Basic Merkle-Hellman Knapsack cryptosystem Crypto analysis by Shamir

By: Kanmogne Pekam Linda

Introduction

- In 1976 the idea of public key cryptosystem was introduced by Diffie and Hellman
- In 1978 Merkle-Hellman Knapsack public key Cryptosystem is published
- in 1982 Adi Shamir's broke the basic Merkle-Hellman Knapsack Cryptosystem

NP-Complete

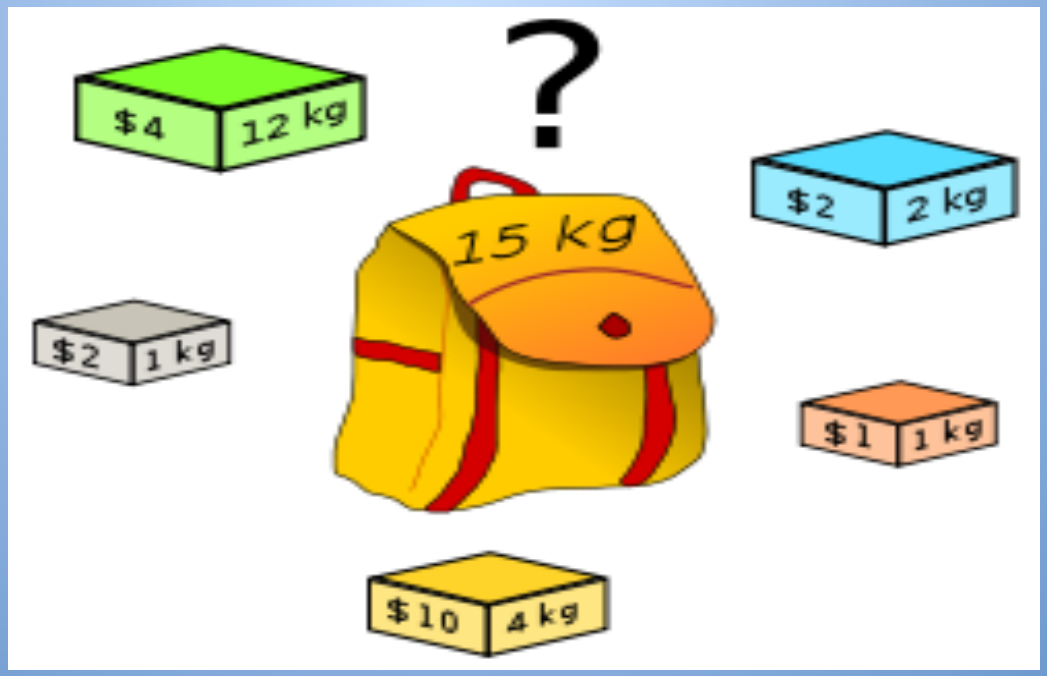
A problem Z is said to be NP-Complete if:

- Z is NP: Meaning there is a nondeterministic turing machine that can solve the problem in polynomial time

- And Z is NP-Hard: every NP problem R can be reduced to Z.

knapsack problem

Which coins should we put in the bag such that the total value of the bag is a big as possible but the total weight at most 15 kg?



Knapsack problem: formal definition

- input: n items, u_i and $w_i \in Z+$, $1 \leq i \leq n$, are resp value and weight of ith item, and the sum $W \in Z+$
- Problem: if exists vector $x = (x_1,...,x_n)$, $x_i \in \{0,1\}$ such that:
  - Maximize $\sum u_i x_i$ for i: 1,...,n
  - Subject to $\sum w_i x_i \leq W$ for i: 1,...,n

This problem is known to be NP-Complete

Super increasing sequence

Vector sequence $a = (a_1, a_2, \ldots, a_j, \ldots, a_n)$ is said to be super increasing if :

$$a_j > \sum a_i \text{ for } i: 1,\ldots, j-1, \text{ with } j \leq n.$$

Example:

- (1, 2, 4, 6) is not super increasing because $6 \not> 1+2+4$
- (1, 2, 4, 8) is super increasing because $8 > 1+2+4$

Easy and Hard Knapsack

- A Knapsack problem is easy if the knapsack vector w = $(w_1,...,w_n)$ weights of the n items

  form a super increasing sequence :

  => $w_j > \sum w_i$ for i: 1,…, j-1, with $j \leq n$.

