Zalo | Tài liệu kỹ thuật

Giao thức mã hóa đầu cuối Zalo

Tháng 4, 2022

Mục lục

Giới thiệu

Tài liệu này cung cấp góc nhìn tổng quan về mặt kỹ thuật của giao thức mã hóa đầu cuối (End-to-end encryption) cho các trò chuyện Zalo.

Người dùng Zalo đã có thể bảo mật trò chuyện 2 người và trò chuyện nhóm bằng giao thức mã hóa đầu cuối. Giao thức này hỗ trợ mã hóa tin nhắn bằng mã khóa. Các mã khóa này chỉ được lưu trữ trên máy của người dùng và không được lưu trữ tại bất cứ nơi nào khác (kể cả trên máy chủ Zalo). Ngoài người dùng trong trò chuyện, không ai có thể giải mã được tin nhắn mã hóa đầu cuối.

Các tin nhắn được mã hóa bao gồm: tin nhắn văn bản, tin nhắn thoại, ảnh, video, file, sticker, GIF, MP3, hình vẽ tay, emoji, vị trí.

Zalo xây dựng giao thức mã hóa đầu cuối dựa trên các giải thuật Double Ratchet, Diffie-Hellman, Elliptic curve DH, v.v. Đây là các giải thuật mã nguồn mở, được nhiều cá nhân, tổ chức uy tín kiểm định.

Phần tiếp theo của tài liệu đưa ra định nghĩa các thuật ngữ và ký hiệu được dùng, phục vụ cho phần giới thiệu các giải thuật được sử dụng. Cuối cùng, tài liệu này sẽ mô tả giao thức mã hóa đầu cuối của Zalo.

Giao thức mã hóa đầu cuối

Thuật ngữ & ký hiệu

I	**Identity Key** Cặp khoá 256 bits được tạo ra theo giải thuật Curve25519, được sử dụng để định danh cho một tài khoản Zalo, sinh ra bởi thiết bị chính (thiết bị di động) của người dùng.
S	**Signed PreKeys** Cặp khoá được tạo ra theo giải thuật Curve25519, đã được xác thực bởi Identity Key, khoá này được thay thế định kỳ (1 ngày, tuần, tháng, v.v.).
O	**One-Time Prekeys** Các cặp khoá được tạo ra theo giải thuật Curve25519, là loại khoá chỉ được sử dụng 1 lần, liên tục được tạo ra khi có yêu cầu.
E	Ephemeral Key cho việc khởi tạo phiên.
D	Ephemeral Key cho việc tạo ra Chain Key.
RK	**Root Key** Khoá được sử dụng để tạo ra Chain Key.
CK	**Chain Key** Khoá được sử dụng để tạo ra Message Key.
MK	**Message Key** Khóa dùng 1 lần cho việc mã hóa thông tin gửi đi.
SK	**Secret Key**
	ECDH(X,Y) Hàm này tạo ra 1 khoá chia sẻ bí mật nếu X và Y là khoá công khai ECDH của 2 người dùng.
	HKDF(X) HMAC-based Key Derivation Function.
	HMAC-SHA256(K,P) Tính toán HMAC sử dụng SHA256 và khoá K của thông tin.

Các giải thuật & giao thức mã hóa đầu cuối

Giao thức đồng thuận khóa X3DH

X3DH là một giao thức đồng thuận dùng để thiết lập một mã khóa chung giữa 2 bên khi muốn xác thực lẫn nhau dựa trên các cặp khoá công khai (Public Keys). Giao thức này cung cấp hai tính năng:

- **Forward secrecy**
  Tính năng đảm bảo khi một mã khoá bị lộ thì cũng không ảnh hưởng tới các tin nhắn trước.

- **Phủ định mã hoá (cryptographic deniability)**
  Chỉ với các tin nhắn đã được mã hóa, bên thứ ba (third party) không thể chứng minh được là cả hai bên tham gia giao thức đã thực sự trao đổi thông tin với nhau. Nói cách khác, mỗi bên tham gia giao thức đều có thể phủ nhận dữ liệu đã mã hóa trên là của họ, hoặc phủ nhận họ có khả năng giải mã nó, mà không có bên nào khác có thể chứng minh được (deniable).

X3DH là là giao thức đồng thuận khoá bất đồng bộ. Người gửi tạo tin nhắn khởi tạo session mã hoá và dùng session đó để mã hoá tin nhắn khởi tạo cho người nhận, ngay cả khi người nhận không trực tuyến. Người nhận sau đó có thể xử lý tin nhắn khởi tạo này và dùng nó để tính toán ra session tương ứng dùng cho giải mã tin nhắn.

Để thiết lập một session mới:

Bước 1: Người thiết lập session (A) lấy bộ mã khóa gồm I_B, S_B, và O_B nếu có từ máy chủ.

Bước 2 A tiếp tục tạo ra 1 cặp key E nhờ vào giải thuật Curve25519 và tính ra R từ công thức:

$$
\begin{aligned}
DH1 &= DH(I_A, S_B) \\
DH2 &= DH(E_A, I_B) \\
DH3 &= DH(E_A, S_B) \\
Master\ SK &= KDF(DH1 \,||\, DH2 \,||\, DH3)
\end{aligned}
$$

Nếu có O_B thì sẽ tính thêm DH4 bằng công thức:

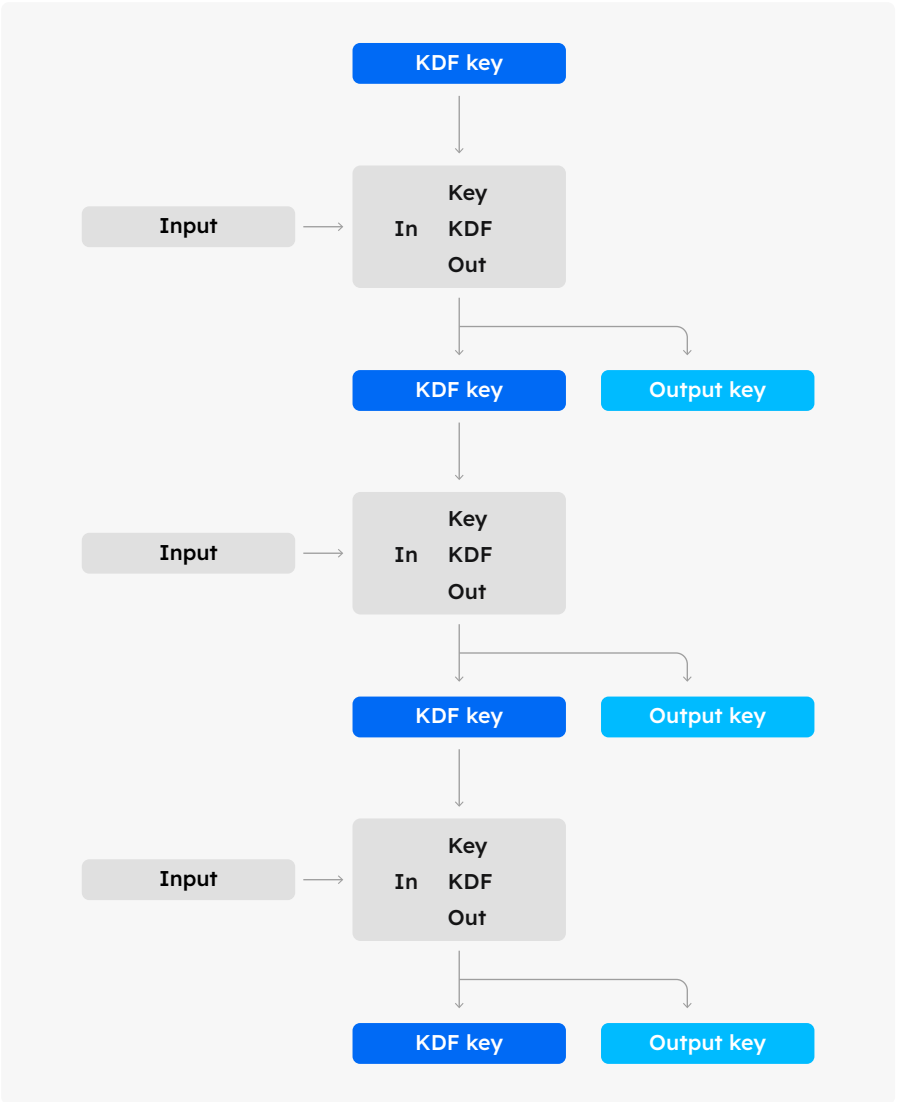
$$
\begin{aligned}
DH4 &= DH(E_A, O_B) \\
Master\ SK &= KDF(DH1 \,||\, DH2 \,||\, DH3 \,||\, DH4)
\end{aligned}
$$

Các công thức trên miêu tả quá trình tính toán khoá sử dụng giao thức DH. Chú ý DH1 và DH2 dùng để tạo ra xác thực chung, trong khi đó DH3, DH4 dùng để tạo ra tính “forward secrecy” của giao thức.

Sau khi có được SK, A có thể dùng trực tiếp SK để mã hóa tin nhắn gửi cho B hoặc dùng để tăng cường bảo mật thông qua các giải thuật như DH Ratchet và Double Ratchet.

Giải thuật Double Ratchet

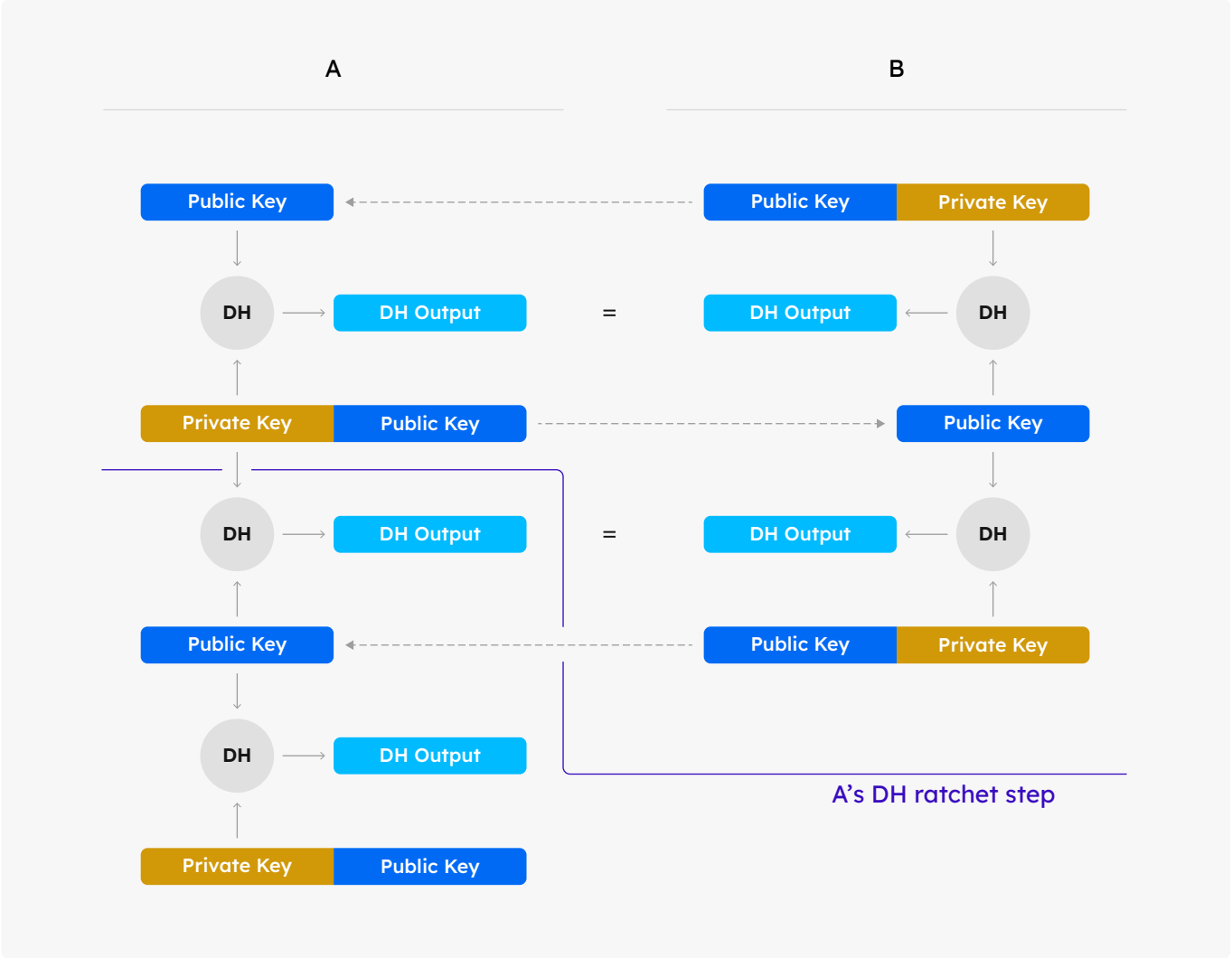
KDF (Key Derivation Function)
Là một hàm mã hoá có đầu vào là một khoá bảo mật và ngẫu nhiên (secure and random) và dữ liệu để tạo ra 1 chuỗi dữ liệu mới có thể dùng làm mã khóa cho các giải thuật mã hoá khác. Tuy nhiên nếu chuỗi dữ liệu này không ngẫu nhiên thì KDF phải cung cấp 1 hàm băm (hash) bảo mật mã khoá và dữ liệu đầu vào (HKDF) có thể đáp ứng đầy đủ tiêu chuẩn của KDF.



