Giới thiệu về Diffie-Hellman key exchange

Long Nguyen

1 Giới thiệu

Anh chị em làm trong lĩnh vực IT thì cũng đã vài lần động chạm tới public-private key rồi. Một trong use case hay sử dụng nhất đó chính là dùng key này để ssh vào trong server, hoặc là để clone từ git về.

Thế nhưng anh chị em đã bao giờ nghĩ về việc khi mà chúng ta gửi tin nhắn thông qua messenger, hoặc là chính trên MS Teams mà công ty chúng ta mới chuyển qua sử dụng thì việc mã hoá này hoạt động như thế nào chưa?

> GenAI section:
>
> Một cách tự nhiên, chúng ta có thể nghĩ đến việc sử dụng public-private key để mã hoá tin nhắn. Tuy nhiên, việc này không thể thực hiện được vì một số lý do như sau:
>
> - Kích thước của public-private key quá lớn, việc mã hoá tin nhắn bằng cách này sẽ tốn nhiều băng thông.
> - Việc mã hoá tin nhắn bằng cách này sẽ tốn nhiều thời gian.

Vậy nên, một cách khác để mã hoá tin nhắn là sử dụng một thuật toán mã hoá khác, đó chính là **Diffie-Hellman key exchange**. Hầu hết điện thoại thông mình của chúng ta đều đang sử dụng thuật toán này để mã hoá tin nhắn. Khi các bạn đang đọc được những dòng chữ mình gửi lên trên Teams thì điện thoại các bạn cũng đã thực hiện thuật toán này để giải mã được tin nhắn của mình.

Đây là một thuật toán mà cơ sở toán học dựa trên chuyên ngành rất thú vị của toán học: **Toán học rời rạc** và **Cấu trúc đại số** (Abstract Algebra).

2 Một mô hình đồ chơi của thuật toán Diffie-Hellman

Ở trên lớp học, mình được giới thiệu về một mô hình đơn giản hoá của thuật toán này, ở dưới cái tên "TOY7". Mô hình này tuy là "đồ chơi", nhưng có thể giúp chúng ta hiểu rõ hơn về cách hoạt động của thuật toán.

Definition 2.1. *Một mô hình đơn giản của thuật toán Diffie-Hellman, gọi là "TOY7", bao gồm các thành phần sau:*

- *Đoạn tin nhắn cần mã hoá.*
- *Một số nguyên tố p.*
- *Một số nguyên g.*
- *Hai người chơi, Long và Huy.*

Trước tiên mình sẽ giới thiệu về hoạt động của thuật toán. Sẽ có một vài bước có thể sẽ rất khó hiểu nếu như không có cơ sở toán học. Mình sẽ giới thiệu về những cơ sở toán học này ở phần đằng sau, để cho những anh chị em nào muốn phá mã ngay có thể nhảy vào luôn.

Tổng quan quá trình

Mô hình được gọi là TOY7 vì trên lớp, chúng mình lấy vị dụ với một đoạn tin nhắn có 7 ký tự.

M_1	M_2	M_3	M_4	M_5	M_6	M_7
S	E	T	A	I	N	T

Hình 1: Ví dụ về một đoạn tin nhắn có 7 ký tự cần mã hoá

Thông thường, dãy ký tự này trước tiên được sắp xếp lại theo một cách ngẫu nhiên (permutation), ví dụ:

M_1	M_2	M_3	M_4	M_5	M_6	M_7
E	N	T	A	I	S	T

Hình 2: Dãy ký tự sau khi sắp xếp lại

Mỗi ký tự đều có thể được biểu diễn bằng một số nguyên từ 0 đến 25. Ví dụ, ký tự "A" có thể được biểu diễn bằng số 0, "B" là số 1, "C" là số 2, v.v. Với cách biểu diễn này, dãy ký tự trên có thể được biểu diễn bằng dãy số sau:

M_1	M_2	M_3	M_4	M_5	M_6	M_7
4	13	19	0	8	18	19

Hình 3: Dãy số biểu diễn cho dãy ký tự trên

Bước tiếp theo là bước quan trọng nhất, đó chính là "encrypt" dãy số trên. Mã hoá được thực hiện bằng cách nhân mỗi số trong dãy số trên với một số *key* nào đó, và sau đó lấy đồng dư với 26. Ví dụ với số $key = 3$, dãy số trên sẽ được mã hoá như sau:

M_1	M_2	M_3	M_4	M_5	M_6	M_7
4	13	19	0	8	18	19
12	13	5	0	24	2	5

Hình 4: Dãy số sau khi được mã hoá

Dãy số sau khi được mã hoá có thể được biểu diễn bằng dãy ký tự sau:

M_1	M_2	M_3	M_4	M_5	M_6	M_7
M	N	F	A	Y	C	F

Hình 5: Dãy ký tự sau khi được mã hoá

và anh chị em thậm chí có thể dùng mã hoá này để gửi thư giấy cho nhau.