  => $\sum w_i x_i =$ W is solvable in polynomial time
- The knapsack vector is hard otherwise and then finding x_i is an NP-Complete problem

Subset-sum problem

It is a particular case of Knapsack problem

- Giving n items with weight vector
  $w = (w_1, w_2, \ldots, w_n)$, $w_i \in Z+$, for i : 1,...,n
- and $S \in Z+$ the sum.
- find subset $\mathbf{w_j'}$ of $\mathbf{w_i}$ such that
  $S = \sum \mathbf{w_j'}$ for j: 1,…,p ($p \leq n$)

=> finding vector $x = (x_1,...,x_n)$, $x_i \in \{0, 1\}$ st:

$S = \sum w_i x_i = w_1x_1 + w_2x_2 + ... + w_nx_n$ for i: 1,...,n

if $x_i = 1 : w_i \in w_j'$, else , $w_i \notin w_j'$

Subset-Sum problem - 2

The subset-sum problem (w, S) is known to be NP-Complete

However if the initial weight vector **w** has a super increasing, the problem (w, S) can be solved in O(n).

Solving Super increasing knapsack

- input:

- n items, super increasing weights vector

  $w = (w_1, w_2, \ldots, w_n)$

- Sum $S \in Z+$

- Problem: find vector $x = (x_1, \ldots, x_n)$, $x_i \in \{0, 1\}$

such that $S = \sum w_i x_i = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n$

Solving Super increasing knapsack

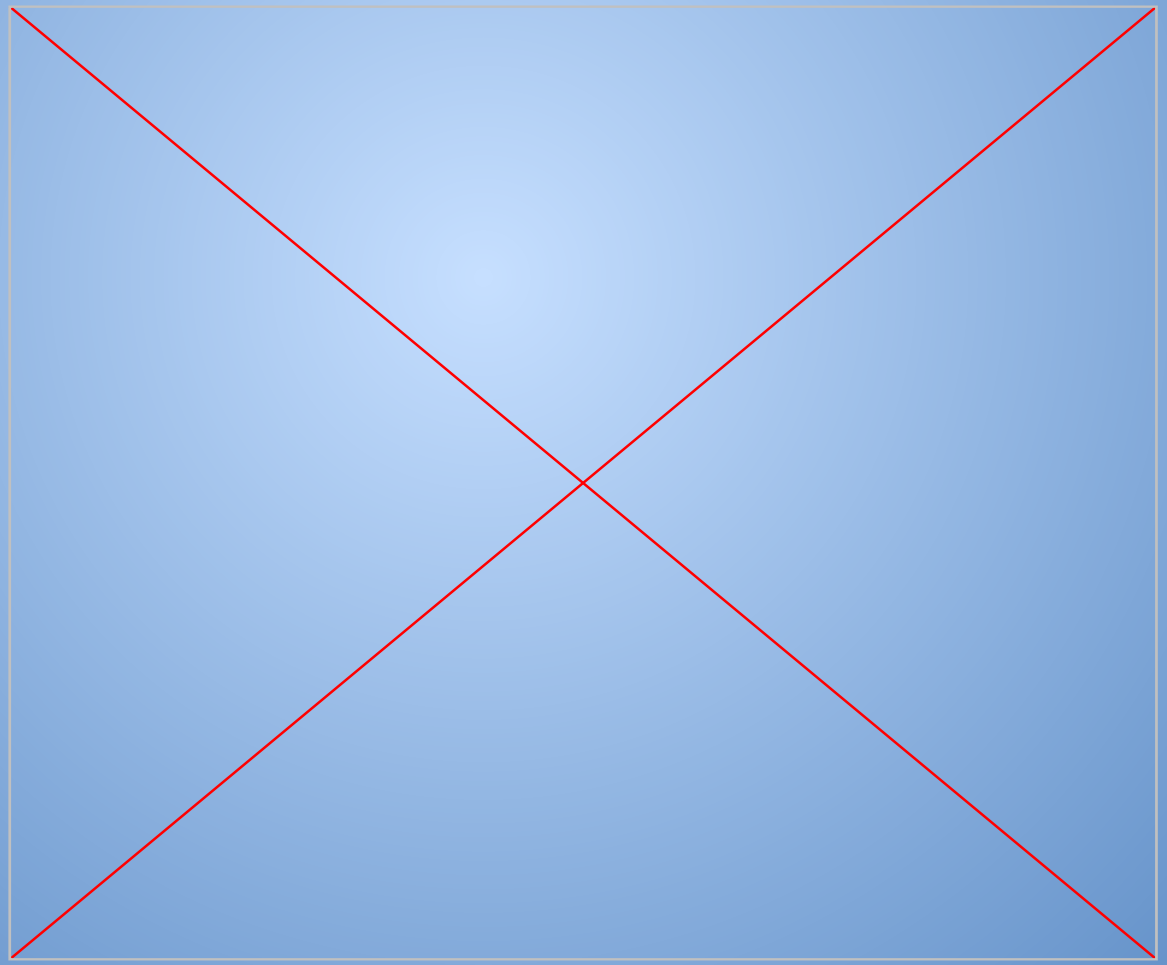
- Algorithm to solve a Subst-Sum problem with a super increasing weights vector:

  for i = n downto 1
  { If $S \geq w_i$ then { $x_i = 1$; $S = S - w_i$; } else $x_i = 0$; }
  return $(x_1, x_2, \ldots, x_n)$.

  Solution if exists is unique!
  Running time: O(n)

Merkle-Hellman Knapsack Cryptosystem: Idea

Encoded message as solution to knapsack problem.



MH -> Key Generation

- n-bit message
- Choose a super increasing vector $\mathbf{a_i : \{a_1, a_2, \ldots, a_n\}}$
- Choose a number q such that $q > \sum a_i$ for $1 \leq i \leq n$. **q** is call the ***modulus***
- choose a number r such that r and q are coprime: gcd (r,q) = 1. **r** is called the ***multiplier.***

MH -> Key generation -2

Now we compute the vector b_i : (b_1, b_2,…, b_n) such that: $b_i = r\, a_i \bmod(q)$, $0 \leq b_i < q$

The keys:

Public key: is b_i

Private key: is (a_i, q, r)

MH -> Encryption

- n-bit message m_i : { m_1 , m_2 ,…, m_n}
- Public key b_i : {b_1, b_2,…, b_n}
- Encrypted message is: $C = \sum m_i b_i$ (E)

  for $1 \leq i \leq n$, with $0 \leq C < q$

(E) is NP-Complete knapsack problem : b_i is a hard-Knapsack

MH -> Decryption

- Private key: (a_i, q, r).
- Message integer $C = \sum m_i b_i$ for $1 \leq i \leq n$.
- Compute r^{-1} inverse of r modulo q using the Extended Euclidean Division

MH -> Decryption -2

We compute $C' = C\, r^{-1} \bmod(q)$

$\Rightarrow C' = \sum m_i\, b_i\, r^{-1} \bmod(q)$, with $b_i = r\, a_i \bmod(q)$

$\Rightarrow a_i = r^{-1}\, b_i \bmod(q)$

$\Rightarrow C' = \sum m_i\, a_i \bmod(q)$

$q > \sum a_i$ and $mi \in \{0,1\}$, $\sum m_i\, a_i < q$

$\Rightarrow C' = \sum m_i\, a_i$ (E')

(E') easy to solve as a_i has a super increasing.

