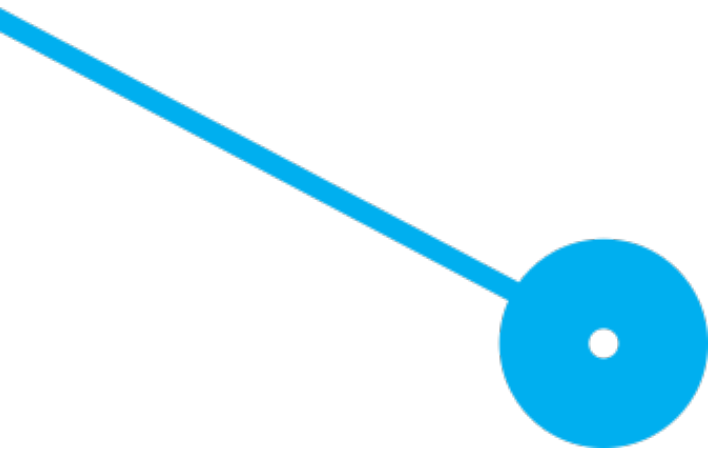
ESCOLA
SUPERIOR
DE TECNOLOGIA
E GESTÃO
POLITÉCNICO
DO PORTO

CTeSP CURSO TÉCNICO SUPERIOR PROFISSIONAL

Análise Forense Comportamental da Aplicação Session Desktop

Ryan da Silva Barbosa — 8240758
Igor Gabriel Macedo Araújo — 8240754
Hugo Danial da Silva Correia — 8240532

31 de dezembro de 2025



ESCOLA
SUPERIOR
DE TECNOLOGIA
E GESTÃO
POLITÉCNICO
DO PORTO

CTeSP CURSO TÉCNICO SUPERIOR PROFISSIONAL

Análise Forense Comportamental da Aplicação Session Desktop

Ryan da Silva Barbosa — 8240758
Igor Gabriel Macedo Araújo — 8240754
Hugo Danial da Silva Correia — 8240532

Sob Orientação de:

Jorge Gonçalo

Resumo

A crescente utilização de aplicações de mensagens focadas na privacidade apresenta novos desafios para a análise forense digital. O presente trabalho realiza uma análise forense comportamental à aplicação Session Desktop, uma plataforma de mensagens descentralizada que utiliza encriptação de ponta a ponta, não requer número de telefone para registo, e armazena os dados localmente com encriptação SQLCipher (AES-256).

Através de uma metodologia sistemática composta por nove cenários de teste, foi mapeada a estrutura da base de dados SQLite — identificando 19 tabelas, 25 índices e 3 *triggers* — e analisado o comportamento resultante de operações como criação de conversas, envio e receção de mensagens de texto, eliminação local e remota, e gestão de anexos (imagens, documentos e áudio).

A investigação revelou múltiplas vulnerabilidades com relevância forense: a chave de encriptação da base de dados está armazenada em texto claro no ficheiro `config.json`; o conteúdo das mensagens encontra-se em texto claro após desencriptação, sem proteção adicional; a eliminação de mensagens não remove o conteúdo do índice *full-text* (`messages_fts`); os ficheiros de anexos persistem na pasta `attachments.noindex` após eliminação das mensagens associadas; e a eliminação remota pelo remetente não remove os dados no dispositivo do destinatário, apenas substitui o conteúdo por um *placeholder*.

Os resultados demonstram que, apesar da encriptação de ponta a ponta proteger as comunicações em trânsito, existem múltiplas oportunidades de recuperação forense para investigadores com acesso físico ao dispositivo.

Palavras-chave: Análise Forense Digital, Session, SQLCipher, SQLite, Encriptação, Privacidade, Recuperação de Dados

Abstract

The increasing use of privacy-focused messaging applications presents new challenges for digital forensic analysis. This work conducts a behavioral forensic analysis of the Session Desktop application, a decentralized messaging platform that uses end-to-end encryption, does not require a phone number for registration, and stores data locally with SQLCipher encryption (AES-256).