Hình 1
Giải thuật KDF

DH Ratchet

Trong quá trình trao đổi tin nhắn giữa A và B, nếu A vô tình để lộ SK từ việc trao đổi mã khóa (như mã khóa được tạo từ giải thuật X3DH), tất cả các tin nhắn cũ đều có thể bị bên thứ ba đọc được. Để giải quyết vấn đề này, DH Ratchet được sử dụng như hình 2. SK sẽ được thay đổi trong 1 lượt gửi nhận tin nhắn.

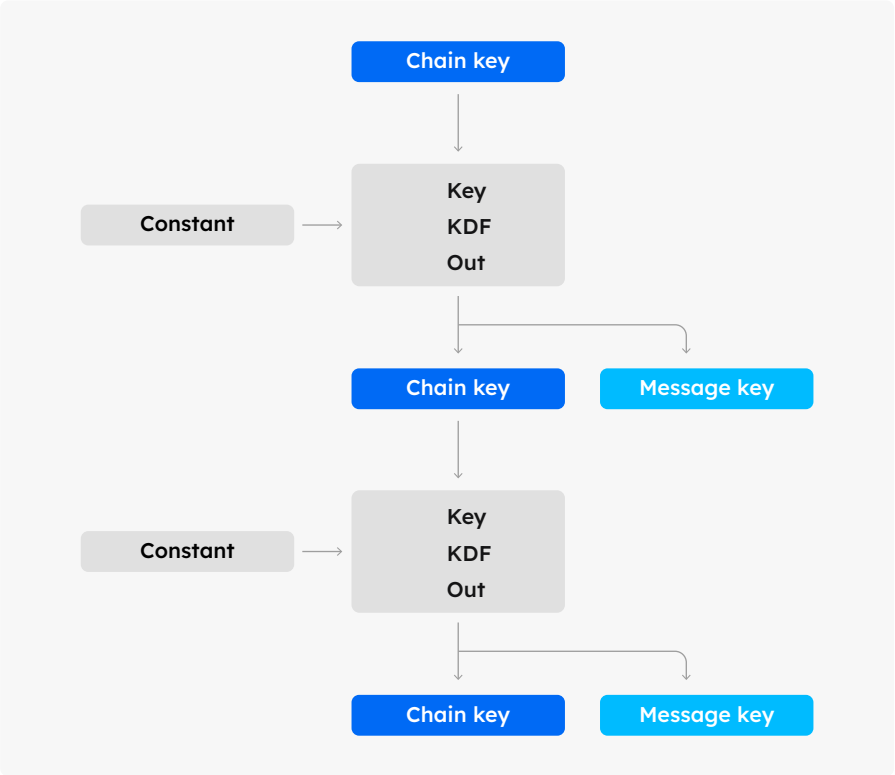


Hình 2
Giải thuật DH Ratchet

Symmetric Key Ratchet

Mỗi tin nhắn gửi nhận được mã hóa với 1 MK duy nhất. MK này là đầu ra từ chuỗi gửi nhận KDF, thường được gọi là Chain Keys. Đầu vào của KDF thường là hằng số (có thể là 0), do đó không đảm bảo được tính khôi phục khi bị thâm nhập, mà chỉ đảm bảo MK dùng để mã hóa hoặc giải mã tin nhắn được xóa ngay lúc đó.

Vì MK không được dùng để sinh ra mã khóa kế tiếp, có thể lưu trữ MK mà không ảnh hưởng tới bảo mật của bất kỳ tin nhắn nào khác, tránh mất và tin nhắn tới không đúng thứ tự.



Hình 3
Giải thuật Symmetric Key Ratchet

Double Ratchet

Việc kết hợp 2 giải thuật DH Ratchet và Symmetric Key Ratchet sẽ tạo nên giải thuật Double Ratchet như sau:

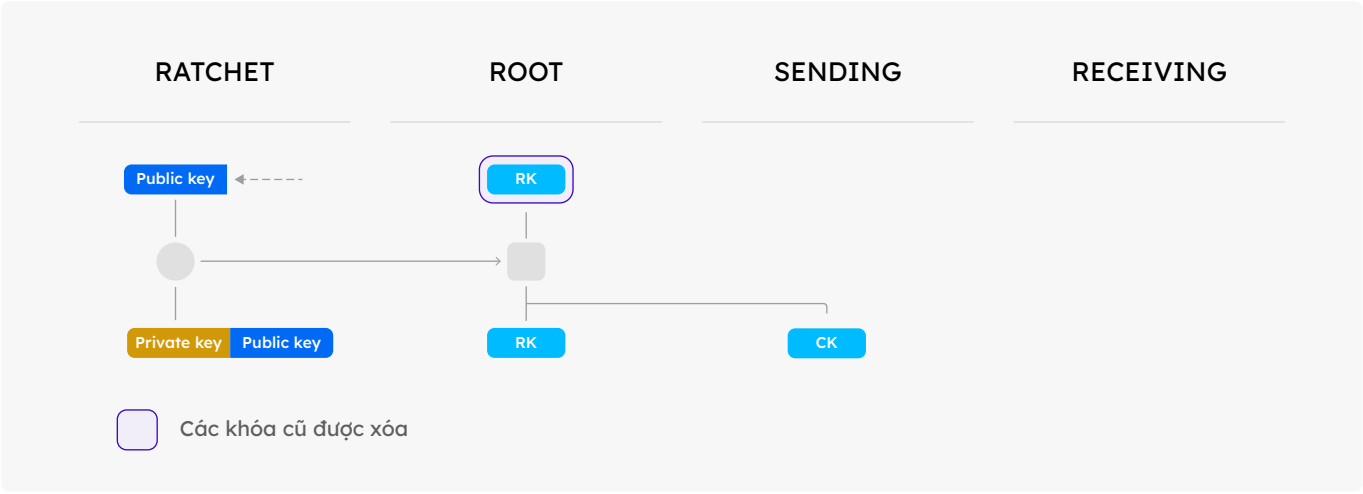
- Khi tin nhắn được gửi và nhận, Symmetric Key Ratchet được áp dụng để tính toán ra MK để mã hóa hoặc giải mã.

- Khi nhận Ratchet Public Key mới, thực hiện DH Ratchet với Symmetric Key Ratchet trước đó để tạo ra một Chain Key hoàn toàn mới.