MH -> Example

- Message "hello", n = 7 bits
- a_i : {3, 5, 15, 25, 54, 110, 225} i: 1,...,7
- q > $\sum a_i$ => q = 439 and r = 10
- $b_i = a_i$ r mod (q) => b_i : {30, 50, 150, 250, 101, 222, 55)}
- Encryption:

h = 1001000 => $C_h = \sum h_i b_i$ = 30 + 250 = 280

e = 1100101 => $C_e = \sum e_i b_i$ = 30 + 50 + 101 + 55 = 236

MH -> Example - 2

l = 1101100 => $C_l = \sum l_i b_i$ = 30 + 50 + 250 + 101 = 431

o = 1101111 => $C_l = \sum l_i b_i$ = 30 + 50 + 250 + 101 + 222 + 55 = 708

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

- Decryption of C_h = 280

  r^{-1} of r modulo q is 44 (10x44 = 1 mod(q))

MH -> Example - 3

$C_h' = C_h\ r^{-1} \bmod (q)$

$\Rightarrow C_h' = 280 \times 44 \bmod (439) = 28$

a_i : {3, 5, 15, 25, 54, 110, 225}

- The largest element of $a_i \leq C_h'$ is 25 $\Rightarrow h_4 = 1$

$C_h' = 28 - 25 = 3$

$a_1 \leq C_h' \Rightarrow h_1 = 1, C_h' = 3 - 3 = 0$

$\Rightarrow h_i$: 1001000

Algo to solve the super increasing knapsack problem:
for j = n downto 1
{ If s ≥ a_i then { xi = 1; s = s - ai; } else xi = 0; }
return (x1, x2,..., xn).

MH- Crypto analysis: Assumptions

- $n \longrightarrow \infty$
- d: expansion factor: Ratio between size of the ciphertext over the size of the plaintext . d > 1 is fixed
- d = 2 for the Basic MH : M = 200, n = 100
- q_0 and b_i grow linearly with n
- $a_1 \cong 2^{-n+1}$, $a_i \cong 2^{-n+i-1}$
- $q_0 \cong 2^{-dn}$, $b_i < q_0$

MH- Crypto analysis: Trapdoor pair

Shamir algorithm find trapdoor pair w and q, with $w = r^{-1} \bmod (q)$ such that given the public key b_i, we can compute a super increasing vector s_i st:

$$s_i = w\, b_i \bmod (q) \text{ and with } \quad q > \sum s_i$$

There is at least one solution by construction w_0, q_0!!

MH- Crypto analysis: Trapdoor pair -2

(w, q) can be different from (w_0, q_0), but will still decrypt the message.

Proof:

- encrypted message $C = \sum m_i b_i$

- $C' = C\ w \bmod (q)$ with $s_i = w\ bi \bmod (q)$

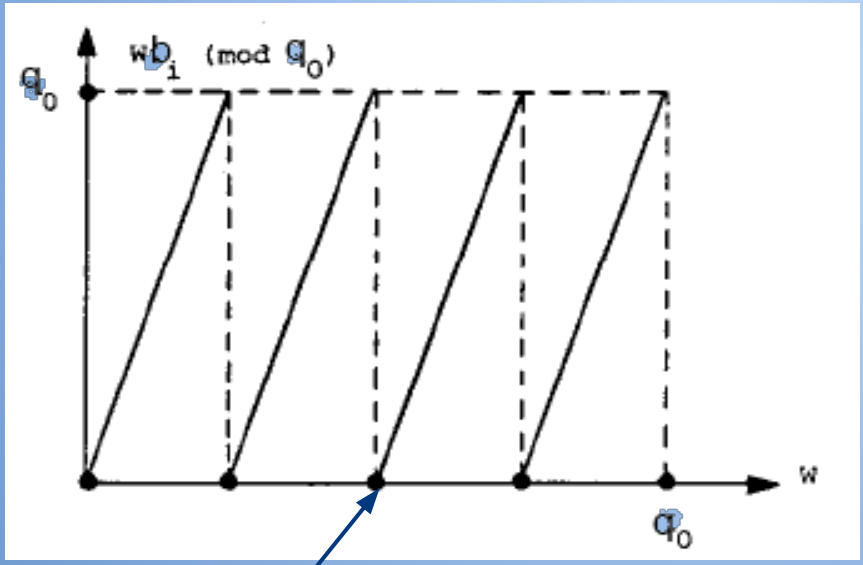
$\Rightarrow C' = \sum m_i b_i\ w \bmod(q) = \sum m_i s_i \bmod (q)$