Through a systematic methodology comprising nine test scenarios, the SQLite database structure was mapped — identifying 19 tables, 25 indexes, and 3 triggers — and the behavior resulting from operations such as conversation creation, sending and receiving text messages, local and remote deletion, and attachment management (images, documents, and audio) was analyzed.

The investigation revealed multiple vulnerabilities with forensic relevance: the database encryption key is stored in plain text in the `config.json` file; message content is stored in plain text after decryption, without additional protection; message deletion does not remove content from the full-text index (`messages_fts`); attachment files persist in the `attachments.noindex` folder after message deletion; and remote deletion by the sender does not remove data on the recipient's device, only replacing the content with a placeholder.

The results demonstrate that, although end-to-end encryption protects communications in transit, there are multiple forensic recovery opportunities for investigators with physical access to the device.

Keywords: Digital Forensics, Session, SQLCipher, SQLite, Encryption, Privacy, Data Recovery

Declaração sobre o uso de Inteligência Artificial

Na elaboração deste relatório foi utilizada Inteligência Artificial generativa. A ferramenta Claude (Anthropic) foi utilizada para apoio na estruturação e revisão do texto, elaboração de queries SQL para análise da base de dados, esclarecimento de dúvidas técnicas sobre SQLite e SQLCipher, formatação LaTeX e organização da documentação. Toda a execução prática — incluindo instalação e configuração do ambiente de testes, realização dos cenários de teste, captura de evidências e interpretação dos resultados forenses — foi realizada integralmente pelo autor. O autor assume total responsabilidade pelo conteúdo final do trabalho.

Capítulo

Índice

Capítulo

Lista de Figuras

Capítulo

Lista de Tabelas

Capítulo

Listagens de Código

Capítulo

Siglas & Acrónimos

AES Advanced Encryption Standard.

API Application Programming Interface.

BD Base de Dados.

CBC Cipher Block Chaining.

E2E End-to-End Encryption.

FTS Full-Text Search.

GUI Graphical User Interface.

HMAC Hash-based Message Authentication Code.

ID Identifier.

IP Internet Protocol.

JSON JavaScript Object Notation.

KDF Key Derivation Function.

MAC Message Authentication Code.

P2P Peer-to-Peer.

PBKDF2 Password-Based Key Derivation Function 2.

SHA Secure Hash Algorithm.

SO Sistema Operativo.

SQL Structured Query Language.

Capítulo

Glossário

análise comportamental Tipo de análise forense focada em compreender como uma aplicação reage a diferentes operações, documentando que tabelas e registos são afetados por cada ação do utilizador.

análise forense digital Processo de identificação, preservação, análise e documentação de evidências digitais armazenadas em dispositivos eletrónicos, seguindo metodologias que garantam a integridade e admissibilidade das provas.

AppData Pasta oculta no Windows onde as aplicações armazenam dados de configuração e ficheiros específicos do utilizador. Localizada em `C:\Users\[utilizador]\AppData`.

BLOB Binary Large Object – tipo de dados em bases de dados utilizado para armazenar dados binários como imagens, ficheiros ou conteúdo encriptado.

encriptação ponta-a-ponta Método de comunicação segura onde apenas os participantes da conversa conseguem decifrar as mensagens. Os servidores intermediários transportam dados encriptados sem acesso ao conteúdo.

freelist Lista de páginas da base de dados SQLite que foram libertadas após eliminação de dados mas que ainda podem conter informação residual. Relevante para recuperação forense de dados eliminados.

hash Valor de tamanho fixo gerado por uma função criptográfica a partir de dados de tamanho arbitrário. Utilizado para verificar integridade de dados e identificar ficheiros de forma única.

metadados Dados que descrevem outros dados. No contexto de mensagens, inclui informação como remetente, destinatário, hora de envio e tamanho, mas não o conteúdo da mensagem.

onion routing Técnica de comunicação anónima onde as mensagens são encapsuladas em múltiplas camadas de encriptação e transmitidas através de uma série de nós intermediários, impedindo que qualquer nó conheça simultaneamente a origem e o destino.

