msgX: Rumo a Mensagens Pós-Quânticas Soberanas para Implantações Governamentais do Protocolo Matrix

Vinícius Lagrota[1], Henrique Viana[2], Kauã Maia[2], Marcos Ortiz[2], Windson Viana[2], Gilvan Maia[2], Paulo Rego[2], Rodrigo Pacheco[1], Rossana Andrade[2], José Antônio Macêdo[2]

[1] Centro de Pesquisa e Desenvolvimento para a Segurança das Comunicações (CEPESC)
Brasília, DF – Brasil.

[2]Universidade Federal do Ceará (UFC) – Fortaleza, CE – Brazil

Abstract. *This paper presents msgX, a secure messaging protocol for replacing Olm-based session distribution in governmental Matrix rooms, combining standardized and government-developed cryptography to support interoperability and digital sovereignty. To mitigate quantum threats, msgX integrates Post-Quantum Extended Diffie–Hellman (PQXDH), a modified Double Ratchet, and msgGX, a Megolm adaptation for scalable group encryption. In 14,040 Android runs across two devices, 30 each, and 234 parametrized configurations, a branch-level cost decomposition against Olm was performed, isolating the msgX-Γ and msgX-Σ profiles. The results show that each profile keeps new-member key agreement at $O(1)$, preserves stable per-message latency, and incurs a one-time session-establishment overhead of approximately $+38\%$ (msgX-Γ) and $+53\%$ (msgX-Σ) relative to Olm. The protocol enables gradual migration from Olm without breaking Matrix federation.*

Resumo. *Este artigo apresenta o msgX, um protocolo de mensagens seguras para substituir a distribuição de sessões baseada no Olm em salas Matrix governamentais, combinando criptografia padronizada e desenvolvida pelo governo para apoiar interoperabilidade e soberania digital. Para mitigar ameaças quânticas, o msgX integra PQXDH, um Double Ratchet modificado e o msgGX, uma adaptação do Megolm para criptografia escalável em grupos. Em 14.040 execuções Android, distribuídas em dois dispositivos, 30 repetições cada e 234 configurações parametrizadas, foi realizada uma decomposição de custo por ramo em relação ao Olm, isolando os perfis msgX-Γ e msgX-Σ. Os resultados mostram que cada perfil mantém o acordo de chaves para novos membros em $O(1)$, preserva latência por mensagem estável e apresenta sobrecarga única de estabelecimento de sessão de aproximadamente $+38\%$ (msgX-Γ) e $+53\%$ (msgX-Σ) em relação ao Olm. O protocolo permite migração gradual a partir do Olm sem romper a federação Matrix.*

1. Introdução

Mensagens seguras evoluíram de sistemas de bate-papo na Internet em texto claro para plataformas de criptografia de ponta a ponta (E2EE) que protegem as comunicações mesmo em ambientes de rede hostis (e.g., Signal). Contudo, modelos modernos de segurança enfrentam a perspectiva de computadores quânticos. Mecanismos de chave pública, como Rivest-Shamir-Adleman (RSA) e criptografia de curvas elípticas (ECC), baseiam-se

em hipóteses de dificuldade vulneráveis ao algoritmo de Shor [Shor 1994]. Em resposta, o processo de criptografia pós-quântica (PQ) do Instituto Nacional de Padrões e Tecnologia (NIST) levou à padronização de novos algoritmos, incluindo o CRYSTALS-Kyber, padronizado como ML-KEM na FIPS-203 [Nist 2024]. Embora o computador quântico criptanaliticamente relevante (CRQC) ainda não esteja disponível, a ameaça de coletar agora, descriptografar depois (HNDL) torna urgente a migração, pois o tráfego criptografado capturado hoje pode ser descriptografado no futuro. Assim, a proteção PQ é essencial para o estabelecimento de chaves e para a confidencialidade das mensagens.

Protocolos modernos de mensagens seguras já iniciaram essa transição. Por exemplo, o Signal introduziu o Extended Diffie–Hellman Pós-Quântico (PQXDH) [Kret and Schmidt 2023], que amplia o Diffie–Hellman de curvas elípticas (ECDH) clássico com um mecanismo de encapsulamento de chaves (KEM) PQ, sugerindo que construções híbridas podem melhorar a resiliência durante o período de transição, preservando a segurança ainda que a componente clássica ou PQ seja posteriormente enfraquecida. Contudo, criptografia isoladamente é insuficiente nas comunicações governamentais. Soberania [1], controle da infraestrutura e independência em relação a provedores estrangeiros de tecnologia também são requisitos centrais. Esse contexto levou ao desenvolvimento de iniciativas soberanas de mensageria segura baseadas no Matrix [Matrix.org 2024], um protocolo aberto e descentralizado de comunicação, como o Tchap [Française 2019].

Este artigo apresenta o msgX, um protocolo criptográfico para a distribuição de sessões em salas Matrix para mensagens governamentais de alta garantia, adotando um projeto híbrido e de dois ramos. O ramo padrão utiliza algoritmos amplamente implantados, incluindo mecanismos PQ alinhados aos esforços atuais de padronização, para preservar a interoperabilidade com clientes e federações Matrix. Seu ramo governamental incorpora primitivas criptográficas soberanas, implementadas por meio da *libharpia* [Pacheco et al. 2022], para fortalecer a resiliência e a soberania digital.

As contribuições deste artigo são três: (i) o projeto do msgX, um protocolo criptográfico de dois ramos que substitui o Olm em implantações governamentais do Matrix, preservando a compatibilidade retroativa com clientes existentes; (ii) a especificação do msgX-PQXDH e do msgX-ratchet, bem como sua integração com o protocolo de mensagens em grupo msgGX; e (iii) uma avaliação experimental por decomposição de custo por ramo em hardware Android de uso corrente, isolando os perfis msgX-Γ e msgX-Σ em relação ao Olm e mostrando que os resultados são consistentes com a preservação dos custos $O(N)$ para distribuição de chaves de grupo e $O(1)$ para entrada de novos membros do Megolm, ao mesmo tempo que cada perfil adiciona um fator pós-quântico constante.

2. Fundamentação

A arquitetura dos protocolos modernos de mensagens seguras foi fortemente influenciada pelo Signal e posteriormente incorporada a sistemas como o Matrix. O Extended Triple Diffie-Hellman (X3DH) [Marlinspike and Perrin 2016] permite o estabelecimento assíncrono de sessões a partir do pacote de pré-chaves do destinatário, mesmo quando ele está *offline*. Em seguida, o Double Ratchet (DR) [Perrin and Marlinspike 2016] evolui a

[1]Usamos soberania digital como a capacidade de uma implantação nacional controlar, auditar e substituir as primitivas criptográficas usadas para comunicação, evitando dependência de um único fornecedor, implantação ou autoridade.

chave raiz, as chaves de cadeia e as chaves de mensagem. Esse processo contribui para o sigilo direto (FS) ao limitar a exposição de mensagens anteriores e para a segurança pós-comprometimento (PCS) ao permitir a recuperação da segurança após a incorporação de novo material secreto. Para enfrentar ameaças quânticas, o PQXDH [Kret and Schmidt 2023] combina ECDH clássico com um KEM pós-quântico e deriva o segredo da sessão a partir das duas contribuições, reduzindo a exposição ao risco de HNDL.

Esses mecanismos também fundamentam a E2EE do Matrix, protocolo descentralizado e federado no qual usuários de diferentes servidores podem se comunicar sem uma autoridade central [Matrix.org 2024]. O Megolm protege o plano de dados de todas as salas criptografadas, inclusive chats entre dois usuários, por meio de uma sessão de remetente por dispositivo. O Olm, baseado no DR do Signal, atua no plano de controle ao estabelecer canais entre dispositivos para distribuir o material das sessões Megolm. Após essa distribuição, cada mensagem pode ser cifrada uma única vez e publicada para todos os participantes [Matrix.org 2019, Matrix.org 2022]. A renovação periódica dessas sessões limita a exposição futura após comprometimentos. A Vodozemac implementa em Rust os protocolos Olm e Megolm usados no ecossistema Matrix [Matrix.org Foundation]. Embora modernize a base criptográfica dos clientes e preserve a retrocompatibilidade das sessões, a biblioteca não possui mecanismos PQ para acordo ou renovação de chaves.

