7, 14, 11, 5}

- Bob Encryption:
  - Plaintext: 1101
  - Ciphertext = 7 + 14 + 5 = 26
- Alice Decryption:
  - $w = 5$ – multiplicative inverse of 7 (mod 17)
  - 5*26 (mod 17) = 11
  - Plaintext: 1101 (11 = 1*1 + 1*2 +0*4 + 1*8)

Ciphertext Only Cryptanalytic Attack on Merkle-Hellman Knapsack: Dynamic Programming Algorithm

- ***Input:*** $B=\{b_1, b_2, \ldots b_n\}$ – public key, C - ciphertext
- ***Output:*** The binary array P – plaintext
- ***Algorithm:*** Let $Q[i, j]$ be TRUE if there is a subset of first i elements of B that sums to j, $0 \le i \le n$, $0 \le j \le C$

Step 1: Computation of P

$Q[0][0] \leftarrow$ TRUE
for $j = 1$ to C do: $Q[0][j] \leftarrow$ FALSE
for $i = 1$ to n do:
  for $j = 0$ to C do:
    if $(j - B[i] < 0)$: $Q[i][j] = Q[i\text{-}1][j]$

    else: $Q[i][j] = Q[i\text{-}1][j\text{-}B[i]]$ *or* $Q[i\text{-}1][j]$

Step 2: Backtracking

Let P be an array of $n + 1$ elements initialized to 0

$i \leftarrow n$, $j \leftarrow C$

```
while i > 0:
    if (j – B[i]) ≥ 0):
        if  (Q[i-1][j-B[i]] is True):
                P[i] ← P[i] + 1
                j ← j – B[i]
        i ← i – 1

    else:  i ← i – 1
```

Output: array P, elements of P that equal to 1 construct a desired subset of B that sums to C

EXAMPLE
Input: B={1, 4, 5, 2}, C =3

	j = 0	j = 1	j = 2	j = 3
i = 0	TRUE	FALSE	FALSE	FALSE
i = 1 B[1] =1	TRUE	TRUE Element is taken	FALSE	FALSE
i = 2 B[2] = 4	TRUE	TRUE	FALSE	FALSE
i = 3 B[3] = 5	TRUE	TRUE	FALSE	FALSE
i = 4 B[4] = 2	TRUE	TRUE	TRUE	TRUE Element is taken

Q[i-1][j-B[i]] *or* Q[i-1][j]

Merkle-Hellman Multiplicative Knapsack Cryptosystem

- Alice:
  - Chooses set of relatively prime numbers
    $P = \{p_1, \ldots p_n\}$ – private (easy) knapsack
  - Chooses prime $M > p_1 * \ldots * p_n$
  - Chooses primitive root b mod M
  - Computes the public (hard) knapsack

  $A = \{a_1, \ldots . a_n\}$, where a_i is discrete logarithm of p_i to base b:

  $1 \le a_i <$ M, such that: $\boldsymbol{p_i \equiv b^{a_i} \pmod{M}}$
  - **Private Key: *P*, *M*, *b***
  - **Public Key: *A***

Merkle-Hellman Multiplicative Knapsack Cryptosystem- Encryption

- Binary Plaintext *T* breaks up into sets of *n* elements long: T = $\{T_1, \ldots T_k\}$
- For each set T_i compute $\sum_{j=1}^{n} T_{ij} a_j = C_i$
- $\boldsymbol{C_i}$ is the ciphertext that corresponds to plaintext T_i
- **$C = \{C_1, \ldots C_k)$** is ciphertext that corresponds to the plaintext *T*
- ***C* is sent to Alice**

Merkle-Hellman Multiplicative Knapsack Cryptosystem- Decryption

- For each C_i computes $S_i \equiv b^{C_i} \pmod{M}$
- S_i is a subset product of the easy knapsack:

$$S_i = b^{C_i} = b^{\sum_{j=1}^{n} T_{ij} a_j} = \prod_{j=1}^{n} b^{T_{ij} a_j} = \prod_{j=1}^{n} (b^{a_j})^{T_{ij}} \equiv \prod_{j=1}^{n} p_j^{T_{ij}} \pmod{M}$$

- $T_{ij} = 1$ if and only if p_j divides S_i

Merkle-Hellman Multiplicative Knapsack Example

- Easy (Private) Knapsack: $P = \{2, 3, 5, 7\}$
- $M = 211$, $b = 17$
- Hard (Public) Knapsack: A= {19, 187, 198, 121}

  $2 \equiv 17^{19} (\text{mod } 211)$, $3 \equiv 17^{187} (\text{mod } 211)$,

  $5 \equiv 17^{198} (\text{mod } 211)$, $7 \equiv 17^{121} (\text{mod } 211)$
- Plaintext: T = 1101
- Ciphertext: C = 327 = 19 + 187 + 121
- Decryption: $S = 42 = 17^{327} (\text{mod } 211)$
- $42 = 2^1 * 3^1 * 5^0 * 7^1$
- Plaintext: 1101