PRAGMA Comando SQL especial do SQLite utilizado para modificar o funcionamento da biblioteca ou consultar informações internas sobre a base de dados.

raw key Chave de encriptação em formato hexadecimal bruto, utilizada diretamente pelo SQLCipher para desencriptar a base de dados, em vez de uma password que necessita derivação.

Service Node Nó da rede descentralizada do Session responsável por armazenar mensagens offline e providenciar funcionalidades de onion routing para ocultar endereços IP dos utilizadores.

Session Protocol Protocolo de encriptação desenvolvido especificamente para o Session, otimizado para redes descentralizadas, fornecendo encriptação sem necessidade de sincronização constante entre dispositivos.

SQLCipher Extensão do SQLite desenvolvida pela Zetetic LLC que fornece encriptação transparente AES de 256 bits para bases de dados SQLite. Utiliza CBC mode, HMAC-SHA512 para autenticação e PBKDF2 para derivação de chaves.

SQLite Sistema de gestão de bases de dados relacional, leve e autónomo, que armazena toda a base de dados num único ficheiro. Amplamente utilizado em aplicações móveis e desktop.

timestamp Valor numérico que representa um momento específico no tempo, geralmente expresso como o número de milissegundos desde 1 de janeiro de 1970 (Unix epoch).

trigger Procedimento armazenado numa base de dados que é executado automaticamente em resposta a determinados eventos, como inserção, atualização ou eliminação de registos numa tabela.

Write-Ahead Log Mecanismo de journaling do SQLite onde as alterações são primeiro escritas num ficheiro de log antes de serem aplicadas à base de dados principal. Pode conter transações não confirmadas úteis para análise forense.

XChaCha20-Poly1305 Algoritmo de encriptação autenticada utilizado pelo Session para encriptar anexos locais. Combina a cifra de fluxo XChaCha20 com o MAC Poly1305.

1. Introdução

A proliferação de aplicações de mensagens instantâneas transformou radicalmente a forma como comunicamos. Estas plataformas, utilizadas por milhares de milhões de pessoas diariamente, armazenam localmente volumes significativos de dados pessoais, incluindo conversas, ficheiros multimédia e metadados de comunicação. Esta realidade confere às aplicações de mensagens um papel central em investigações forenses digitais, onde a capacidade de recuperar e interpretar estes dados pode ser determinante.

O **Session** é uma aplicação de mensagens multiplataforma que se distingue pelo seu foco na privacidade e anonimato do utilizador (Session Technology Foundation, 2024). Desenvolvida pela Session Technology Foundation como uma derivação (*fork*) do Signal, a aplicação implementa encriptação ponta-a-ponta e dispensa a necessidade de número de telefone ou email para registo, utilizando identificadores alfanuméricos aleatórios (Jefferys et al., 2024). A comunicação é realizada através de uma rede descentralizada de nós que implementa *onion routing*, ocultando os endereços Internet Protocol (IP) dos utilizadores e dificultando a recolha de metadados.

Do ponto de vista forense, o Session apresenta características particularmente interessantes. A aplicação armazena todos os dados localmente numa base de dados SQLite encriptada com SQLCipher, uma extensão que fornece encriptação Advanced Encryption Standard (AES) de 256 bits de forma transparente (Zetetic LLC, 2024). Esta camada de proteção, embora represente um desafio para a análise, não impossibilita o acesso aos dados quando a chave de encriptação é conhecida, cenário comum em investigações com acesso físico ao dispositivo.

O presente trabalho realiza uma análise forense comportamental ao Session Desktop, documentando sistematicamente como a Base de Dados (BD) reage às operações típicas de utilização da aplicação.

1.1 Motivação

A crescente adoção de aplicações de mensagens encriptadas coloca novos desafios às equipas de investigação forense. Enquanto plataformas como WhatsApp, Telegram ou Discord são frequentemente analisadas e documentadas na literatura forense, aplicações focadas em privacidade como o Session permanecem comparativamente menos estudadas.