Em um contexto mais amplo de mensageria segura, diferentes propostas têm investigado mecanismos de renovação PQ. O Triple Ratchet [Dodis et al. 2025], por exemplo, distribui material criptográfico PQ ao longo de várias mensagens com o uso de códigos de apagamento. Essa abordagem reduz a sobrecarga de banda e aumenta a robustez em fluxos de comunicação desbalanceados, embora introduza maior complexidade de sincronização e gerenciamento de estado.

Outros trabalhos concentram-se na avaliação e otimização dessa transição. Auerbach et al. [Auerbach et al. 2025] propõem uma metodologia baseada em Sparse Continuous Key Agreement (SCKA) e na métrica *Vulnerable Message Set* (VMS), que quantifica as mensagens expostas após o comprometimento de um estado criptográfico, fundamentandoo *ML-KEM Braid* do Signal [Schmidt and Connell 2025], que fragmenta e transmite progressivamente o material PQ em ambientes com restrição de banda. Esses trabalhos evidenciam o compromisso entre frequência de renovação, volume de dados e recuperação após comprometimentos.

No contexto de grupos, o Messaging Layer Security (MLS), padronizado pela IETF no RFC 9420 [Barnes et al. 2023], organiza os participantes em uma árvore criptográfica e faz o estado do grupo avançar por épocas. Inclusões, remoções e atualizações são consolidadas por mensagens de compromisso, permitindo FS e PCS com custos de atualização logarítmicos. A comunidade Matrix investiga o MLS como uma possível evolução de sua criptografia de grupos [Hodgson and Chathi 2023]. Essa integração enfrenta um desafio de conciliar a sequência linear de épocas do MLS com o complexo ambiente federado do Matrix. Em paralelo, a IETF desenvolve suítes PQ e híbridas para MLS baseadas em ML-KEM [Mahy and Barnes 2026].

Essas iniciativas representam trajetórias complementares para a evolução das mensagens seguras. O msgX atua na distribuição e renovação de sessões de grupo em implantações governamentais federadas, com o objetivo de preservar a interoperabilidade com

o Matrix, incorporar mecanismos criptográficos soberanos e reduzir a exposição ao risco de HNDL no ecossistema. Para isso, substitui os canais Olm usados no transporte do material de sessão por mecanismos híbridos PQ, mas mantém o modelo operacional do Megolm. Assim, preserva a distribuição de chaves de grupo em $O(N)$ e a entrada de novos membros em $O(1)$ sob a camada PQ adicionada, oferecendo uma trajetória incremental de migração com sobrecarga constante.

3. O Protocolo msgX

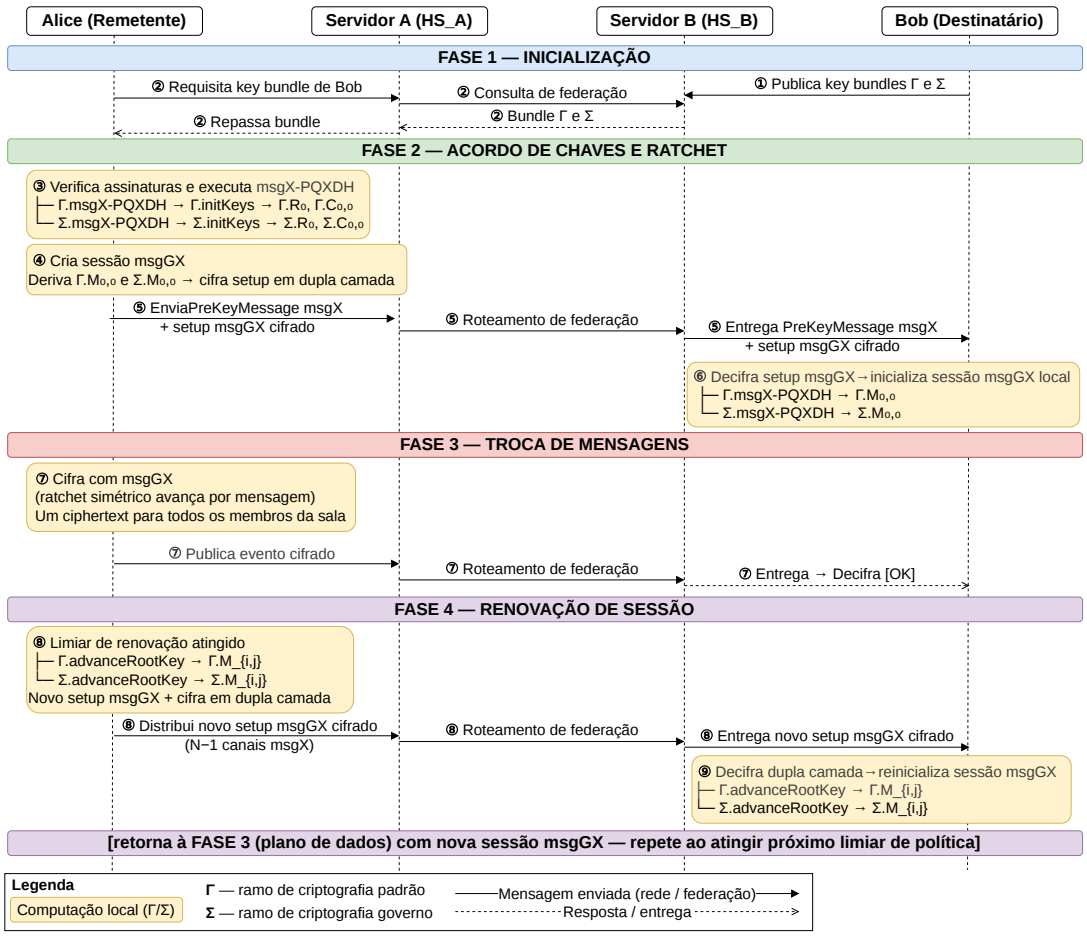

Figura 1. Fluxo fim a fim dos protocolos msgX e msgGX.

Embora o msgX introduza uma arquitetura distinta, seu desenvolvimento é orientado por ideias centrais do PQXDH e do Olm. Ao longo desta seção, Γ denota o ramo de criptografia padrão e Σ o ramo desenvolvido pelo governo. Todas as chaves, algoritmos e procedimentos recebem prefixos correspondentes. Antes de detalhar a estrutura interna do msgX, é importante explicar como ele se integra às salas Matrix e ao seu protocolo complementar de mensagens em grupo, o msgGX.

No Matrix, conversas criptografadas são salas, incluindo chats 1:1; portanto, o msgX mira o cenário de grupo ao proteger a distribuição e a renovação do material de sessão de grupo. O tráfego de grupo de alto volume é criptografado pelo msgGX, uma

adaptação do protocolo Megolm com quatro modificações principais: (i) dois *ratchets* simétricos paralelos, um por ramo, em vez de um único ratchet; (ii) criptografia autenticada com dados associados (AEAD) em dupla camada usando *message keys* específicas de cada ramo; (iii) constantes Hash-based Message Authentication Code (HMAC) separadas por domínio para garantir que as duas cadeias evoluam de forma independente; e (iv) distribuição de sessão por canais msgX em vez de Olm, possibilitando o estabelecimento de sessão PQ descrito na Seção 3.1. Isso preserva a comunicação eficiente em grupo, mas mantém o compromisso do Megolm: o comprometimento de um estado simétrico de grupo atual pode afetar mensagens futuras até a próxima renovação de sessão.

Para mitigar essa limitação, msgX e msgGX são usados em conjunto. O msgGX renova periodicamente sua sessão ao gerar um *setup* inicial novo, que deve ser entregue com segurança a cada participante da sala. O msgX fornece esse canal de entrega: ele transmite o *setup* da sessão msgGX individualmente a cada membro da sala. Nesse projeto, o msgX protege a distribuição da sessão de grupo, enquanto o msgGX criptografa as mensagens de grupo efetivamente trocadas na sala. Uma visão geral da interação entre um remetente e um membro da sala (i.e., Alice e Bob) é apresentada na Figura 1; a mesma etapa de distribuição é repetida para cada membro da sala. Como ilustrado, o msgX é estruturado em quatro fases principais: inicialização; acordo de chaves e ratchet; troca de mensagens; e renovação de sessão.