$\Rightarrow C' = \sum m_i s_i$ (D), $q > \sum s_i$, m_i: {0,1}

(s_i super increasing => (D) easy to solve)

MH- Cryptanalysis: Step 1

- q_0 is unknown
- We study the curves : w bi mod(q_0), i:1..n



Minimum: w bi mod(q_0) = 0

for real multipliers
$0 \leq w < q_0$,
w bi mod(q_0) graph
has a sawtooth form

MH- Cryptanalysis: Step 1 - 2

- The slope of the sawtooth curves is b_i
- Minimum is reached when $b_i w = x q_0$

=> $w = x q_0 / b_i$, with $0 \leq w < q_0$

=> $0 \leq x < b_i - 1$: there is b_i minima

- distance between two successive minima is q_0 / b_i
- for i = 1, w_o is such that $a_1 = w_o b_1 \bmod(q_0)$

=> $a_1 = w_o b_1 - x q_0$,

MH- Cryptanalysis: Step 1 - 3

$=>a_1/b_1 = w_o - xq_0/b_1$, $a_1 \cong 2^{dn-n}$, $b_1 < q_0 \cong 2^{dn}$

`$=> w_o - xq_0/b_1 \cong 2^{-n}$: distance between w_o and the xth closest minimum to the left w^x_1 of the b_1 curves is at most 2^{-n}.

- w_o and w^x_1 are closed to each other

MH- Cryptanalysis: Step 1 - 4

- Similarly for i = 2, the distance between w_o and the *yth* closest minimum to the left of the b_2 curves w^y_2 is at most 2^{-n+1}
- w_0 and w^y_2 are also really closed