O Session, em particular, representa um caso de estudo relevante por várias razões:

- **Arquitetura descentralizada** – Ao contrário da maioria das aplicações de mensagens, o Session não depende de servidores centrais, utilizando uma rede de nós distribuídos para transmissão e armazenamento temporário de mensagens;
- **Anonimato por design** – A ausência de requisitos de identificação pessoal (telefone, email) e a implementação de *onion routing* tornam a aplicação atrativa para utilizadores que valorizam a privacidade, mas também potencialmente para atividades ilícitas;
- **Herança do Signal** – Sendo um *fork* do Signal, o Session partilha algumas estruturas de dados, mas implementa o seu próprio protocolo de encriptação, criando uma combinação única de artefactos forenses;
- **Encriptação local** – A utilização de SQLCipher para proteger a BD local representa um desafio técnico que exige metodologias específicas de acesso.

Compreender o comportamento interno do Session – que tabelas são utilizadas, como os dados são estruturados e o que acontece quando mensagens são eliminadas – é essencial para investigadores forenses que possam encontrar esta aplicação em dispositivos sob análise.

1.2 Objetivos

O presente trabalho tem como objetivo principal realizar uma **análise forense comportamental** à aplicação Session Desktop, documentando de forma sistemática o funcionamento interno da sua base de dados e identificando vulnerabilidades de segurança no armazenamento local de dados.

1.2.1 Objetivo Geral

Investigar e documentar como a base de dados do Session Desktop responde a diferentes operações de utilização, identificando os artefactos forenses gerados, avaliando a persistência de dados após eliminação, e documentando vulnerabilidades que possam ser exploradas em contexto de investigação criminal.

1.2.2 Objetivos Específicos

Para a concretização do objetivo geral, foram definidos os seguintes objetivos específicos:

1. **Localizar e aceder aos dados** — Identificar a localização dos ficheiros de dados do Session Desktop no sistema de ficheiros Windows e documentar o processo de desencriptação da base de dados SQLCipher;
2. **Mapear a estrutura da base de dados** — Documentar as tabelas, índices e *triggers* existentes na base de dados SQLite, identificando a função de cada componente no armazenamento de mensagens e metadados;
3. **Analisar o ciclo de vida das mensagens** — Verificar as alterações na BD durante o envio e receção de mensagens, identificando os campos populados, os valores registados e as diferenças entre mensagens `incoming` e `outgoing`;
4. **Investigar mecanismos de eliminação** — Avaliar se as mensagens eliminadas são efetivamente removidas da BD ou se permanecem vestígios recuperáveis, comparando o comportamento das opções "Limpar para mim" e "Limpar para todos";
5. **Analisar a eliminação remota** — Documentar o comportamento da base de dados do destinatário quando o remetente elimina uma mensagem "para todos";
6. **Examinar o armazenamento de anexos** — Analisar como ficheiros enviados e recebidos (imagens, documentos, áudio) são armazenados localmente e a sua relação com a base de dados principal;
7. **Identificar vulnerabilidades de segurança** — Documentar falhas no modelo de armazenamento local que possam comprometer a privacidade dos utilizadores ou facilitar a recuperação forense de dados.

Nota Metodológica

Os objetivos foram definidos seguindo uma abordagem incremental: partindo da localização e acesso aos dados (objetivos 1–2), progredindo para a análise comportamental (objetivos 3–6), e culminando na identificação de vulnerabilidades (objetivo 7). Esta estrutura permite uma compreensão progressiva do sistema em análise.

1.3 Estrutura do Relatório

O presente relatório está organizado em quatro capítulos, estruturados da seguinte forma:

Capítulo 1 — Introdução: Apresenta o contexto do trabalho, as motivações para a escolha do Session Desktop como objeto de estudo, os objetivos definidos e a estrutura do documento.

Capítulo 2 — Metodologia: Descreve o ambiente de testes utilizado, as ferramentas empregues na análise, os procedimentos de acesso à base de dados encriptada e a metodologia dos cenários de teste.

Capítulo 3 — Análise Forense: Constitui o núcleo do trabalho, apresentando a estrutura da base de dados do Session Desktop e documentando os resultados dos nove cenários de teste realizados, incluindo criação de conversas, envio e receção de mensagens, eliminação local e remota, e gestão de anexos.