3.1. Inicialização

Esta fase envolve a geração e publicação antecipada das chaves públicas de Bob, que gera vários pares de chaves e envia a parte pública de cada um ao seu homeserver(vide etapa de setup ❶ na Figura 1). Para suportar tanto Γ quanto Σ, Bob gera dois pares de identity keys: $\Gamma.\mathsf{IK}_B$ e $\Sigma.\mathsf{IK}_B$. Ele também gera um conjunto de pares de ephemeral keys: $\Gamma.\mathsf{EK}_B$, $\Sigma.\mathsf{EK}_B$, e ephemeral keys pós-quânticas $\Gamma.\mathsf{PQE}_B$ e $\Sigma.\mathsf{PQE}_B$. Todas as chaves associadas à criptografia padrão são assinadas usando a identity key privada de Bob $\Gamma.\mathsf{IK}_B^{priv}$, enquanto as chaves governamentais são assinadas com $\Sigma.\mathsf{IK}_B^{priv}$. Note-se que identity keys fornecem uma identidade criptográfica de longo prazo, permitindo autenticação e estabelecimento de confiança. Ephemeral keys, por sua vez, são usadas para FS, garantindo que, mesmo se chaves de longo prazo forem comprometidas posteriormente, chaves de sessões passadas permaneçam seguras.

Bob então cria um key bundle composto pelas partes públicas de todas as chaves geradas, seus identificadores e assinaturas correspondentes. Esse key bundle é enviado ao homeserver, onde fica disponível para recuperação por outros usuários ❷. Isso permite que Alice inicie um acordo de chaves com Bob mesmo quando ele está offline, viabilizando o estabelecimento assíncrono e seguro de sessão.

3.2. Acordo de Chaves e Ratchet

Esta fase começa quando Alice recupera o key bundle de Bob a partir do homeserver. Ela verifica cada assinatura digital incluída no bundle usando $\Gamma.\mathsf{IK}_B^{pub}$ para a criptografia padrão e $\Sigma.\mathsf{IK}_B^{pub}$ para a criptografia governamental. Se qualquer verificação falhar, o protocolo é imediatamente abortado para evitar uma potencial violação de segurança. Uma vez que todas as assinaturas sejam verificadas com sucesso, Alice executa o msgX-PQXDH, como indicado em ❸. O procedimento completo é descrito na Figura 2.

Cada ramo do msgX-PQXDH, isto é, o ramo padrão (Γ) e o ramo governamental (Σ), realiza um encapsulamento KEM, denotado apenas por Enc na Figura 2, e três operações ECDH. O encapsulamento KEM produz um ciphertext e um shared secret para cada ramo, denotados por $\Gamma.\mathsf{ct}, \Gamma.\mathsf{ss}$ e $\Sigma.\mathsf{ct}, \Sigma.\mathsf{ss}$, respectivamente. As operações ECDH envolvem combinar as chaves privadas de Alice com as chaves públicas de Bob, usando identity keys e ephemeral keys. Como resultado, cada ECDH produz três shared secrets adicionais por ramo: $\Gamma.\mathsf{DH}_1, \Gamma.\mathsf{DH}_2, \Gamma.\mathsf{DH}_3$ e $\Sigma.\mathsf{DH}_1, \Sigma.\mathsf{DH}_2, \Sigma.\mathsf{DH}_3$. Para cada ramo, o shared secret do encapsulamento e os três segredos derivados por ECDH são concatenados para formar os segredos combinados $\Gamma.\mathsf{S}$ e $\Sigma.\mathsf{S}$. Esses valores são então usados como entradas para a HKDF dentro do procedimento msgX-ratchet.initKeys($\cdot$), responsável por derivar as root keys e chain keys iniciais do msgX-ratchet: $\Gamma.\mathsf{R}_0, \Gamma.\mathsf{C}_{0,0}$ e $\Sigma.\mathsf{R}_0, \Sigma.\mathsf{C}_{0,0}$.

Figura 2. Protocolo msgX.

Γ.msgX-PQXDH($\Gamma.\mathsf{IK}_A^{priv}, \Gamma.\mathsf{EK}_A^{priv}, \Gamma.\mathsf{EK}_B^{pub}, \Gamma.\mathsf{PQE}_B^{pub}$)

1 : $\Gamma.\mathsf{ss}, \Gamma.\mathsf{ct} \leftarrow \Gamma.\mathsf{Enc}(\Gamma.\mathsf{PQE}_B^{pub})$
2 : $\Gamma.\mathsf{DH}_1 \leftarrow \Gamma.\mathsf{ECDH}(\Gamma.\mathsf{IK}_A^{priv}, \Gamma.\mathsf{EK}_B^{pub})$
3 : $\Gamma.\mathsf{DH}_2 \leftarrow \Gamma.\mathsf{ECDH}(\Gamma.\mathsf{EK}_A^{priv}, \Gamma.\mathsf{IK}_B^{pub})$
4 : $\Gamma.\mathsf{DH}_3 \leftarrow \Gamma.\mathsf{ECDH}(\Gamma.\mathsf{EK}_A^{priv}, \Gamma.\mathsf{EK}_B^{pub})$
5 : $\Gamma.\mathsf{S} \leftarrow \Gamma.\mathsf{DH}_1 || \Gamma.\mathsf{DH}_2 || \Gamma.\mathsf{DH}_3 || \Gamma.\mathsf{ss}$

Σ.msgX-PQXDH($\Sigma.\mathsf{IK}_A^{priv}, \Sigma.\mathsf{EK}_A^{priv}, \Sigma.\mathsf{EK}_B^{pub}, \Sigma.\mathsf{PQE}_B^{pub}$)

1 : $\Sigma.\mathsf{ss}, \Sigma.\mathsf{ct} \leftarrow \Sigma.\mathsf{Enc}(\Sigma.\mathsf{PQE}_B^{pub})$
2 : $\Sigma.\mathsf{DH}_1 \leftarrow \Sigma.\mathsf{ECDH}(\Sigma.\mathsf{IK}_A^{priv}, \Sigma.\mathsf{EK}_B^{pub})$
3 : $\Sigma.\mathsf{DH}_2 \leftarrow \Sigma.\mathsf{ECDH}(\Sigma.\mathsf{EK}_A^{priv}, \Sigma.\mathsf{IK}_B^{pub})$
4 : $\Sigma.\mathsf{DH}_3 \leftarrow \Sigma.\mathsf{ECDH}(\Sigma.\mathsf{EK}_A^{priv}, \Sigma.\mathsf{EK}_B^{pub})$
5 : $\Sigma.\mathsf{S} \leftarrow \Sigma.\mathsf{DH}_1 || \Sigma.\mathsf{DH}_2 || \Sigma.\mathsf{DH}_3 || \Sigma.\mathsf{ss}$

Γ.msgX-ratchet.initKeys($\Gamma.\mathsf{S}$)

1 : $\Gamma.\mathsf{R}_0 || \Gamma.\mathsf{C}_{0,0} \leftarrow \mathsf{HKDF}(0, \Gamma.\mathsf{S}, \text{“OLM\_ROOT”}, 64)$

Σ.msgX-ratchet.initKeys($\Sigma.\mathsf{S}$)

1 : $\Sigma.\mathsf{R}_0 || \Sigma.\mathsf{C}_{0,0} \leftarrow \mathsf{HKDF}(0, \Sigma.\mathsf{S}, \text{“GC\_ROOT”}, 64)$

A sessão msgGX é então criada ❹. Seu setup inicial é criptografado sob a sessão msgX estabelecida, permitindo que Bob reconstrua o mesmo estado msgGX. As root keys e chain keys previamente derivadas produzem duas chaves de criptografia de mensagens, $\Gamma.\mathsf{M}_{0,0}$ e $\Sigma.\mathsf{M}_{0,0}$, usadas com AEAD nos ramos PQ. O setup inicial criptografado da sessão msgGX, juntamente com as chaves públicas usadas por Alice no protocolo msg-PQXDH e os ciphertexts $\Gamma.\mathsf{ct}$ e $\Sigma.\mathsf{ct}$, é enviado a Bob ❺. Do seu lado, Bob recebe e processa a mensagem ❻. Usando os dados recebidos, Bob executa o protocolo msgX-PQXDH e deriva os mesmos shared secrets combinados que Alice. A partir deles, ele obtém as mesmas root keys e chain keys do ratchet para a sessão msgX padrão e governamental. Com isso, Bob consegue derivar as chaves de criptografia de mensagens $\Gamma.\mathsf{M}_{0,0}$ e $\Sigma.\mathsf{M}_{0,0}$ usadas por Alice para criptografar o setup inicial da sessão msgGX. Em seguida, Bob descriptografa o setup inicial e inicializa sua própria sessão msgGX.