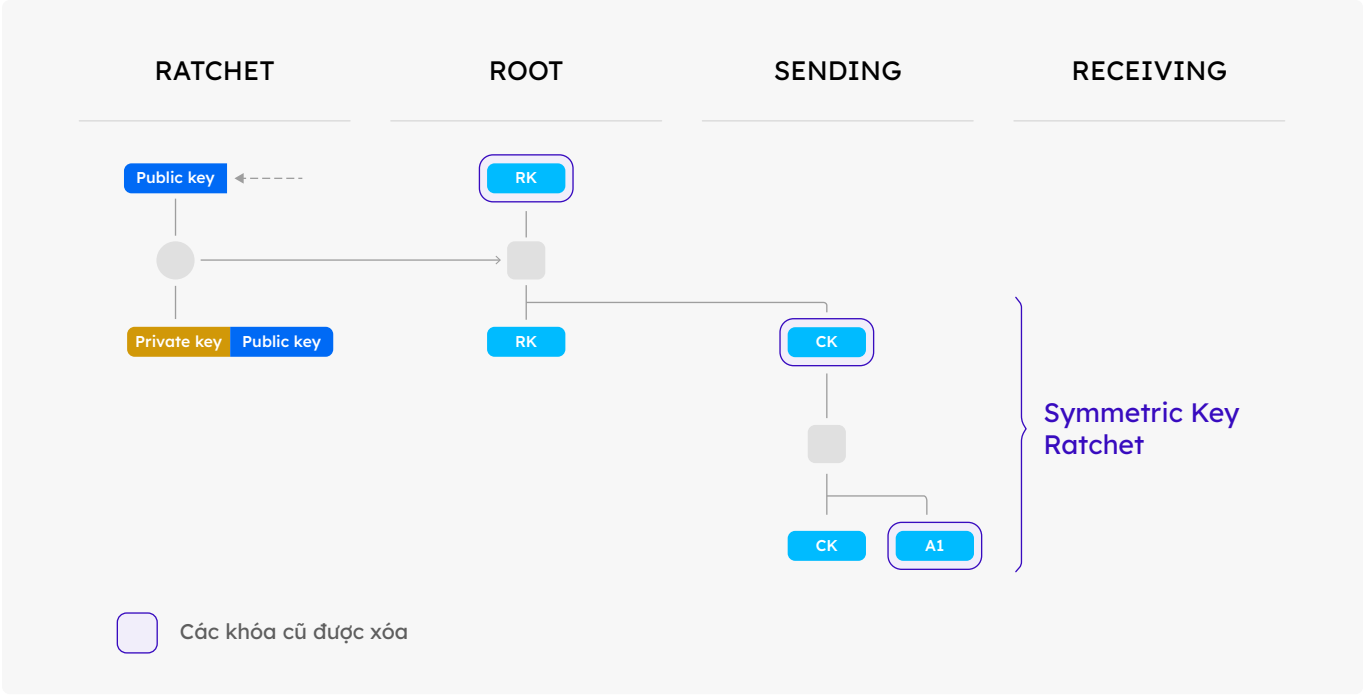
Các giải thuật Ratcheting như Double Ratchet cho phép 2 bên cập nhật Session Key khi trao đổi tin nhắn với nhau, đảm bảo được tính chất “forward secrecy”.

Ví dụ cụ thể:

Khi bắt đầu trao đổi tin nhắn, A tạo ra 1 cặp khoá mới và lấy Public Key của B. Sử dụng giải thuật DH Ratchet để tạo đầu vào cho KDF, tính toán ra Root Key (RK) và Sending Chain Key (CK).



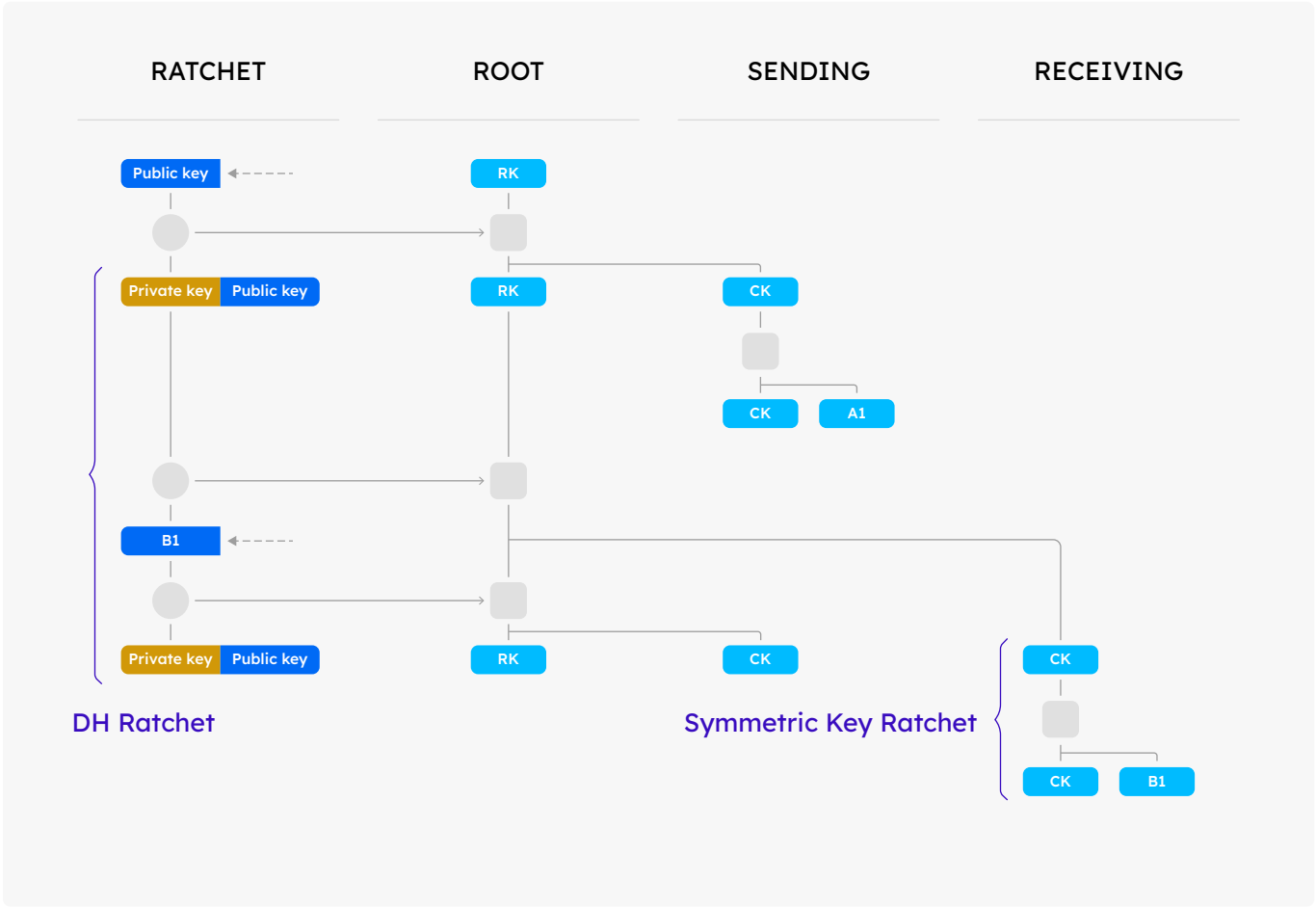
Khi A gửi tin nhắn đầu tiên A1, A sử dụng Symmetric Key Ratchet với Sending Chain Key (chuỗi key dùng cho việc gửi tin nhắn mã hoá) và đồng thời tạo ra một MK mới. Chain Key này sẽ được lưu lại. MK và Chain Key cũ sẽ bị xóa.