Làm thế nào để giải mã

Để giải mã, chúng ta cần biết *key* mà đã được sử dụng để mã hoá dãy số trên. Sau đó chúng ta sẽ nhân mỗi số trong dãy số đã được mã hoá với số nghịch đảo của *key*, và lấy đồng dư với 26. Rồi sau đó đảo ngược lại quá trình sắp xếp ban đầu, chúng ta sẽ thu được dãy ký tự ban đầu.

Vậy thì có 2 thông tin quan trọng mà Long và Huy cần phải trao đổi được cho nhau, đó chính là cách thức sắp xếp lại dãy ký tự ban đầu, và số *key* mà đã được sử dụng để mã hoá dãy số.

Trong bài viết này, chúng ta bỏ qua thông tin ban đầu, và coi như không có sắp xếp lại dãy ký tự. Chúng ta chỉ quan tâm đến việc trao đổi số *key* giữa Long và Huy, vì phần này bao gồm "toán thú vị hơn".

Bước tạo khoá

1. Long chọn một số nguyên a ngẫu nhiên từ 1 đến $p-1$.
2. Huy chọn một số nguyên b ngẫu nhiên từ 1 đến $p-1$.
3. Long tính $l = g^a \mod p$, và Huy tính $h = g^b \mod p$.
4. Long gửi l cho Huy, và Huy gửi h cho Long.
5. Long tính $h^a \mod p$, và Huy tính $l^b \mod p$.
6. Kết quả của cả hai phép tính trên bằng nhau và bằng g^{ab} , và được dùng làm khoá chung để mã hoá và giải mã tin nhắn.

Trong bước ở trên, những số màu đỏ là thông tin private, những số màu xanh là thông tin public. Các bạn có thể thấy, thông tin public rất nhiều.

Một ví dụ về những con số được public, với các số private a và b dài khoảng 15 chữ số:

- $p = 281474976710677$
- $g = 6$
- $l = 106080165434618$
- $h = 251434599145966$

Để phá được mã, ta bắt buộc phải có được thông tin private, đó chính là a hoặc b. Thậm chí nếu ta dùng h và l để cố tìm ra a và b, giả sử ta nhân h với l, ta sẽ thu được g^{ab}, và khi lấy đồng dư với p ta sẽ thu được một con số gần như không liên quan.

Thử thách vui: Anh em có thể viết code để tìm ra được a và b trong ví dụ trên hay không? Mình viết code C++ và chạy hết mất khoảng 13s để tìm ra.

Trong thực tế, số nguyên tố p được chọn là một số vô cùng lớn, thường là 2048 bit hoặc thậm chí 4096 bit.

3 Một vài nét về cơ sở toán học

Và bắt đầu phần thú vị. Trong thuật toán ở trên, anh chị em có thể, một cách vô cùng bản năng, hỏi những câu hỏi như sau:

- Tại sao phải là số nguyên tố?
- Số g có liên quan gì? Có thể chọn ngẫu nhiên hay không?
- Tại sao phải lấy đồng dư?
- Tại sao $h^a = l^b (\mod p)$?

Note: Tất nhiên là tôi không thể nào trình bày hết lý thuyết của cấu trúc đại số hay toán rời rạc mà có liên quan tới thuật toán ra đây được. Tôi chỉ có thể trình bày một vài định nghĩa và đưa ra một vài ý khái quát thôi. Nếu như ai có đọc được bài viết này mà tò mò thêm thì luôn luôn có thể nhắn tin trao đổi thêm.

Trả lời câu hỏi dễ trước:

$$h^a = g^{ab} = g^{ba} = l^b$$

Vậy thì tại sao lại phải chọn số nguyên tố?