3.3. Troca de Mensagens

Os membros da sala trocam mensagens via msgGX, que avança seu próprio ratchet a cada nova mensagem ❼. A sessão msgGX deve ser renovada após um número predeterminado de mensagens trocadas ou um intervalo de tempo especificado, i.e., um novo setup inicial para a sessão msgGX deve ser gerado e transmitido com segurança aos membros da sala.

3.4. Renovação de Sessão

A sessão msgX já estabelecida é reutilizada para renovar a sessão msgGX.No entanto, antes de criptografar o novo setup inicial, o msgX-ratchet deve ser avançado ❽. Isso envolve atualizar as root keys e chain keys para derivar uma nova chave de criptografia de mensagens, que será usada para transmitir com segurança o novo setup inicial da sessão msgGX. A Figura 3 apresenta o msgX-ratchet.advanceRootKey(·), um procedimento para avançar a root key e derivar uma nova chain key. A abordagem proposta se baseia no algoritmo DR [Perrin and Marlinspike 2016], com alguns aprimoramentos introduzidos para atender aos requisitos de segurança e projeto do msgX.

Figura 3. Avançando as root keys no msgX-ratchet.

Γ.msgX-ratchet.advanceRootKey($\Gamma.\mathsf{R}_{i-1}, \Gamma.\mathsf{T}_{i-1}^{priv}, \Gamma.\mathsf{T}_{i}^{pub}, \Gamma.\mathsf{ct}_i$)

1 : $\Gamma.\mathsf{ss}_i \leftarrow \textbf{null}$
2 : **if** $\exists n \mid i = 2^N n - 1$ **then**
3 : $\quad(\Gamma.\mathsf{PQT}_i^{pub}, \Gamma.\mathsf{PQT}_i^{priv}) \leftarrow \Gamma.\mathsf{KGen}()$
4 : **else if** $\exists n \mid i = 2^N n$ **then**
5 : $\quad\Gamma.\mathsf{ss}_i \leftarrow \Gamma.\mathsf{Dec}(\Gamma.\mathsf{ct}_i, \Gamma.\mathsf{PQT}_i^{priv})$
6 : **endif**
7 : $\Gamma.\mathsf{R}_i || \Gamma.\mathsf{C}_{i,0} \leftarrow \mathsf{HKDF}(\Gamma.\mathsf{R}_{i-1}, \Gamma.\mathsf{ECDH}(\Gamma.\mathsf{T}_{i-1}^{priv}, \Gamma.\mathsf{T}_{i}^{pub}) || \Gamma.\mathsf{ss}_i, \text{“OLM\_RATCHET”}, 64)$

Σ.msgX-ratchet.advanceRootKey($\Sigma.\mathsf{R}_{i-1}, \Sigma.\mathsf{T}_{i-1}^{priv}, \Sigma.\mathsf{T}_{i}^{pub}, \Sigma.\mathsf{ct}_i$)

1 : $\Sigma.\mathsf{ss}_i \leftarrow \textbf{null}$
2 : **if** $\exists n \mid i = 2^N n - 1$ **then**
3 : $\quad(\Sigma.\mathsf{PQT}_i^{pub}, \Sigma.\mathsf{PQT}_i^{priv}) \leftarrow \Sigma.\mathsf{KGen}()$
4 : **else if** $\exists n \mid i = 2^N n$ **then**
5 : $\quad\Sigma.\mathsf{ss}_i \leftarrow \Sigma.\mathsf{Dec}(\Sigma.\mathsf{ct}_i, \Sigma.\mathsf{PQT}_i^{priv})$
6 : **endif**
7 : $\Sigma.\mathsf{R}_i || \Sigma.\mathsf{C}_{i,0} \leftarrow \mathsf{HKDF}(\Sigma.\mathsf{R}_{i-1}, \Sigma.\mathsf{ECDH}(\Sigma.\mathsf{T}_{i-1}^{priv}, \Sigma.\mathsf{T}_{i}^{pub}) || \Sigma.\mathsf{ss}_i, \text{“GC\_RATCHET”}, 64)$

O algoritmo da Figura 3 define o procedimento para atualizar as root keys e derivar chain keys em ambos os ramos criptográficos, padrão e governamental, sendo invocado sempre que uma renovação da sessão msgGX é necessária. Recebe-se como entrada as root keys anteriores $\Gamma.\mathsf{R}_{i-1}$ e $\Sigma.\mathsf{R}_{i-1}$, as ephemeral keys anteriores e atuais $\Gamma.\mathsf{T}_{i-1}, \Gamma.\mathsf{T}_i$ e $\Sigma.\mathsf{T}_{i-1}, \Sigma.\mathsf{T}_i$, juntamente com os ciphertexts $c\Gamma.\mathsf{T}_i$ e $c\Sigma.\mathsf{T}_i$, que podem conter segredos encapsulados PQ.

Dois tipos de operação são realizados dependendo do índice de etapa atual i. Se i corresponde ao padrão $2^N n - 1$, em que $N \in \mathbb{Z}$ é um valor predeterminado, novos pares de chaves pós-quânticas ($\Gamma.\mathsf{PQT}_i^{pub}, \Gamma.\mathsf{PQT}_i^{priv}$ e $\Sigma.\mathsf{PQT}_i^{pub}, \Sigma.\mathsf{PQT}_i^{priv}$) são gerados para cada ramo, e seus componentes públicos são enviados ao par anexados à mensagem msgX atual. Se $i = 2^N n$, os ciphertexts $c\Gamma.\mathsf{T}_i$ e $c\Sigma.\mathsf{T}_i$ são recebidos anexados à mensagem msgX e decapsulados usando as chaves privadas correspondentes para recuperar os shared secrets PQ $\Gamma.\mathsf{ss}_i$ e $\Sigma.\mathsf{ss}_i$. Esses segredos são usados apenas em intervalos específicos para injetar periodicamente entropia PQ no processo de evolução das chaves.

Por fim, as novas root keys e chain keys $\Gamma.\mathsf{R}_i, \Gamma.\mathsf{C}_{i,0}, \Sigma.\mathsf{R}_i, \Sigma.\mathsf{C}_{i,0}$ são derivadas usando uma HKDF, que recebe como entrada as root keys anteriores $\Gamma.\mathsf{R}_{i-1}$ e $\Sigma.\mathsf{R}_{i-1}$, o resultado de duas operações ECDH separadas, o shared secret PQ opcional $\Gamma.\mathsf{ss}_i$ e $\Sigma.\mathsf{ss}_i$, e um rótulo específico de domínio. Uma operação usa criptografia padrão e a outra usa criptografia desenvolvida pelo governo, entre as ephemeral keys relevantes. Esse mecanismo mantém FS e habilita PCS contra adversários clássicos, ao mesmo tempo que fornece PCS parcial na presença de atacantes com capacidade quântica. Quanto menor o valor de N, mais rápido é o efeito de “healing” em caso de ataque quântico.

Observe que o algoritmo executado pelo destinatário difere ligeiramente. No índice $i = 2^N n$, em vez de gerar um par de chaves ou realizar decapsulamento, o destinatário deve realizar um encapsulamento KEM para derivar os shared secrets $\Gamma.\mathsf{ss}_i$ e $\Sigma.\mathsf{ss}_i$, usando as chaves públicas recebidas $\Gamma.\mathsf{PQT}_i^{pub}$ e $\Sigma.\mathsf{PQT}_i^{pub}$, respectivamente. O restante do algoritmo permanece inalterado. Se a mesma parte que renovou anteriormente a sessão msgGX atual tentar renová-la novamente, ela não terá recebido uma nova ephemeral key do outro par (isto é, $\Gamma.\mathsf{T}_i^{pub}$ e $\Sigma.\mathsf{T}_i^{pub}$) e, portanto, não poderá avançar as root keys. Nesse caso, apenas as chain keys são avançadas usando o procedimento msgX-ratchet.advanceChainKey(·), conforme descrito na Figura 4.