Khi A nhận tin nhắn trả lời B1 từ B, tin nhắn này đính kèm Public Key mới từ B:

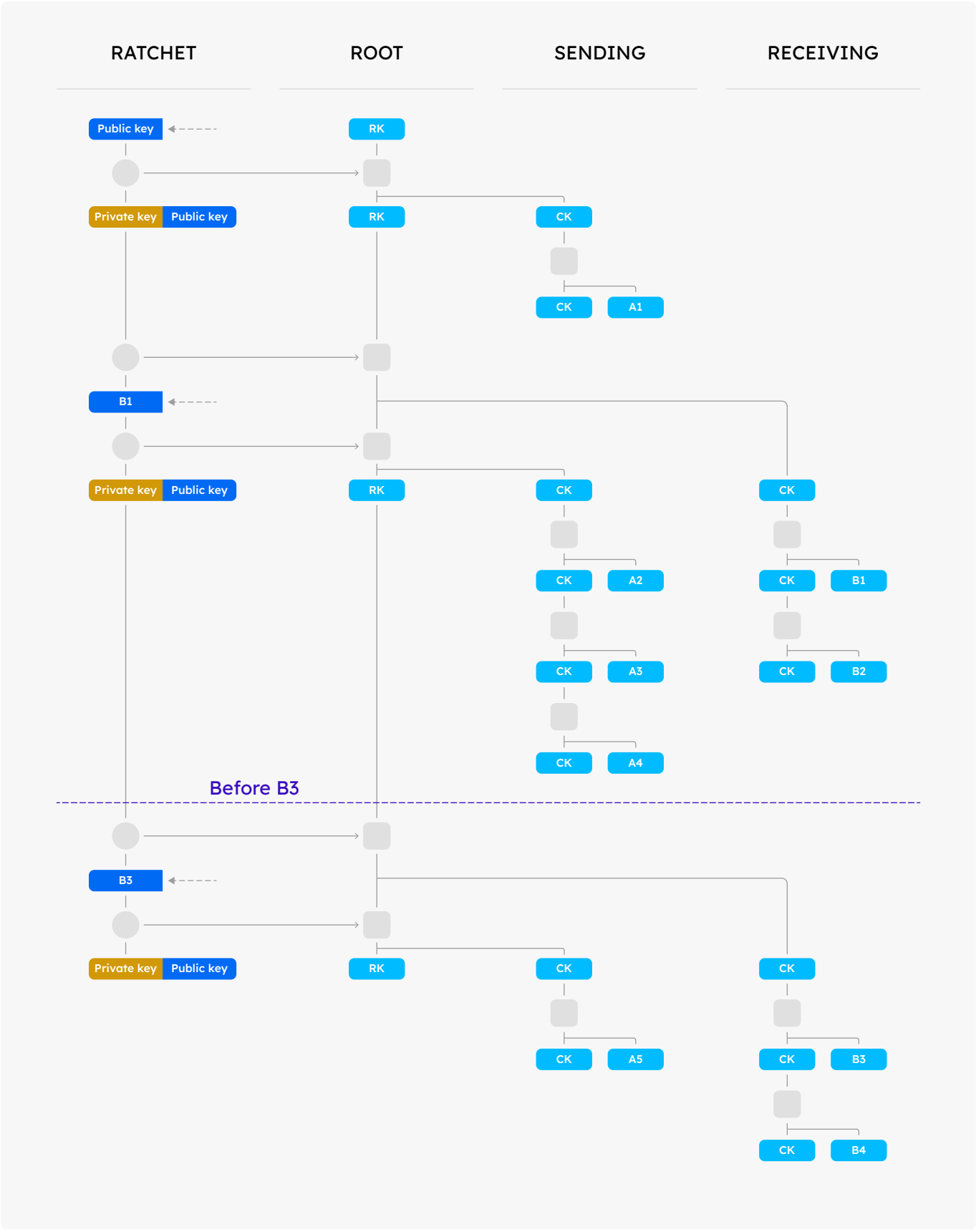
- A áp dụng DH Ratchet với Private Key hiện tại để tạo ra Receiving Chain Key mới, đồng thời sử dụng KDF tạo ra MK cho việc giải mã tin nhắn B1.

- A tạo ra một cặp khoá mới. Sau đó áp dụng DH Ratchet với Private Key mới này và Public Key nhận được từ B để tạo ra Sending Chain Key mới.



Giả sử A gửi tin nhắn A2 và nhận tiếp tin nhắn B2 với Public Key cũ giống B1. Sau đó A gửi tiếp tin nhắn A3 và A4, Sending Chain Key của A sẽ áp dụng 3 lần Ratchet để tạo ra MK tương ứng cho mỗi tin nhắn, còn Receiving Chain Key sẽ áp dụng 1 lần Ratchet để tạo ra MK giải mã B2.

Sau đó A nhận được tin nhắn B3, B4 với Ratchet Key mới (Public Key mới). A tiếp tục gửi tin nhắn A5. Toàn bộ quá trình gửi nhận tin nhắn được mô tả như hình 4.



Hình 4
Giải thuật Double Ratchet

Giải thuật Sesame

Kết hợp giải thuật X3DH và Double Ratchet có thể gặp phải những vấn đề sau:

- A và B tham gia trò chuyện mã hóa đầu cuối sử dụng nhiều thiết bị. Vì vậy, việc mã hoá một tin nhắn từ A tới B có thể phải tạo ra nhiều session: từ thiết bị của A gửi tới tất cả các thiết bị của B, đồng thời tới các thiết bị khác của A (mỗi thiết bị nhận một bản sao của tin nhắn này).

- A và B có thể thêm và bỏ thiết bị, do đó phải thêm hoặc xóa session để xử lý các thay đổi này.

- A và B có thể cùng lúc khởi tạo một session tới lẫn nhau để 2 session mới được tạo ra. Để Double Ratchet đạt được hiệu quả tối ưu nhất, A và B phải gửi và nhận tin nhắn sử dụng chung một session mà cả hai đồng thuận sử dụng.