Trước tiên, cần phải biết được chúng ta đang làm việc ở trên **tập số nào**. Trong trường hợp này, chúng ta đang làm việc trên tập số nguyên modulo p,

$$\mathbb{Z}_p = \{0, 1, 2, \ldots, p-1\}$$

với phép toán nhân modulo p.

Tập số này, cùng với phép toán nhân, tạo thành một **nhóm** (group). Nhóm này được gọi là **multiplicative group** modulo p, được ký hiệu là $\mathbb{Z}_p^*$.

Vậy thì một nhóm là gì?

Definition 3.1. *A group is a set G together with an operation $\cdot$ that satisfies the following axioms:*

- *Closure: For all $a, b \in G$, $a \cdot b \in G$.*
- *Associativity: For all $a, b, c \in G$, $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.*
- *Identity element: There exists an element $e \in G$ such that for all $a \in G$, $a \cdot e = e \cdot a = a$.*

- *Inverse element: For all $a \in G$, there exists an element $a^{-1} \in G$ such that $a \cdot a^{-1} = a^{-1} \cdot a = e$.*

Một bài tập nhỏ: Hãy chứng minh rằng $\mathbb{Z}_p^*$ là một nhóm.

Vậy thì tại sao lại là số nguyên tố? Trước tiên thì ta cần phải biết rằng nếu p là số nguyên tố, nhóm $\mathbb{Z}_p^*$ là một nhóm **cyclic** (cyclic group).

Theorem 3.1. *Every group of prime order is cyclic.*

Definition 3.2. *Let G be a group and $a \in G$. The cyclic group generated by a is the set*

$$\langle a \rangle = \{a^n : n \in \mathbb{Z}\}$$

*a is called the **generator** of $\langle a \rangle$.*

Và như vậy là chúng ta trả lời được hai câu hỏi

Tại sao phải là số nguyên tố?

Vì nhóm $\mathbb{Z}_p^*$ là cyclic. Có nghĩa là ta có thể tới được mọi phần tử trong nhóm này bằng cách lấy lũy thừa của phần tử generator.

Điều này là cần thiết bởi vì ta mong muốn chọn được a và b ngẫu nhiên, nếu không thì sẽ phải có một quy luật nào đó cho sự lựa chọn, và từ đó khiến cho thuật toán yếu đi nhiều.

Số g có liên quan gì? Có thể chọn ngẫu nhiên hay không?

Số g phải là một generator của nhóm $\mathbb{Z}_p^*$.

Để trả lời được câu hỏi tại sao thuật toán này lại sử dụng được, an toàn ở điểm nào, ta cần biết thêm một vài định nghĩa sau đây

Definition 3.3. *Let G be a group and $a \in G$. The order of a is the smallest positive integer n such that $a^n = e$. If no such n exists, then the order of a is infinite.*

Ở đây ta có thể mặc định rằng nhóm số mà ta đang hoạt động ở trên là một nhóm hữu hạn.

Tại sao thuật toán này lại an toàn?

Như vậy thì tổng hợp lại, để có thế phá được mã và có được chìa khoá để giải mã. Ta cần phải tìm được g^{ab}. Để làm được điều này, ta cần phải biết

được hoặc a hoặc b. Từ đó, ta có thể tính được g^{ab}, và đây chính là chìa khoá để đảo ngược thuật toán.

Câu trả lời ngắn gọn: Từ $l = g^a$ hoặc $h = g^b$, với một số g đủ lớn thì việc tìm được a hoặc b là một điều rất khó.

Tại sao lại khó?

Ta đang làm việc ở trên một nhóm số *multiplicative*, từ một *generator* g, để từ một *element*, để tới được một *element* khác, ta cần phải thực hiện phép nhân. (Các thành phần của nhón $\langle g \rangle$ là $\{g^1, g^2, g^3 \dots\}$)

Có kẽ anh chị em đều biết rằng, phép toán nghịch đảo của phép tính luỹ thừa là phép tính logarithm. Đối với tập số rời rạc mà ta đang làm việc ở trên thì ký hiệu bằng $dlog_g$

Trước tiên, có một điểm cần chú ý là 2 số a và b, đối với một người ngoài cuộc (i.e. ngoài Long và Huy ở ví dụ trên) thì ta phải coi chúng là các số nguyên ngẫu nhiên. Thậm chí trong nhiều trường hợp, ta còn không thể biết được cận trên của các số này là bao nhiêu.

