MATRIX CRYPTOGRAPHIC PROTOCOL

Student : Guan-Cheng Chien (數三 簡冠程)
Advisor : Prof . Pin-Chi Hung (洪斌哲 教授)

Abstract:
Diffie-Hellman key exchange is a method of securely exchanging cryptographic keys over a public channel and one of the first public-key protocols as conceived by Ralph Merkle and named after Whitfield Diffie and Martin Hellman, which wins Turing Award in 2015. The method was followed shortly afterwards by RSA, an implementation of public-key cryptography using asymmetric algorithms, which wins Turing Award in 2002.In this project, we introduce a new method based on **matrices** and **discrete logarithm** to construct a exchanging protocol .

Notation:

Let L, N, X, Y and $G \in M_n(\mathbb{Z}_p)$ (p is a prime number) and let t and $s \in \mathbb{N}$.

We put $\Psi_G(L,N) = \begin{pmatrix} L & G \\ O & N \end{pmatrix}$

Define $(\Psi_G(L,N))^t = \begin{pmatrix} L & G \\ O & N \end{pmatrix}^t \equiv \begin{pmatrix} L^t & \Psi_G(L,N)_t \\ O & N^t \end{pmatrix}$

and $\begin{pmatrix} X & \Psi_G(L,N)_t \\ O & Y \end{pmatrix}^s \equiv \begin{pmatrix} X^k & \Psi_G(L,N)_{t,s} \\ O & Y^k \end{pmatrix}$

Lemma:

$$\Psi_G(L,N)_t = \sum_{i=0}^{t-1} L^{t-1-i} G N^i, \forall t \geq 2$$

Proof:

For $t = 2$,

$$\Psi_G(L,N)_2 = \sum_{i=0}^{2-1} L^{2-1-i} G N^i = \sum_{i=0}^{1} L^{1-i} G N^i = LG + GN$$

$$(\Psi_G(L,N))^2 = \begin{pmatrix} L & G \\ O & N \end{pmatrix}^2 = \begin{pmatrix} L^2 & LG+GN \\ O & N^2 \end{pmatrix}$$

Assume that

$$\Psi_G(L,N)_k = \sum_{i=0}^{k-1} L^{k-1-i} G N^i, \forall k > 2$$

Then

$$(\Psi_G(L,N))^{k+1} = (\Psi_G(L,N))^k (\Psi_G(L,N))$$

$$= \begin{pmatrix} L^k & \Psi_G(L,N)_k \\ O & N^k \end{pmatrix} \begin{pmatrix} L & G \\ O & N \end{pmatrix} = \begin{pmatrix} L^k & \sum_{i=0}^{k-1} L^{k-1-i} G N^i \\ O & N^k \end{pmatrix} \begin{pmatrix} L & G \\ O & N \end{pmatrix}$$

$$= \begin{pmatrix} L^{k+1} & L^k G + \sum_{i=0}^{k-1} L^{k-1-i} G N^i N \\ O & N^{k+1} \end{pmatrix}$$

$$= \begin{pmatrix} L^{k+1} & L^k G N^0 + \sum_{i=0}^{k-1} L^{k-(i+1)} G N^{i+1} \\ O & N^{k+1} \end{pmatrix} = \begin{pmatrix} L^{k+1} & \sum_{i=0}^{k} L^{k-i} G N^i \\ O & N^{k+1} \end{pmatrix}$$

Hence by Mathematical Induction , the lemma is proved.

Theorem:

If $LX = XL$ and $NY = YN$, then $\Psi_G(L,N)_{t,s} = \Psi_G(X,Y)_{s,t}$

Proof:
By the above lemma , it follows that

$$\Psi_G(L,N)_{t,s} = \sum_{j=0}^{s-1} X^{s-1-j} \Psi_G(L,N)_t Y^j = \sum_{j=0}^{s-1} X^{s-1-j} \sum_{i=0}^{t-1} L^{t-1-i} G N^i Y^j = \sum_{j=0}^{s-1}\sum_{i=0}^{t-1} X^{s-1-j} L^{t-1-i} G N^i Y^j$$

Similarly ,

$$\Psi_G(L,N)_{s,t} = \sum_{i=0}^{t-1}\sum_{j=0}^{s-1} L^{t-1-i} X^{s-1-j} G Y^j N^i$$

Thus $\Psi_G(L,N)_{t,s} = \Psi_G(X,Y)_{s,t}$ if $LX = XL$ and $NY = YN$.