- A có thể chọn quyền xoá các trạng thái (state) của session của hoặc khôi phục từ từ một bản lưu trữ (backup). Vì vậy, A và B có nhiều session cũ (orphaned session) không còn được sử dụng bởi người còn lại.

Sesame là giải thuật quản lý session mã hoá tin nhắn bất đồng bộ và cho nhiều thiết bị. Sesame được thiết kế để quản lý session của giải thuật Double Ratchet được tạo ra với giải thuật đồng thuận X3DH.

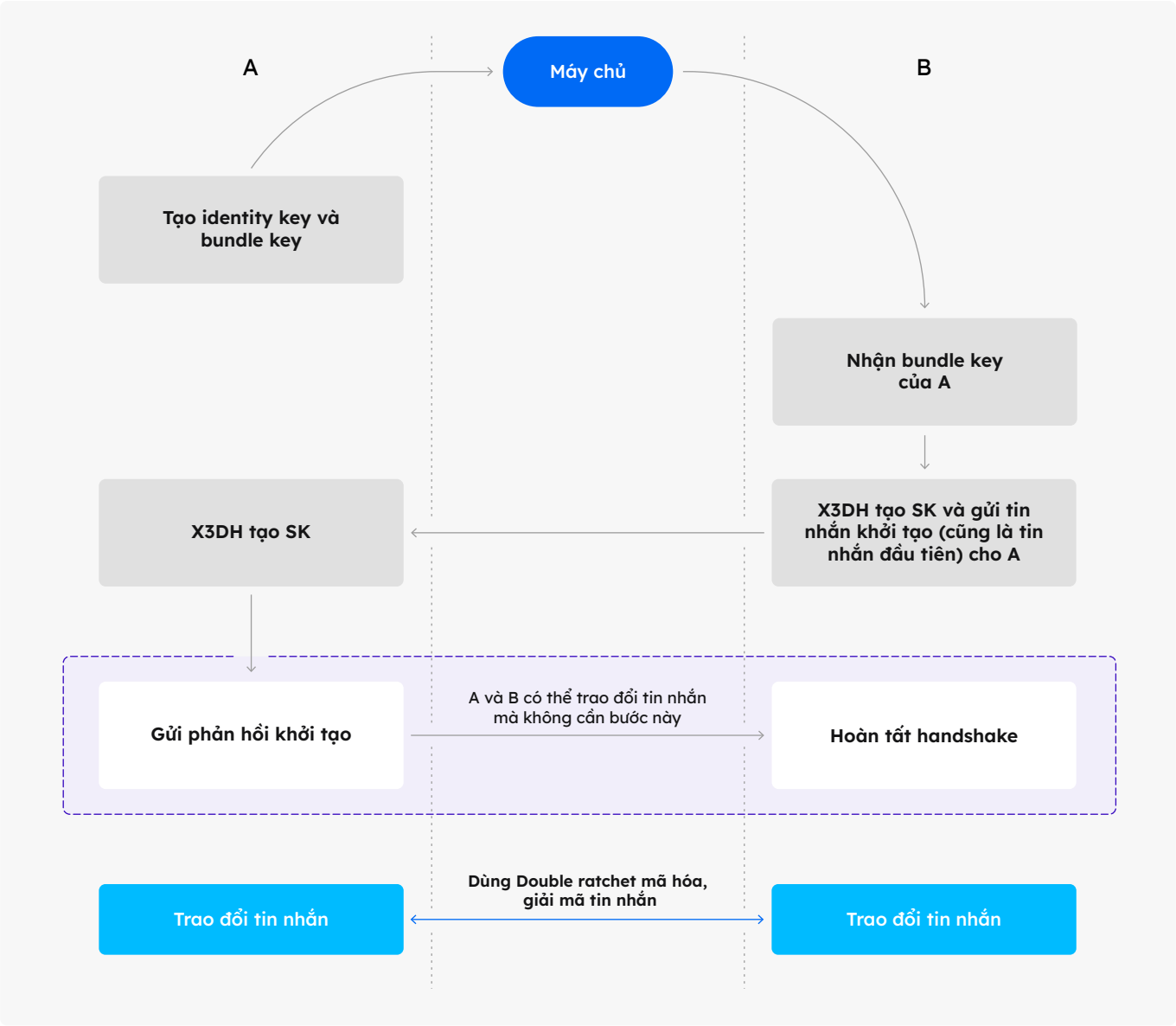
Lúc cài đặt, các thiết bị sẽ gửi O và S lên máy chủ cùng với I của chúng. Để khởi tạo một session, thiết bị gửi sẽ lấy bộ mã khóa này từ máy chủ, sau đó sử dụng giải thuật X3DH để tạo ra một SK dùng cho Double Ratchet session và tin nhắn khởi tạo X3DH. Các dữ liệu này được đính kèm trên mỗi tin nhắn để người nhận có thể dùng để tạo ra một session chung bằng Double Ratchet. Khi nhận được một phản hồi từ các tin nhắn khởi tạo, người gửi sẽ bỏ các dữ liệu liên quan tới X3DH cho các tin nhắn mã hoá sau này và chỉ sử dụng Double Ratchet.

Sesame quản lý việc tạo, xóa, và sử dụng session để đáp ứng các yêu cầu trên. Theo giải thuật này, thiết bị sẽ giữ số “active session” tương ứng với mỗi thiết bị khác mà nó đang giao tiếp. Các “active session” này được dùng để gửi tin nhắn tới các thiết bị khác. Khi nhận một tin nhắn được mã hoá bởi một “inactive session” thì session này sẽ trở thành một “active session” mới.

Mã hóa đầu cuối trong Zalo

Tổng quan

Mã hóa đầu cuối trong Zalo sử dụng giải thuật X3DH cho việc trao đổi key tạo ra SK và dùng cho giải thuật Double Ratchet để mã hoá tin nhắn. Tuy X3DH hỗ trợ việc trao đổi tin nhắn ngoại tuyến mà không cần có session từ trước, Zalo vẫn có thêm bước khởi tạo, phản hồi để dùng cho các tính năng khác.



Hình 5
Quá trình thiết lập mã hóa đầu cuối

Tại thời điểm cài đặt hoặc cập nhật, ứng dụng Zalo của người dùng tạo và gửi 1 bộ khoá công khai: I, S, O tới máy chủ. Máy chủ Zalo sẽ lưu trữ bộ khoá này với định danh của người dùng.

Khi một người dùng B muốn chat với người dùng A, quá trình thiết lập được diễn ra như sau:

Bước 1: B yêu cầu máy chủ trả về bộ khoá I, S, O của người dùng A.

Bước 2: Máy chủ trả về bộ khoá I_A, S_A, O_A của A, và xoá O_A đã được trả về cho B.

Bước 3: B tạo ra 1 khóa E(E_B).

Bước 4: B tính toán Master Key theo công thức như hình 5.