Như vậy thì từ một số (rất lớn) ở trong một nhóm cyclic, ta cần tìm một số nguyên bình thường trong tập số $\mathbb{Z}$.

Những anh chị em có trực giác mạnh với toán học hơn sẽ thấy được rằng ta đang từ một nhóm mà tìm tới một số ở trong nhóm khác ($\langle g \rangle \to \mathbb{Z}$).

Về mặt toán học thì điều này có gây ra cản trở gì không? Câu trả lời là không vì chúng ta có khái niệm *isomorphism*

Definition 3.4. *An isomorphism between two groups G and H is a bijective homomorphism $\phi : G \to H$. If such an isomorphism exists, we say that G and H are isomorphic.*

Definition 3.5. *A homomorphism between two groups G and H is a function $\phi : G \to H$ such that*

$$\phi(a \cdot b) = \phi(a) \cdot \phi(b)$$

for all $a, b \in G$.

Nhắc lại về *song ánh* (bijection)

Definition 3.6. *A bijection is a function that is both injective and surjective. A function is injective if it maps distinct elements of the domain to distinct elements of the codomain. A function is surjective if every element of the codomain is the image of at least one element of*

the domain.

Một cách nôm na thì 2 nhóm isomorphic là "tương đương", theo kiểu "bình mới rươu cũ": Bản chất bên trong là như nhau, nhưng chỉ khác nhau về mặt ký hiệu

Kết quả sau đây tôi chỉ nêu ra:

Nhóm $(\mathbb{Z}_p, *)$ là một nhóm cyclic, và mọi nhóm cyclic đều là isomorphic với $(\mathbb{Z}_n, +)$

Bằng việc tìm được một isomorphism giữa 2 nhóm này, ta có thể chuyển từ một nhóm này sang nhóm khác mà không làm mất đi tính chất của nhóm.

Ví dụ:

Nhóm $(\mathbb{Z}_5, *) = \{1, 2, 3, 4\}$ isomorphic với nhóm $(\mathbb{Z}_4, +) = \{0, 1, 2, 3\}$

Ta có thể tìm được song ánh từ nhóm $(\mathbb{Z}_5, *)$ sang nhóm $(\mathbb{Z}_4, +)$ như sau:

- $2^1 = 2 \mod 5 \to 2$
- $2^2 = 4 \mod 5 \to 1$
- $2^3 = 3 \mod 5 \to 3$
- $2^4 = 1 \mod 5 \to 0$
- $2^5 = 2^1 = 2 \mod 5 \to 2$

Và ngược lại, từ nhóm $(\mathbb{Z}_4, +)$, ví dụ khi ta có $2 + 1 = 3$, ở trong nhóm $(\mathbb{Z}_5, *)$ ta sẽ có $2^2 * 2^1 = 2^3 = 3 \mod 5$

Quay lại với mô hình đồ chơi ở trên, câu hỏi trở thành

Tìm số a sao cho

$$6^a = 106080165434618 \mod 281474976710677$$

Các bạn có thể thấy được với mô hình đồ chơi như vậy thôi mà việc tính toán cũng khá là kinh khủng.

Cho một số y ở trong nhóm $\mathbb{Z}_4$, làm thế nào để ta có thể tìm lại được số x ở trong nhóm $\mathbb{Z}_5$ mà đã sinh ra số y này thông qua phép luỹ thừa?

Nếu như ta có sẵn bảng tính như trong ví dụ ở trên thì điều này là rất dễ dàng, ta có thể tìm ngay ra được, với thời gian tìm là $O(1)$. Tuy nhiên thì với một con số vô cùng lớn, việc tính toán hết ra và lưu và trong bộ nhớ là từ không khả thi cho tới không thể.

Vậy thì một cách làm khác là phải biết được phép nghich đảo của phép luỹ thừa, tức là tính được *dlog* từ nhóm $(\mathbb{Z}_5, *)$ sang nhóm $(\mathbb{Z}_4, +)$.

Từ phép nhân ở trong nhóm $(\mathbb{Z}_p, *)$, ta có thể chuyển sang phép cộng ở trong nhóm $(\mathbb{Z}_n, +)$. Như vậy thì việc tìm ra isomorphism giữa 2 nhóm này còn có thể giúp ta đơn giản hoá phép tính đi rất nhiều.

Nhưng câu hỏi lại quay lại: Làm thế nào để tìm được isomorphism đó?