Para avançar apenas as chain keys $\Gamma.\mathsf{C}_{i,j}$ e $\Sigma.\mathsf{C}_{i,j}$, as chain keys anteriores $\Gamma.\mathsf{C}_{i,j-1}$ e $\Sigma.\mathsf{C}_{i,j-1}$ são usadas como chaves em HMACs separados, com um valor fixo usado como mensagem de entrada. Uma vez atualizadas as chain keys, as message keys correspondentes também podem ser derivadas seguindo um procedimento semelhante, o msgX-ratchet.deriveMessageKey(·). As novas message keys $\Gamma.\mathsf{M}_{i,j}$ e $\Sigma.\mathsf{M}_{i,j}$ são geradas usando HMACs (Figura 5), em que as chain keys atualizadas $\Gamma.\mathsf{C}_{i,j}$ e $\Sigma.\mathsf{C}_{i,j}$ servem como chaves dos HMACs e um valor fixo é usado como mensagem de entrada.

Figura 4. Evolução das chain keys no msgX-ratchet.

Γ.msgX-ratchet.advanceChainKey($\Gamma.\mathsf{C}_{i,j-1}$)	Σ.msgX-ratchet.advanceChainKey($\Sigma.\mathsf{C}_{i,j-1}$)
1 : $\Gamma.\mathsf{C}_{i,j} \leftarrow \mathsf{HMAC}(\Gamma.\mathsf{C}_{i,j-1}, 0\text{x}2)$	1 : $\Sigma.\mathsf{C}_{i,j} \leftarrow \mathsf{HMAC}(\Sigma.\mathsf{C}_{i,j-1}, 0\text{x}4)$

Figura 5. Derivação das message keys no msgX-ratchet.

Γ.msgX-ratchet.deriveMessageKey($\Gamma.\mathsf{C}_{i,j}$)	Σ.msgX-ratchet.deriveMessageKey($\Sigma.\mathsf{C}_{i,j}$)
1 : $\Gamma.\mathsf{M}_{i,j} \leftarrow \mathsf{HMAC}(\Gamma.\mathsf{C}_{i,j}, 0\text{x}1)$	1 : $\Sigma.\mathsf{M}_{i,j} \leftarrow \mathsf{HMAC}(\Sigma.\mathsf{C}_{i,j}, 0\text{x}3)$

Com as novas message keys, o setup inicial da sessão pode ser compartilhado com segurança com cada membro da sala usando o esquema de criptografia em dupla camada baseado em criptografia simétrica padrão e desenvolvida pelo governo. No lado receptor, a mensagem criptografada contendo o novo setup inicial da sessão msgGX pode ser descriptografada ao avançar as root keys, chain keys e message keys correspondentes ❾. Uma vez descriptografada, a sessão msgGX pode ser reconstruída por cada destinatário, e a sala retoma a troca de mensagens usando a sessão atualizada.

3.5. Compatibilidade Retroativa

O msgX oferece suporte a um perfil de implantação compatível com Olm para interoperabilidade com clientes Matrix existentes no qual as etapas criptográficas desenvolvidas pelo governo são omitidas. No ramo padrão, o msgX-PQXDH (Figura 2) inclui um componente opcional $\Gamma.\mathsf{ss}$, que pode ser desabilitado para corresponder ao comportamento original do Olm baseado em X3DH. Analogamente, operações PQ envolvendo $\Gamma.\mathsf{ss}_i$ são omitidas em Γ.msgX-ratchet.advanceRootKey(·). Quando implementado de forma modular, o msgX pode emular o Olm ao desabilitar recursos PQ e desenvolvidos pelo governo, permitindo migração gradual em vez de uma transição obrigatória em data única.

3.6. Algoritmos Padrão

A Tabela 1 resume as primitivas criptográficas usadas nos ramos padrão e desenvolvido pelo governo do protocolo msgX. A arquitetura híbrida assegura agilidade criptográfica e aumenta a resiliência contra adversários clássicos e quânticos.

Tabela 1. Algoritmos usados no protocolo msgX.

	Criptografia padrão	Criptografia desenvolvida pelo governo
Simétrica	AES-256-CBC	Segurança equivalente a 256 bits
Curva elíptica (acordo de chaves)	Curve25519 ou Curve448	Segurança equivalente a 256 ou 448 bits
Curva elíptica (assinatura)	Ed25519 ou Ed448	Segurança equivalente a 256 ou 448 bits
PQC (encapsulamento de chaves)	ML-KEM ou Katana	KEM PQC com nível de segurança de 256 bits
HMAC	HMAC-SHA-256	HMAC-SHA-256
HKDF	Baseada em HMAC-SHA-256	Baseada em HMAC-SHA-256

4. Análise de Segurança

4.1. Contexto de Segurança Herdado

O Olm [Matrix.org 2024] fornece confidencialidade, autenticidade, FS e PCS para canais par a par, embora a PCS dependa da assinatura de pré-chaves que a especificação não impõe por padrão [Albrecht et al. 2024], deixando espaço para ataques de compartilhamento indevido de chaves e personificação. O Megolm estende o Olm para grupos com um ratchet simétrico unidirecional, enfraquecendo FS e PCS pois estados anteriores do ratchet podem persistir em armazenamento, e a renovação de sessão é necessária para restaurar a segurança futura após um comprometimento. No nível do ecossistema, análises anteriores [Albrecht et al. 2023, Li et al. 2023] mostram que um homeserver malicioso pode explorar seu controle sobre a associação às salas, as listas de dispositivos e o roteamento de eventos para comprometer a confiança dos clientes. Essas vulnerabilidades decorrem da própria arquitetura do ecossistema e também afetam o msgX. Ainda assim, sua camada criptográfica foi projetada para preservar a segurança mesmo diante de um servidor malicioso que atue de forma passiva.

4.2. Propriedades de Segurança do msgX

São considerados adversários capazes de interceptar ou modificar ciphertexts, homeservers potencialmente maliciosos, adversários com capacidade quântica que empreguem HNDL e adversários capazes de comprometer apenas um dos ramos criptográficos. Para mitigar essas ameaças, todas as chaves incluídas no bundle de Bob são assinadas e verificadas por Alice antes da execução do msgX-PQXDH. Além disso, a integridade dos dados é protegida por AEAD em ambos os ramos.

O msgX-ratchet fornece FS em dois níveis: cada chamada `advanceRootKey` descarta root keys e chain keys anteriores, tornando irrecuperáveis setups passados de sessões msgGX, e o ratchet simétrico do msgGX deriva uma chave única por mensagem. A PCS segue a mesma estrutura: `advanceRootKey` reinjeta entropia assimétrica a cada 2^N renovações via ECDH e, nas fronteiras de KEM, um encapsulamento PQ novo, permitindo recuperação após um ciclo completo. O intervalo de renovação R do msgGX então

controla quão rapidamente um estado de sessão de grupo comprometido é substituído, conforme detalhado na Seção 5.

O projeto de dois ramos fornece redundância contra ameaças ortogonais: o setup da sessão msgGX é criptografado independentemente sob Γ e Σ, de modo que um adversário deve comprometer ambos os ramos simultaneamente. Um futuro CRQC capaz de quebrar o ramo padrão não comprometeria o ramo Σ. Da mesma forma, uma eventual fraqueza nas primitivas soberanas não afetaria o ramo Γ. A resistência a HNDL decorre da mesma construção, pois o msgX-PQXDH vincula o segredo de sessão tanto a um componente ECDH clássico quanto a um componente KEM PQ em cada ramo. Por fim, o msgX exige a assinatura das pré-chaves, mitigando a fraqueza identificada no Olm [Albrecht et al. 2024]. Os ataques no nível do homeserver [Albrecht et al. 2023, Li et al. 2023], por sua vez, permanecem fora do escopo criptográfico e devem ser tratados por mecanismos de segurança da implantação. As propriedades de segurança acima dependem de hipóteses de composição que devem ser verificadas formalmente em trabalhos futuros.