Exchanging protocol:

Step1. Publish $\Psi_G(\bigcirc, \blacktriangle) = \begin{pmatrix} \bigcirc & G \\ O & \blacktriangle \end{pmatrix}$ as public key for any $\bigcirc$, $\blacktriangle$, O and $G \in M_n(\mathbb{Z}_p)$. (O is a zero matrix and G is fixed)

Step2.

1) Alice choose a private key : $t \in \mathbb{N}$, $L \in M_n(\mathbb{Z}_p)$, and published $Z(L) = \{X \in M_n(\mathbb{Z}_p) : LX = XL\}$.
2) Bob choose a private key : $s \in \mathbb{N}$, $Y \in M_n(\mathbb{Z}_p)$, and published $Z(Y) = \{N \in M_n(\mathbb{Z}_p) : NY = YN\}$.

($p(L)$ *and* $p(Y)$ *for some* $p(x) \in \mathbb{Z}[x]$ *, so they are nonempty*)

Step3.

1) Alice choose an other private key : $N \in Z(Y)$.
   After that ,she compute $(\Psi_G(L,N))^t = \begin{pmatrix} L^t & \Psi_G(L,N)_t \\ O & N^t \end{pmatrix}$, and send $\Psi_G(L,N)_t$ to Bob .
2) Bob choose an other private key : $X \in Z(L)$.
   After that ,he compute $(\Psi_G(X,Y))^s = \begin{pmatrix} X^s & \Psi_G(X,Y)_s \\ O & Y^s \end{pmatrix}$, and send $\Psi_G(X,Y)_s$ to Alice .

Step4.

1) With t, L, N and $\Psi_G(X,Y)_s$, Alice can compute $\begin{pmatrix} L^t & \Psi_G(X,Y)_{s,t} \\ O & N^t \end{pmatrix}$
2) With s, X, Y and $\Psi_G(L,N)_t$, Alice can compute $\begin{pmatrix} L^t & \Psi_G(L,N)_{t,s} \\ O & N^t \end{pmatrix}$

By the above theorem , $\Psi_G(L,N)_{t,s} = \Psi_G(X,Y)_{s,t}$ since $LX = XL$ and $NY = YN$.

Encryption :

Let key $\Psi = \Psi_G(L,N)_{t,s} = \Psi_G(X,Y)_{s,t}$ and p be a plaintext , then the ciphertext is

$$c = E_k(m) \equiv \Psi^{-1} m \Psi$$

Decryption :

$$D_k(m) \equiv \Psi c \Psi^{-1}$$

Then

$$D_k(c) = D_k(E_k(m)) = \Psi E_k(m) \Psi^{-1} = \Psi\Psi^{-1} m \Psi\Psi^{-1} = m$$

Complexity :

If another person wants to know Ψ , he must solve

$$\Psi_G(L,N)_t = \sum_{i=0}^{t-1} L^{t-1-i} G N^i$$

or

$$\Psi_G(X,Y)_s = \sum_{j=0}^{s-1} X^{s-1-j} G Y^s$$

but t, L and N are unknown; likewise, s, X and Y .
Moreover , the complexity is just like to solve **discrete logarithm in matrix form** as we set any matrix over field $\mathbb{Z}_p$.

Conclusion :

``Can the discrete logarithm be computed in polynomial time on a classical computer?'' This is an unsolved problem in computer science . We turned the open problem into matrices to construct a protocol.

Reference :

1) Johannes A. Buchmann *Introduction to Cryptography*
2) Horn, Roger A.; Johnson,Charles *Matrix Analysis*