Bước 5: B gửi I_B, E_B của mình sang cho A, A dựa vào 2 khoá này và tính ra Master Key giống như B. A gửi lại E_A cho B.

Bước 6: A và B tạo ra Shared-Key chung (Root Key).

Bước 7: A và B tiến hành ECDH(E_A,E_B) tạo ra 1 khoá SK và sử dụng khoá SK và Root Key ở bước 6 để tạo ra khoá CK, RK.

Bước 8: A, B tạo ra MK dùng cho việc mã hoá tin nhắn.

Bước 9: B và A có thể bắt đầu gửi tin nhắn cho nhau.

Mã hóa đầu cuối trò chuyện 1-1

Mỗi khi nhận tin nhắn mới, 2 người dùng sẽ kiểm tra E đã tồn tại chưa. Nếu đã tồn tại, sẽ lấy R tương ứng E kèm theo tin nhắn đó tính MK dựa theo công thức bên dưới để cập nhật Chain Key.

$$\begin{aligned} \mathbf{MsgKey} &= \mathbf{HKDF(ChainKey, 0x01)} \text{ với hàm HMAC là HMAC-Sha256} \\ \mathbf{ChainKey_{NEW}} &= \mathbf{HMAC\text{-}Sha256(ChainKey_{PRE}, 0x02)} \end{aligned}$$

Chi tiết quá trình tính toán xem lại hình 3 (Symmetric Key Ratchet).

Khi nhận tin nhắn mới mà E chưa tồn tại thì sẽ thực hiện việc tính toán lại R và Chain Key mới dựa vào giải thuật DH Ratchet. Kèm theo mỗi tin nhắn sẽ có một bộ đếm cho E tương ứng dùng để tính toán key cho các trường hợp mất và tin nhắn đến không đúng thứ tự.

$$\begin{aligned} \mathbf{SK} &= \mathbf{ECDH(E_R, E_S)} \\ \mathbf{ChainKey,\ RootKey_{NEW}} &= \mathbf{HKDF(RootKey_{PRE}, SK)} \end{aligned}$$

Mã hóa đầu cuối chia sẻ file, ảnh, video, v.v.

Việc chia sẻ file được thực hiện theo các bước dưới đây:

Bước 1: Máy của người dùng tự tạo ra 1 khoá ngẫu nhiên dài 256 bits cho việc mã hoá.

Bước 2: Mã hoá file với khoá ngẫu nhiên bằng giải thuật AES256-GCM.

Bước 3: Tải file lên máy chủ Zalo.

Bước 4: Gửi link đính kèm với mã khoá về trò chuyện 2 người đã được bảo mật.

Bước 5: Máy của người dùng nhận tiến hành giải mã bằng khoá nhận được.

Mã hóa đầu cuối trò chuyện nhóm

Sau khi một trò chuyện nhóm được tạo ra, các thành viên trong nhóm sẽ tiến hành các bước dưới đây:

Bước 1: Các thành viên tự tạo ra 1 Chain Key 32 bytes ngẫu nhiên.

Bước 2: Gửi Chain Key này cho các thành viên còn lại trong nhóm thông qua đường trò chuyện 1-1 bảo mật đã được thiết lập từ trước (nếu chưa có thì máy của người dùng sẽ chủ động thiết lập từ đầu).

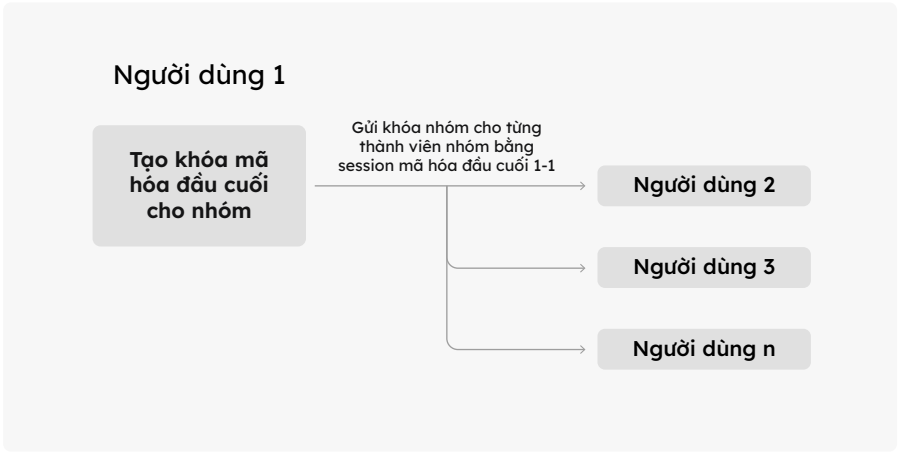
Bước 3: Tạo ra MK.

Bước 4: Gửi tin nhắn đã được mã hoá lên máy chủ.

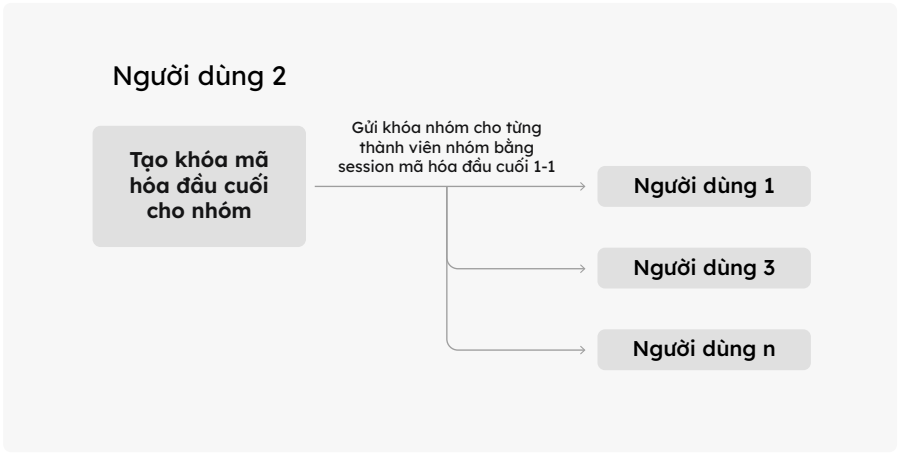
Bước 5: Máy chủ tiến hành gửi tin nhắn cho các thành viên trong nhóm.

Bước 6: Các thành viên dựa vào Chain Key đã nhận được để tạo ra khoá giải mã cho tin nhắn nhận được.

Khi có 1 thành viên rời khỏi trò chuyện nhó hoặc mới tham gia trò chuyện nhóm, tất cả các thành viên sẽ thực hiện lại Bước 1 và Bước 2.