Multiply-Iterated Merkle-Hellman Knapsack Cryptosystem

- $A = \{a_1, \ldots a_n\}$ super-increasing sequence,

  A is a private (easy) knapsack, $a_1 + \ldots + a_n = E$
- For the m-times iterated knapsack cryptosystem: set of m multiplier-modulus pairs (w_i, M_i), $1 \le i \le m$
- To construct a public key knapsack: $\boldsymbol{B = \{b_1^m, b_2^m, \ldots, b_n^m\}}$

$$w_1 b_i^1 \equiv a_i (\mathrm{mod}\, M_1),\ 1 \le i \le n,\ M_1 > E$$

$$w_2 b_i^2 \equiv b_i^1 (\mathrm{mod}\, M_2), 1 \le i \le n,\ M_2 > \sum_{i=1}^{n} a_i^1$$

.....

$$w_m b_i^m \equiv b_i^{m-1} (\mathrm{mod}\, M_m), 1 \le i \le n,\ M_m > \sum_{i=1}^{n} a_i^{m-1}$$

Multiply-Iterated Merkle-Hellman Knapsack Cryptosystem Example

- A={1, 2, 4, 8}- super-increasing sequence (easy) knapsack, $m = 3$ (number of iterations)
- 1st iteration: $M_1 = 17$, $W_1 = 7$, $w_1 = 5$

  B^1 ={7 mod 17, 14 mod 17, 28 mod 17, 56 mod 17}={7, 14, 11, 5}
- 2nd iteration: $M_2 = 41$, $W_2 = 18$, $w_2 = 16$

  B^2 ={126 mod 41, 252 mod 41, 198 mod 41, 90 mod 41}={3, 6, 34, 8}
- 3rd iteration: $M_2 = 53$, $W_2 = 25$, $w_2 = 17$

  B^3 ={75 mod 53, 150 mod 53, 850 mod 53, 200 mod 53}={22, 44, 2, 41}
- **Public Key: {22, 44, 2, 41}**

REFERENCES

1. R. C. Merkle, M. E. Hellman, Hiding Information and Signatures in Trapdoor Knapsacks, IEEE Transactions on Information Theory, vol. IT-24, 1978, pp. 525-530.
2. W. Diffie, M. E. Hellman, New Directions in Cryptography, IEEE Transactions on Information Theory, vol. IT-22, no. 6, November 1976, pp. 644-654.
3. R. L. Rivest, A. Shamir, and L. M. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM, vol. 21, no. 2, 1978, pp. 120-126
4. Adi Shamir. A Polynomial-time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem. Proceedings of the IEEE Symposium on Foundations of Computer Science. IEEE, New York, 1982, pp. 145-152.
5. Adi Shamir. A Polynomial Time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem. In David Chaum, Ronald L. Rivest, Alan T. Sherman. editors, Advances in Cryptology – CRYPTO '82. Plenum, New York, 1983.
6. Adi Shamir. A Polynomial-time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem. IEEE Transactions on Information Theory, vol. IT-30, no. 5, September 1984, pp. 699-704.

REFERENCES

7. Hendrik W. Lenstra Jr, Integer Programming with a Fixed Number of Variables, Mathematics and Operations Research, vol. 8, no. 4, 1983, pp. 538-548

8. Ming Kin Lai, Knapsack Cryptosystems: The Past and the Future, http://www.cecs.uci.edu/~mingl/knapsack.html

9. Ernest F. Brickell, Breaking Iterated Knapsacks. In G. R. Blakley, David C. Chaum, editors, Advances in Cryptology – CRYPTO '84, Lecture Notes in Computer Science, vol. 196. Springer, Berlin, 1985, pp. 342-358.

10. M. Carey and D.S. Johnson, Computers and Intractability: A guide to the Theory of NP-Completeness, Freeman, 1979

11. Lecture Notes on Cryptography, S. Goldwasser, M. Bellare

12. J. C. Lagarias, Performance Analysis of Shamir's Attack on the Basic Merkle-Hellman Knapsack Cryptosystem. Proceedings of the 11th International Colloquium on Automata Languages and Programming, Lecture Notes in Computer Science, vol. 172. Springer, Berlin, 1984.

13. A. M. Odlyzko. The Rise and Fall of Knapsack Cryptosystems. In Carl Pomerance, editor, Cryptology and Computational Number Theory, Proceedings of Symposia in Applied Mathematics, vol. 42. American Mathematics Society, Providence, RI, 1990, pp. 75-88, http://www.dtc.umn.edu/~odlyzko/doc/complete.html

14. A. M. Odlyzko. Cryptanalytic Attacks on the Multiplicative Knapsack Cryptosystem and on Shamir's Fast Signature Scheme. IEEE Transactions on Information Theory, IT-30, 1984, pp. 594-601, http://www.dtc.umn.edu/~odlyzko/doc/complete.html