Capítulo 4 — Conclusão: Sintetiza as principais descobertas, avalia o cumprimento dos objetivos, discute os contributos e limitações do trabalho, e apresenta sugestões para investigação futura.

O relatório inclui ainda anexos com:

- Estrutura detalhada das tabelas da base de dados;
- Capturas de ecrã das queries executadas em cada cenário;
- Código Structured Query Language (SQL) dos *triggers* analisados.

2. Metodologia

Este capítulo descreve o ambiente de testes utilizado, as ferramentas empregues na análise e os procedimentos metodológicos seguidos para a realização da análise forense comportamental ao Session Desktop.

2.1 Ambiente de Testes

A análise forense foi realizada num ambiente controlado, utilizando dois dispositivos para simular comunicações reais entre utilizadores. Para complementar a análise e simular cenários de comunicação bidireccional, foi utilizado um dispositivo móvel como segundo interlocutor.

2.1.1 Contas de Teste

Para a realização dos cenários de teste, foram criadas duas contas Session distintas, permitindo simular conversas reais entre utilizadores diferentes:

- **Spider** — Conta principal, utilizada no Session Desktop (Windows), sobre a qual foi realizada a análise forense da base de dados;
- **Elliot** — Conta secundária, utilizada no Session Android, servindo como interlocutor para envio e receção de mensagens.

Cada conta foi identificada por um *Account ID* único — uma cadeia alfanumérica de 66 caracteres gerada automaticamente pela aplicação através de criptografia de curva elíptica. Este identificador substitui o número de telefone utilizado por outras aplicações de mensagens, contribuindo para o anonimato dos utilizadores.

2.2 Ferramentas Utilizadas

A análise forense foi conduzida utilizando as seguintes ferramentas:

2.2.1 Session Desktop

Aplicação de mensagens objeto de estudo, instalada a partir do website oficial (Session Technology Foundation, 2024). A instalação padrão no Windows armazena os dados do utilizador na pasta `%AppData%\Roaming\Session\`.

O Session utiliza encriptação End-to-End Encryption (E2E) baseada no protocolo Signal e armazena os dados localmente numa base de dados SQLite encriptada com SQLCipher.

2.2.2 DB Browser for SQLite (SQLCipher)

Ferramenta gráfica de código aberto para visualização e manipulação de bases de dados SQLite (DB Browser for SQLite Development Team, 2024). Foi utilizada especificamente a versão compilada com suporte a SQLCipher, que permite abrir bases de dados encriptadas mediante fornecimento da chave de desencriptação.

A escolha desta ferramenta justifica-se pela sua capacidade de:

- Abrir bases de dados encriptadas com SQLCipher 4;
- Executar queries SQL diretamente sobre os dados;
- Visualizar a estrutura de tabelas, índices e *triggers*;
- Exportar resultados para análise posterior.

2.2.3 Explorador de Ficheiros do Windows

Utilizado para navegação na estrutura de pastas da aplicação, identificação de ficheiros relevantes e monitorização de alterações no sistema de ficheiros durante os testes — nomeadamente na pasta `attachments.noindex` onde são armazenados os anexos.

2.3 Localização dos Dados

O Session Desktop armazena todos os dados do utilizador na pasta AppData do Windows. A estrutura de ficheiros relevantes para análise forense encontra-se representada na Tabela 2.1.

Tabela 2.1: Estrutura de ficheiros do Session Desktop

Ficheiro/Pasta	Descrição
`config.json`	Ficheiro de configuração contendo a chave de encriptação da base de dados em formato hexadecimal
`sql/db.sqlite`	Base de dados principal encriptada com SQLCipher, contendo mensagens, conversas e metadados
`attachments.noindex/`	Pasta contendo ficheiros anexados às mensagens, encriptados com XChaCha20-Poly1305
`blob_storage/`	Armazenamento de objetos binários
`logs/`	Ficheiros de registo da aplicação
`Session Storage/`	Dados de sessão da aplicação

O caminho completo para acesso aos dados é:

```
C:\Users\[utilizador]\AppData\Roaming\Session\
```

2.4 Acesso à Base de Dados Encriptada

A base de dados do Session Desktop está encriptada com SQLCipher versão 4, utilizando o algoritmo AES-256 em modo Cipher Block Chaining (CBC). Esta encriptação impede a abertura direta com ferramentas SQLite convencionais, requerendo a obtenção da chave de encriptação e a utilização de uma ferramenta compatível.