5. Avaliação Experimental

5.1. Objetivo, Perfis e Configuração Experimental

A avaliação teve como objetivo quantificar o custo marginal dos ramos criptográficos do msgX em relação ao Olm. Para isso, foram considerados três perfis de execução. O primeiro perfil corresponde ao *Olm*, linha de base clássica com X3DH e Double Ratchet. O segundo perfil corresponde ao *msgX*-Γ, que isola o ramo padrão com msgX-PQXDH, ML-KEM-1024 e Curve25519. O terceiro perfil corresponde ao *msgX*-Σ, que isola o ramo governamental com KEM PQC de nível 256 bits em conjunto com Curve448. Cada execução isola um único perfil de protocolo, permitindo decomposição do custo por ramo sem confundir as sobrecargas dos dois ramos do msgX.

Os experimentos foram executados em dois dispositivos: um Samsung Galaxy S25 Ultra (Snapdragon 8 Elite, ARM64, Android 15, 12 GB RAM, Wi-Fi 5) e um Samsung Galaxy S23 (Snapdragon 8 Gen 2, ARM64, Android 15, 8 GB RAM, Wi-Fi 5), ambos conectados ao mesmo *homeserver* Synapse local com PostgreSQL 15. Os resultados quantificam a sobrecarga em rede local controlada, portanto, a latência de federação em escala de Internet constitui trabalho futuro. Synapse e banco de dados foram apagados antes de cada execução individual para eliminar contaminação de estado.

5.2. Plano Experimental e Métricas

O plano experimental organiza-se em doze cenários de *benchmark*. As métricas coletadas abrangem latência de *bootstrap*, latência média por mensagem em regime estacionário, *overhead* nas mensagens de renovação do *ratchet*, custo de rotação de sessão msgGX, tempo de distribuição de chave de grupo, latência de entrada de novo membro (*handshake* PQXDH), tamanho do *payload* de OTK e consulta de chaves.

Três fatores foram variados, cada um com 4 níveis: (i) fator R (intervalo de *rekeying*), com níveis $R \in \{16, 32, 64, 128\}$ mensagens, que controla de quantas em quantas mensagens uma nova sessão OLM PQ é negociada; (ii) fator N (comprimento de sessão), com níveis $N \in \{32, 64, 128, 256\}$ mensagens, que verifica se há degradação com sessões mais longas; e (iii) fator G (tamanho do grupo), com níveis $G \in \{8, 16, 32, 64\}$ membros,

que caracteriza a escalabilidade da distribuição de chaves Megolm. Cenários cujo resultado não depende de R, N ou G são executados com $R = 32$ fixo. A Tabela 2 detalha os doze cenários, que foram repetidos 30 vezes por dispositivo. O conjunto de dados totaliza 14.040 execuções: 78 cfg. $\times$ 3 protocolos $\times$ 30 repetições $\times$ 2 dispositivos.

Tabela 2. Plano experimental por fator e número de configurações por protocolo.

Fator	Cenários	Cfg.
Fixo (R=32)	*Bootstrap*; rotação msgGX; *payload* OTK; consulta de chaves; persistência; reposição de OTKs	6×1
R	Renovação do *ratchet*; *broadcast* de *re-key* em grupo	2×4
$R\times N$	Latência sustentada por mensagem	16
$R\times G$	Distribuição de chave de grupo; entrada de novo membro; saída de membro	3×16
Total		**78**

5.3. Metodologia Estatística e Hipóteses

Adotamos o seguinte protocolo de análise para cada comparação. (1) Normalidade: Shapiro-Wilk; (2) Duas amostras independentes: teste-t de Welch quando as duas amostras são compatíveis com normalidade; Mann-Whitney U caso contrário; (3) Três protocolos: ANOVA unidirecional quando a comparação é paramétrica; Kruskal-Wallis quando a normalidade é rejeitada; (4) Tendência com G: Spearman para verificar associação monotônica entre tamanho do grupo e latência.

Todos os resultados reportam média $\pm$ IC$_{95\%}$. Cada hipótese define explicitamente H_0, H_A, nível de significância $\alpha = 0{,}05$ e decisão baseada no valor-p: rejeita-se H_0 se $p < \alpha$; caso contrário, não há evidência suficiente para rejeitá-la. Foram definidas 7 hipóteses (H_1–H_7, $\alpha = 0{,}05$), organizadas em 2 grupos. O primeiro grupo (H_1–H_5) avalia o comportamento dos protocolos em ambos os dispositivos: H_1–H_3 comparam latências de *bootstrap* entre os 3 perfis (Mann-Whitney e Welch-t); H_4 avalia se há diferença global na latência sustentada entre os protocolos por meio do teste de Kruskal–Wallis; H_5 verifica ausência de custo de renovação do *ratchet* (pareado por execução). O segundo grupo (H_6–H_7) compara os 2 dispositivos: H_6 e H_7 testam se *bootstrap* e latência sustentada PQ diferem entre S23 e S25 Ultra (Welch-t/Mann-Whitney).

5.4. Custo de Estabelecimento e Latência Sustentada

A Figura 6 compara as latências de estabelecimento da sessão e de processamento sustentado das mensagens nos dois dispositivos, enquanto a Tabela 4 reúne as principais métricas de desempenho. O custo de estabelecimento do msgX concentra-se no *handshake* msgX-PQXDH. No S25 Ultra, a latência de *bootstrap* aumenta de 717 ± 20 ms com o Olm para 989 ± 36 ms com o msgX-Γ, correspondendo a um acréscimo de 38%. Para o msgX-Σ, a latência alcança 1095 ± 38 ms, um aumento de 53% em relação ao Olm. Ambas as diferenças são estatisticamente significativas (H_1 e H_2, $p < 10^{-10}$). A diferença entre os perfis Γ e Σ também é significativa (H_3, $p < 10^{-3}$), evidenciando a sobrecarga adicional do perfil governamental em relação ao baseado em ML-KEM-1024.

O mesmo comportamento é observado no S23, cujas latências de *bootstrap* são de 934 ± 28 ms para o Olm, 1286 ± 48 ms para o msgX-Γ e 1425 ± 57 ms para o msgX-Σ. Em comparação com o S25 Ultra, as diferenças são estatisticamente significativas para os três protocolos (H_6 rejeitada, $p < 0{,}001$), refletindo o maior custo de processamento no dispositivo menos recente.

Tabela 3. Hipóteses estatísticas avaliadas na Seção 5.

Hip.	Pergunta	H_0	H_A	Teste	Valor-p
H_1	*Bootstrap* msgX-Γ vs. Olm (ambos disp.)	$\mu_\Gamma = \mu_{\mathrm{Olm}}$	$\mu_\Gamma \neq \mu_{\mathrm{Olm}}$	Mann-Whitney	$< 0{,}001$
H_2	*Bootstrap* msgX-Σ vs. Olm (ambos disp.)	$\mu_\Sigma = \mu_{\mathrm{Olm}}$	$\mu_\Sigma \neq \mu_{\mathrm{Olm}}$	Mann-Whitney	$< 0{,}001$
H_3	*Bootstrap* msgX-Σ vs. msgX-Γ (ambos disp.)	$\mu_\Sigma = \mu_\Gamma$	$\mu_\Sigma \neq \mu_\Gamma$	Welch-t	$0{,}001$
H_4	Latência sustentada entre protocolos (ambos disp.)	$\mu_{\mathrm{Olm}} = \mu_\Gamma = \mu_\Sigma$	Pelo menos uma média difere	Kruskal-Wallis	$< 0{,}001$
H_5	Renovação do *ratchet* vs. mensagem normal	$\mu_{\mathrm{rekey}} = \mu_{\mathrm{normal}}$	$\mu_{\mathrm{rekey}} \neq \mu_{\mathrm{normal}}$	Pareado (t)	Olm: 0,29; Γ: 0,67; Σ: 0,02
H_6	*Bootstrap* S23 vs. S25 Ultra	$\mu_{\mathrm{S23}} = \mu_{\mathrm{S25U}}$	$\mu_{\mathrm{S23}} \neq \mu_{\mathrm{S25U}}$	Welch-t/Mann-Whitney	$< 0{,}001$
H_7	Sustentado PQ S23 vs. S25 Ultra	$\mu_{\mathrm{S23}} = \mu_{\mathrm{S25U}}$	$\mu_{\mathrm{S23}} \neq \mu_{\mathrm{S25U}}$	Welch-t/Mann-Whitney	$< 0{,}001$

μ representa a média populacional da métrica avaliada. A escolha paramétrica/não-paramétrica é determinada pelo teste de Shapiro-Wilk ($\alpha = 0{,}05$). H_5: não rejeitada para Olm e msgX-Γ; rejeitada para msgX-Σ ($+2{,}2\%$, $p = 0{,}02$), diferença de magnitude negligenciável na prática.