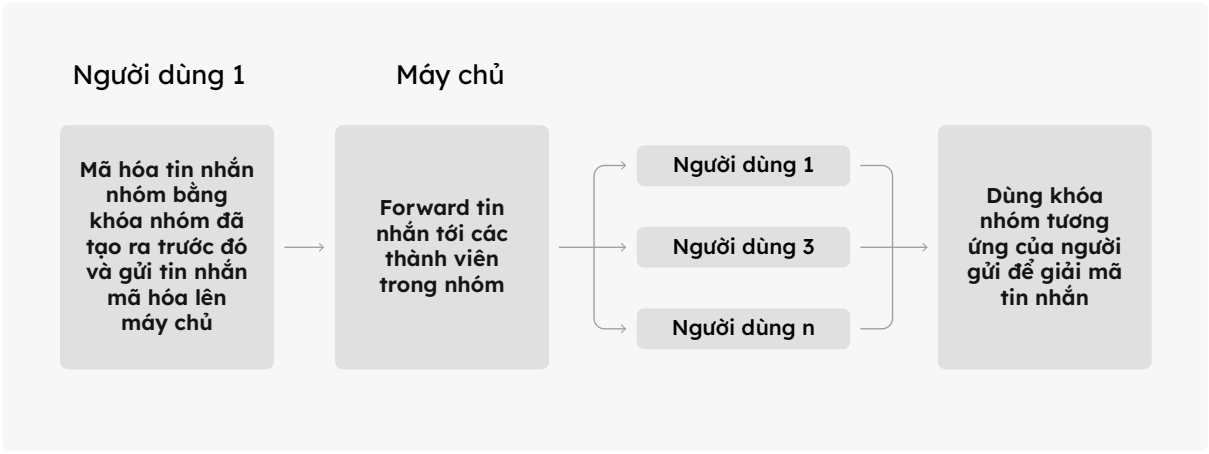


A gửi khóa nhóm cho các thành viên còn lại



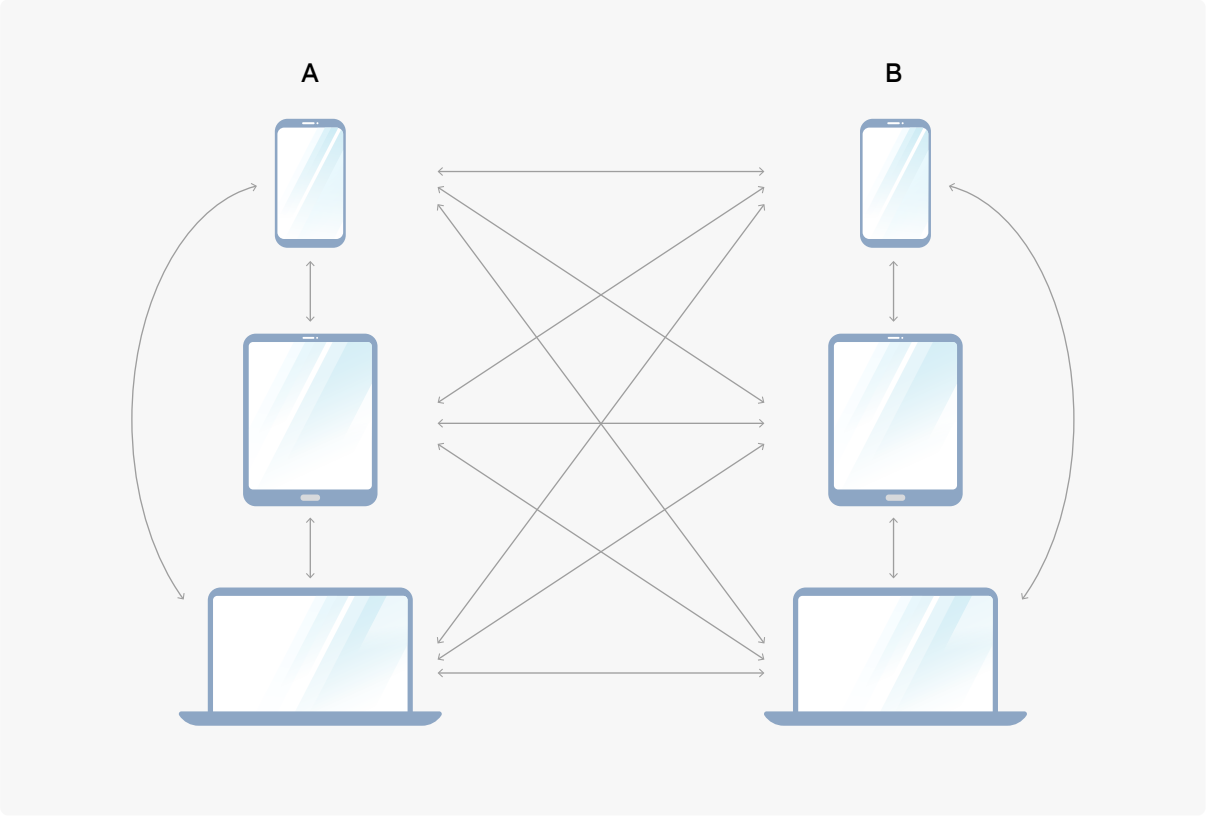
B gửi khóa nhóm cho các thành viên còn lại

Khi A gửi tin nhắn mã hóa đầu cuối tới một nhóm.



Mã hóa đầu cuối nhiều thiết bị

Mã hóa đầu cuối trên Zalo hỗ trợ gửi nhận trên nhiều thiết bị khác nhau bằng cách sử dụng giải thuật Sesame cho việc quản lý các session, mã hoá và giải mã tin nhắn. Khi một thiết bị của người dùng A muốn gửi một tin nhắn mã hoá đầu cuối tới B, A sẽ phải thiết lập session tới từng thiết bị của B và chính các thiết bị của mình. Sau đó, A dùng các session tương ứng mã hoá tin nhắn đó gửi tới từng thiết bị của B và các thiết bị của chính mình để đồng bộ tin nhắn.



Hình 6
Mã hóa đầu cuối nhiều thiết bị

Kết luận

Mã hóa đầu cuối là giao thức bảo mật tối ưu cho trò chuyện trên Zalo, được xây dựng dựa trên các giải thuật đã được công nhận và kiểm định bởi các nguồn uy tín. Bằng việc dùng các cặp mã khóa được lưu trữ độc lập trên máy của người dùng và không phụ thuộc vào máy chủ, giao thức này đảm bảo không bên thứ ba nào (kể cả Zalo) có thể giải mã và đọc được nội dung gốc của tin nhắn.

Để nâng cấp mã hóa đầu cuối cho các trò chuyện Zalo, hãy làm theo hướng dẫn này.

Tài liệu tham khảo

1. X3DH Key Agreement Protocol Xem thêm

2. Signal Protocol Xem thêm

3. Double Ratchet Xem thêm

4. Sesame Protocol Xem thêm