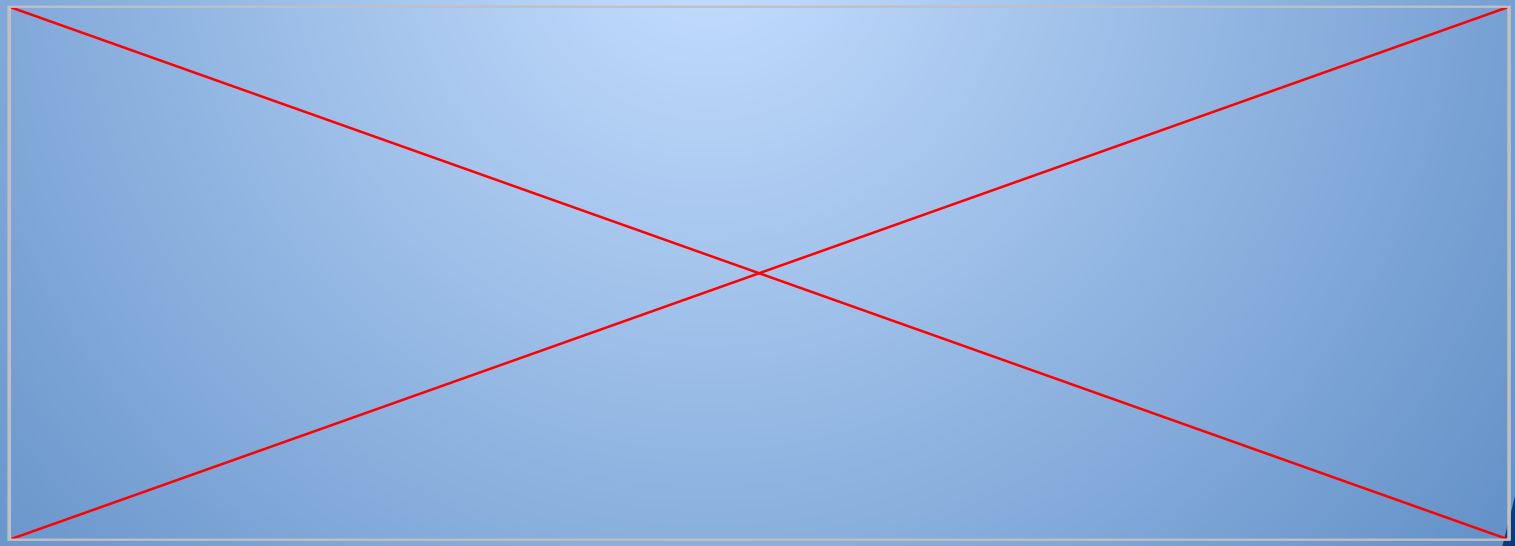
=> w^x_1 and w^y_2 are also closed

=> distance factor between w_0 and closest minimum to left of the *ith* curves is : 2^{-n+i-1}

=> There is a point where all the minima at are closed to each other.

MH- Cryptanalysis: Step 2

If we superimpose b_i curves, there would an interval (s) where all minina of b_i curves are closed to each other meaning closed to w_0



MH- Cryptanalysis: Step 2 - 2

- So instead of finding w_o, we can compute the accumulation points of the super imposes b_i curves
- Choose l out of n curves to superimpose. what is should be the value of l?
- Shamir proved that l is not dependant on n but instead on the factor d = M/n
- l = d+2 is enough to estimate the accumulation points

MH - Cryptanalysis : Step 3

- We pick l b_i curves
- the pth minimum of b_1 is pq_0/b_1,
- we don't have q_0
- Observation: the location of accumulation points depend on b_1 and not on tq_0