Tabela 4. Resumo de desempenho nos dois dispositivos (médias $\pm$ IC$_{95\%}$, $n = 30$). Métricas algorítmicas independem do dispositivo.

	Olm		msgX-Γ		msgX-Σ	
Métrica	S25 Ultra	S23	S25 Ultra	S23	S25 Ultra	S23
Bootstrap (ms)	717±20	934±28	989±36	1286±48	1095±38	1425±57
Sustentado (ms/msg)	444±2	421±6	651±2	705±10	794±9	853±12
Overhead ratchet (R=128)	≈0%		≈0%		≈0%	
Overhead rotação msgGX	+12,1%		+9,0%		+11,2%	
Payload OTK (B)	395		3.700		4.042	
Distrib. grupo G=64 (s)	21,1	22,1	34,3	37,0	39,7	42,6
PQXDH novo membro (ms)	489±5	525±1	787±8	852±2	942±24	1022±3

H_5 não rejeitada para Olm e Γ; overhead marginal ($+2{,}2\%$) para Σ. *Payload* OTK: $9{,}4\times$/$10{,}2\times$ o Olm (independe do dispositivo). PQXDH com novo membro: custo $O(1)$ em G (sem tendência monotônica detectada).

Após o estabelecimento da sessão, a latência por mensagem permanece estável para todos os níveis do fator $N \in \{32, 64, 128, 256\}$, sem evidência de degradação à medida que a sessão se prolonga. No S25 Ultra, foram observadas latências de 444 ± 2 ms/msg para o Olm, 651 ± 2 ms/msg para o msgX-Γ e 794 ± 9 ms/msg para o msgX-Σ. O teste global de Kruskal–Wallis indica diferenças estatisticamente significativas entre as três distribuições (H_4, $p < 10^{-10}$).

No S23, a latência sustentada do Olm é de 421 ± 6 ms/msg, valor próximo aos 444 ms/msg observados no S25 Ultra. Esse resultado é consistente com o predomínio da componente de rede no protocolo clássico. Como os dois dispositivos utilizam o mesmo servidor Synapse, as diferenças de processamento entre os respectivos SoCs permanecem dentro da variabilidade de rede observada.

Por outro lado, o maior peso das operações criptográficas torna mais perceptível a diferença entre os dispositivos nos protocolos PQ. O S23 apresenta latências de 705 ± 10 ms/msg para o msgX-Γ, um aumento de $8{,}2\%$ em relação ao S25 Ultra, e de 853 ± 12 ms/msg para o msgX-Σ, correspondente a um aumento de $7{,}4\%$. Embora estatis-

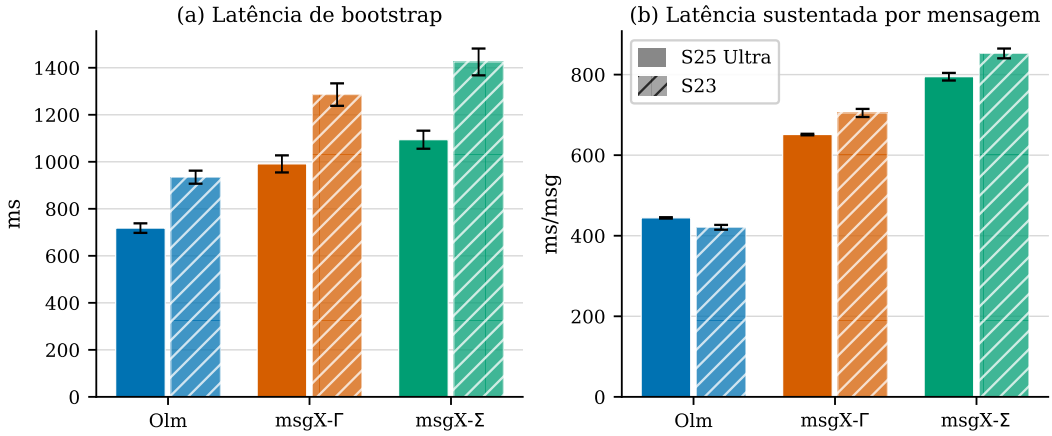

Figura 6. (a) Latência de *bootstrap* (descriptografia da 1ª mensagem, *handshake* PQXDH) e (b) latência sustentada por mensagem ($R = 32$), para os três protocolos em dois dispositivos. Barras sólidas: S25 Ultra; barras hachuradas: S23. Barras de erro: $\pm$ IC$_{95\%}$ ($n = 30$).

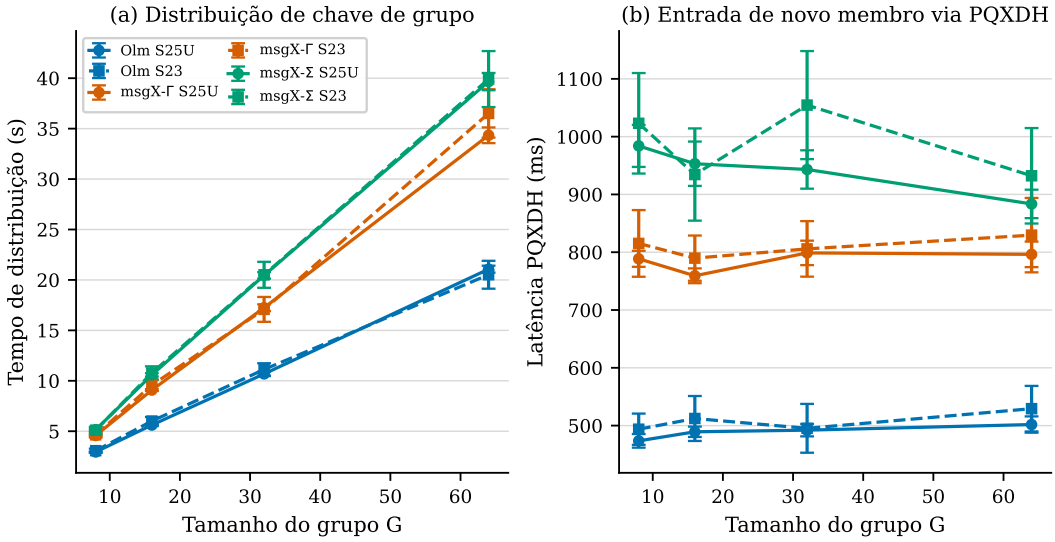

Figura 7. Escalabilidade de grupo (média e IC$_{95\%}$, $n = 30$). Linhas contínuas: S25 Ultra; tracejadas: S23. (a) Distribuição de chave cresce aproximadamente O(G) para todos os níveis do fator G. (b) PQXDH de novo membro permanece aproximadamente constante em G.

ticamente significativas (H_7, $p < 10^{-10}$), essas diferenças possuem magnitude moderada.

5.5. Overhead de Ratchet, Escalabilidade de Grupo e Payload OTK

A renovação do *ratchet* não apresenta custo estatisticamente significativo no S25 Ultra para o Olm ($p = 0{,}29$) nem para o msgX-Γ ($p = 0{,}67$), de modo que H_5 não é rejeitada nesses casos. Para o msgX-Σ, observa-se uma sobrecarga marginal de $+2{,}2\%$ ($p = 0{,}02$), equivalente a $\approx$17 ms por mensagem de renovação, valor pequeno diante da variabilidade de rede observada. A sobrecarga da rotação de sessão msgGX também permanece moderada, com $+12{,}1\%$ para o Olm, $+9{,}0\%$ para o msgX-Γ e $+11{,}2\%$ para o msgX-Σ, pois a rotação reutiliza a sessão existente sem realizar um novo *claim* de OTK.