MH-Cryptanalysis : step 3 - 2

- we can get rid of q_0 by dividing the function by q_0

=> $b_i v \bmod(1)$ with $v = w/q_0$, $0 \leq v < 1$

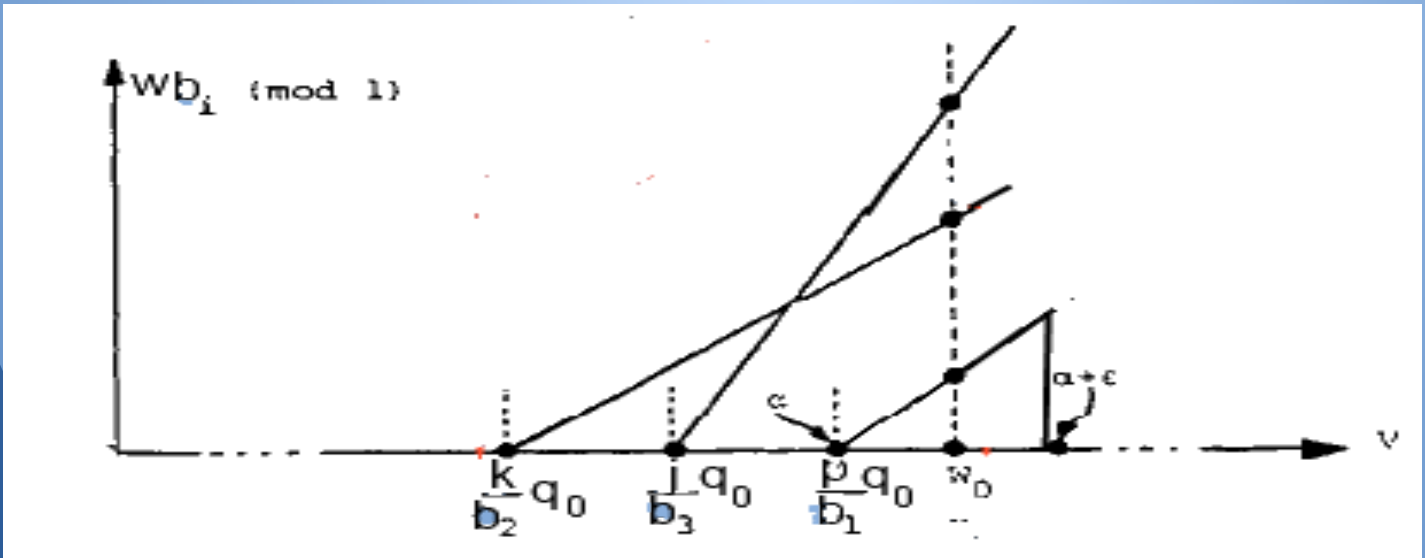
=> slope is unchanged: b_i

=> the distance between two consecutives minima : $1/b_i$

=> distance between w_o and the b_i minima will be reduced by 2^{dn}, => $v_o - v_i \leq 2^{-dn-n+i-1}$

MH-Cryptanalysis : step 3 - 3

for i=1, the pth minimum of b_1 curve is an accumulation point if it is closed enough to all other neighboring b_i minima



MH-Cryptanalysis : step 3 - 4

=> This gives the following system :

- (l-1) inequalities equations
- l unknow value p, q , r…integers
- ϵ , ϵ’ : allowable deviation between pth minimum and other minima.

$-\epsilon_2 < p/b_1 - q/b_2 < -\epsilon_2'$ $\quad 1 \le p < b_1\text{-}1,\ 1 \le q < b_2\text{-}1$

$-\epsilon_3 < p/b_1 - r/b_3 < -\epsilon_3'$ $\quad 1 \le p < b_1\text{-}1,\ 1 \le r < b_3\text{-}1$

...

MH-Cryptanalysis : step 3 - 5

Multiplying the inequalities by their denominators gives:

$$-\epsilon_2 < pb_2 - qb_1 < -\epsilon_2' \quad 1 \le \mathbf{p} < b_1\text{-}1,\ 1 \le \mathbf{q} < b_2\text{-}1$$

$$-\epsilon_3 < pb_3 - rb_1 < -\epsilon_3' \quad 1 \le \mathbf{p} < b_1\text{-}1,\ 1 \le \mathbf{r} < b_3\text{-}1$$

...

Since the number of variable is fixed , We can apply the Lenstra's algorithm to find p values. running time is $\mathbf{O(n^{F(l)})}$, F(l) grows exponentially with l, but l is small (l = 4 for the Basic MH) .

MH-Cryptanalysis : step 3 - 6

Using the Lenstra integer programming will output all possible value of p, satisfying the inequalities system

The number of accumulation points k should not exceed 100 else the algorithm is aborted. This condition make sure the algorithm runs in polynomial time.

Example: all b_i are similar => all minima are accumulation points

MH-Cryptanalysis : step 4

∀ p found in step 3:

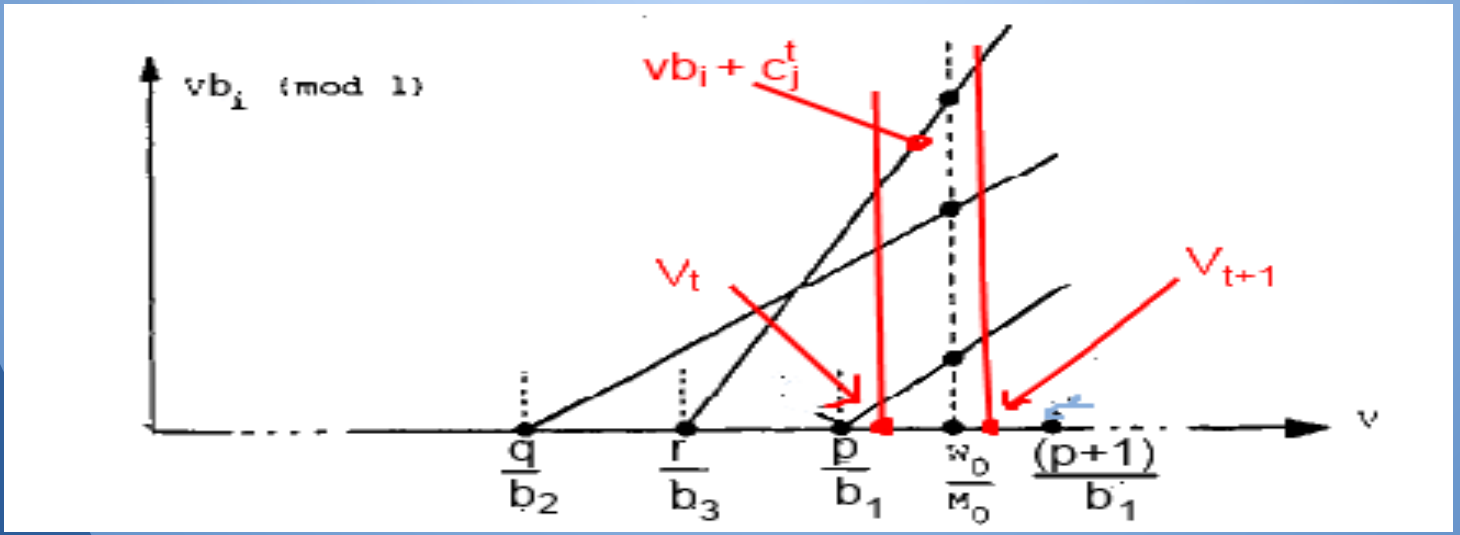
- $[p/b_1, (p+1)/b_1]$: interval between 2 successive minima of b_1
- $v_1,...,v_1$: the list of coordinates of discontinuity points of all n curves lying in the sorted order in this interval
- We divide $[p/b_1, (p+1)/b_1]$ in subintervals such as $[v_t, v_{t+1})$.

MH-Cryptanalysis : step 4 - 2

- in $[v_t, v_{t+1})$, each b_i curves is a line segment.

=> the ith linear segment : $vb_i + C^t_i$ where C^t_i

: number of b_i minima in $(0, v_t]$, $v_t \le v < v_{t+1}$



MH-Cryptanalysis : step 4 - 3

- $v = C^t_i / b_i$
- Conditions: v trapdoor ratio w/q if:
- modulus Size: $\sum(vb_i + C^t_i) < 1$ i: 1,...,n
- Superincreasing: $(vb_i + C^t_i) > \sum(vb_j + C^t_j)$

for i: 2,...,n and j: 1,...,i-1

The solution to this system of linear inequalities in v, is possible non empty subinterval of $[v_t, v_{t+1})$.

MH-Cryptanalysis : step 4 -4

There would be at least one non empty subintervals by construction

The membership of w/q to this subinterval for some p, t value is a ***necessary*** and ***sufficient*** condition for w and q to be a *trapdoor pair*.

MH-Cryptanalysis : step 5

- We have the ratio (s) w/p = k, with k real value
- We need w, p

Diophantine approximation: For a given real value k, output the rational number w/q such that w/q is an approximated value of k.

- With w, q, and b_i we can compute s_i
- The new private key is then (s_i, q, w)

MH- Crypto analysis: Performance

The most consuming part of the algorithm is the Lenstra's algorithm to find p values. running time is **polynomial time in n but exponential in l.**

Thank you for your attention!!!