Quanto à escalabilidade de grupo, a Figura 7 mostra que o tempo de distribuição de chaves cresce linearmente com os níveis do fator G em ambos os dispositivos, conforme esperado do modelo O(G) do Megolm. Para $G = 64$, os tempos são $21{,}1$ s / $22{,}1$ s para o Olm, $34{,}3$ s / $37{,}0$ s para o msgX-Γ e $39{,}7$ s / $42{,}6$ s para o msgX-Σ, sendo cada par correspondente ao S25 Ultra e ao S23, respectivamente.

Em contraste, o *handshake* PQXDH para a entrada de um novo membro não apresenta crescimento monotônico com o número de participantes existentes. Embora o teste de Kruskal–Wallis detecte variação entre os níveis de G, não se observa tendência de aumento. No msgX-Σ executado no S25 Ultra, por exemplo, a latência diminui de 979 ms em $G = 8$ para 885 ms em $G = 64$. Os valores médios são 489 ms / 525 ms para o Olm, 787 ms / 852 ms para o msgX-Γ e 942 ms / 1022 ms para o msgX-Σ. Assim, o msgX adiciona um custo PQ constante ao estabelecimento da sessão com o novo membro, preservando o comportamento assintótico O(G) do Megolm na redistribuição de chaves.

Por fim, o maior custo de largura de banda é o *payload* de OTK, que cresce de 395 B no Olm para 3.700 B no msgX-Γ ($9{,}4\times$) e 4.042 B no msgX-Σ ($10{,}2\times$). Tais valores são determinados pelos algoritmos e independem do dispositivo. Como a latência de *claim* permanece entre 18 e 22 ms, o principal impacto prático está no volume de material criptográfico armazenado e distribuído pelo *homeserver* quando há muitos usuários.

5.6. Síntese e Ameaças à Validade

Os resultados mostram que msgX-Γ e msgX-Σ introduzem sobrecarga constante em relação ao Olm, sem alterar as propriedades assintóticas herdadas do Megolm. Os principais custos concentram-se no *bootstrap* e no *payload* de OTK. A latência sustentada por mensagem, por sua vez, é predominantemente influenciada pela rede no protocolo clássico e apresenta diferenças moderadas entre os dispositivos nos protocolos PQ. Todavia, os experimentos realizados em uma rede local controlada suscitam avaliações adicionais para generalização a redes com maior latência.Além disso, os ramos Γ e Σ foram avaliados isoladamente, e o custo da configuração combinada $\Gamma+\Sigma$ não foi medido diretamente.

6. Conclusão

A ameaça de ataques HNDL torna urgente a transição pós-quântica em mensagens governamentais, enquanto a soberania digital exige independência de primitivas cujas futuras vulnerabilidades não possam ser auditadas nem substituídas pelo operador nacional. O msgX combina criptografia PQ padronizada e primitivas soberanas de forma interdependente: a quebra isolada de um ramo não revela texto claro, e o estabelecimento de sessão via msgX-PQXDH vincula o segredo a ambos os componentes de cada ramo, mitigando HNDL mesmo se um único vetor criptográfico for comprometido.

Ademais, o custo da proteção PQ soberana é viável em hardware Android atual e escalável para grupos de até 64 membros, oferecendo um caminho concretopara implantações nacionais do Matrix. Prioridades futuras incluem: (i) verificação formal das propriedades de composição dos dois ramos, requisito para adoção por órgãos de padronização; (ii) avaliação da configuração combinada $\Gamma+\Sigma$, que representa a implantação plena e não teve custo medido diretamente; (iii) experimentos em hardware restrito e redes de maior latência;e (iv) otimização do *payload* de OTK, cujo aumento de $9{,}4\times$/$10{,}2\times$ frente ao Olm é o principal impacto de largura de banda em servidores com muitos usuários.

Referências

[Albrecht et al. 2023] Albrecht, M. R., Celi, S., Dowling, B., and Jones, D. (2023). Practically-exploitable cryptographic vulnerabilities in matrix. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 164–181. IEEE. pages 9, 10

[Albrecht et al. 2024] Albrecht, M. R., Dowling, B., and Jones, D. (2024). Device-oriented group messaging: a formal cryptographic analysis of matrix'core. In *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE. pages 9, 10

[Auerbach et al. 2025] Auerbach, B., Dodis, Y., Jost, D., Katsumata, S., and Schmidt, R. (2025). How to compare bandwidth constrained two-party secure messaging protocols: a quest for a more efficient and secure post-quantum protocol. In *Proceedings of the 34th USENIX Conference on Security Symposium*, pages 6717–6736. pages 3

[Barnes et al. 2023] Barnes, R., Beurdouche, B., Robert, R., Millican, J., Omara, E., and Cohn-Gordon, K. (2023). The messaging layer security (mls) protocol. RFC 9420, Internet Engineering Task Force. pages 3

[Dodis et al. 2025] Dodis, Y., Jost, D., Katsumata, S., Prest, T., and Schmidt, R. (2025). Triple ratchet: A bandwidth efficient hybrid-secure signal protocol. In *Annual Int. Conference on the Theory and Applications of Cryptographic Techniques*, pages 302–331. Springer. pages 3

[Française 2019] Française, R. (2019). Tchap. `https://www.tchap.gouv.fr/`. pages 2

[Hodgson and Chathi 2023] Hodgson, M. and Chathi, H. (2023). A giant leap forwards for encryption with MLS. `https://matrix.org/blog/2023/07/a-giant-leap-with-mls/`. Published July 18, 2023. Accessed July 29, 2026. pages 3

[Kret and Schmidt 2023] Kret, E. and Schmidt, R. (2023). The pqxdh key agreement protocol. *`https://signal.org/docs/specifications/pqxdh/pqxdh.pdf`*. pages 2, 3

[Li et al. 2023] Li, H., Wu, Y., Huang, R., Mi, X., Hu, C., and Guo, S. (2023). Demystifying decentralized matrix communication network: Ecosystem and security. In *2023 IEEE 29th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 260–267. IEEE. pages 9, 10

[Mahy and Barnes 2026] Mahy, R. and Barnes, R. L. (2026). Ml-kem and hybrid cipher suites for messaging layer security. Internet-Draft draft-ietf-mls-pq-ciphersuites-04, Internet Engineering Task Force. Work in Progress. pages 3

[Marlinspike and Perrin 2016] Marlinspike, M. and Perrin, T. (2016). The x3dh key agreement protocol. *Open Whisper Systems*, 283(10):10. pages 2

[Matrix.org 2019] Matrix.org (2019). Olm: A cryptographic ratchet. `https://gitlab.matrix.org/matrix-org/olm/-/blob/master/docs/olm.md`. libolm repository. pages 3

[Matrix.org 2022] Matrix.org (2022). Megolm group ratchet. `https://gitlab.matrix.org/matrix-org/olm/-/blob/master/docs/megolm.md`. libolm repository. pages 3

[Matrix.org 2024] Matrix.org (2024). Matrix specification. `https://spec.matrix.org/`. pages 2, 3, 9

[Matrix.org Foundation] Matrix.org Foundation. Vodozemac: An implementation of Olm and Megolm in pure Rust. `https://github.com/matrix-org/vodozemac`. Accessed July 29, 2026. pages 3

[Nist 2024] Nist, G. M. D. (2024). Module-lattice-based key-encapsulation mechanism standard. Technical report, National Institute of Standards and Technology, Gaithersburg, MD. pages 2

[Pacheco et al. 2022] Pacheco, R., Braga, D., Passos, I., Araújo, T., Lagrota, V., and Coutinho, M. (2022). libharpia: a new cryptographic library for brazilian elections. In *Anais do XXII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pages 250–263. SBC. pages 2

[Perrin and Marlinspike 2016] Perrin, T. and Marlinspike, M. (2016). The double ratchet algorithm. *GitHub wiki*, 112(4). pages 2, 7

[Schmidt and Connell 2025] Schmidt, R. and Connell, G. (2025). The ml-kem braid protocol. Signal Messenger Specification. Available at: `https://signal.org/docs/specifications/mlkembraid/`. pages 3

[Shor 1994] Shor, P. W. (1994). Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. Ieee. pages 2