2.4.1 Obtenção da Chave de Encriptação

A chave de encriptação encontra-se armazenada no ficheiro `config.json`, localizado na raiz da pasta de dados da aplicação. O ficheiro contém um objeto JavaScript Object Notation (JSON) com a seguinte estrutura simplificada:

```
{
  "key": "3c2e7de3...f93624",
  "opengroupPruning": true
}
```

Listagem 2.1: Estrutura do ficheiro config.json

O campo `key` contém uma *string* hexadecimal de 64 caracteres, correspondendo a uma chave AES-256 (256 bits = 32 bytes = 64 caracteres hexadecimais).

Caso o utilizador tenha definido uma *password* de proteção na aplicação Session, essa *password* substitui a chave armazenada no ficheiro de configuração como método de desencriptação. Neste cenário, a análise forense requer o conhecimento da *password* ou a utilização de técnicas de recuperação.

Vulnerabilidade de Segurança

O armazenamento da chave de encriptação em texto claro no ficheiro `config.json` representa uma vulnerabilidade significativa. Qualquer utilizador ou processo com acesso de leitura ao sistema de ficheiros pode obter a chave e desencriptar toda a base de dados, anulando a proteção oferecida pelo SQLCipher.

2.4.2 Processo de Abertura

Para aceder à base de dados encriptada, foi seguido o procedimento documentado abaixo:

1. **Encerrar a aplicação** — Fechar completamente o Session Desktop para libertar o *lock* sobre o ficheiro da base de dados;
2. **Abrir o DB Browser** — Executar o DB Browser for SQLite na versão compilada com suporte a SQLCipher;
3. **Selecionar a base de dados** — Aceder a `File` → `Open Database` e navegar até ao ficheiro `sql/db.sqlite`;
4. **Configurar a encriptação** — Na janela de configuração:
   - Selecionar **SQLCipher 4 defaults** nas definições;
   - Alterar o tipo de chave de *Password* para **Raw key**;
   - Inserir a chave precedida do prefixo `0x`:

   ```
   0x[chave_do_config.json]
   ```

5. **Confirmar abertura** — Validar a configuração e aceder à base de dados desencriptada.

A Figura 2.1 ilustra a configuração correta para abertura da base de dados no DB Browser for SQLite.

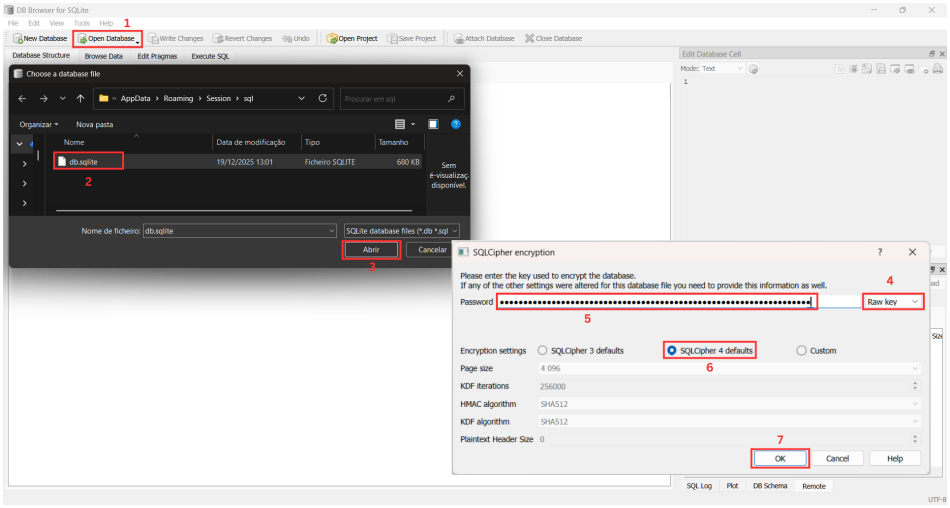


Figura 2.1: Configuração do DB Browser para abertura da base de dados encriptada

2.5 Procedimentos de Teste

A análise comportamental foi estruturada em cenários de teste, cada um focado numa operação específica da aplicação. Para cada cenário, foi adotada a seguinte metodologia:

1. **Documentar estado inicial** — Registar o número de registos em cada tabela relevante antes da operação;
2. **Executar operação** — Realizar a ação pretendida na aplicação Session Desktop;
3. **Fechar aplicação** — Encerrar o Session para garantir que todas as alterações foram persistidas na base de dados;
4. **Documentar estado final** — Registar as alterações observadas nas tabelas, identificando novos registos, campos modificados ou dados eliminados;
5. **Analisar diferenças** — Comparar os estados inicial e final, documentando as conclusões forenses relevantes.

2.5.1 Cenários de Teste Definidos

Foram definidos os cenários de teste apresentados na Tabela 2.2, abrangendo as operações principais da aplicação.

2.5.2 Queries de Monitorização

Para documentar o estado da BD em cada cenário, foram utilizadas as seguintes queries SQL de forma sistemática:

```sql
SELECT 'conversations' AS tabela, COUNT(*) AS total FROM conversations
UNION ALL SELECT 'messages', COUNT(*) FROM messages
UNION ALL SELECT 'seenMessages', COUNT(*) FROM seenMessages
UNION ALL SELECT 'attachment_downloads', COUNT(*) FROM attachment_downloads;
```

Listagem 2.2: Query para contagem de registos por tabela

Tabela 2.2: Cenários de teste realizados

ID	Cenário	Descrição
C0	Estado Inicial	Documentar estrutura e conteúdo da BD numa conta recém-criada
C1	Criação de Conversa	Iniciar uma nova conversa com envio de pedido de mensagem
C2	Receção de Mensagem	Receber uma mensagem de texto e comparar com o cenário de envio
C3a	Eliminação Local (para mim)	Apagar mensagem com opção "Limpar para mim"
C3b	Eliminação Local (para todos)	Apagar mensagem com opção "Limpar para todos"
C4	Envio de Anexo	Enviar uma imagem e analisar o armazenamento local
C5	Receção de Anexo	Receber imagem e documento, verificar comportamento
C6a	Receção de Áudio	Receber mensagem de áudio e analisar armazenamento
C6b	Eliminação Remota	Remetente apaga mensagem "para todos" e verifica-se impacto no destinatário

```sql
SELECT
    id, conversationId, type, body,
    hasAttachments, hasFileAttachments, hasVisualMediaAttachments,
    datetime(sent_at/1000, 'unixepoch', 'localtime') AS data_hora
FROM messages
ORDER BY sent_at DESC;
```

Listagem 2.3: Query para visualização de mensagens

```sql
SELECT rowid, body FROM messages_fts;
```

Listagem 2.4: Query para análise do índice full-text

```sql
SELECT name, sql FROM sqlite_master WHERE type = 'trigger';
```

Listagem 2.5: Query para análise de triggers

2.6 Considerações Éticas

Todos os testes foram realizados em contas Session criadas especificamente para este trabalho, não tendo sido analisados dados de terceiros. As chaves de encriptação e identificadores de conta apresentados neste relatório são fictícios ou foram alterados por razões de segurança.

A metodologia descrita destina-se exclusivamente a fins educativos e de investigação forense legítima, devendo qualquer aplicação prática respeitar o enquadramento legal aplicável e os direitos de privacidade dos indivíduos.

3. Análise Forense

Este capítulo apresenta os resultados da análise forense comportamental realizada à aplicação Session Desktop. A primeira secção documenta a estrutura da base de dados, incluindo tabelas, índices, *triggers* e o sistema de armazenamento de anexos. A segunda secção detalha os cenários de teste executados e as respetivas descobertas forenses.

3.1 Estrutura da Base de Dados

A base de dados do Session Desktop está armazenada no ficheiro `db.sqlite`, localizado em `%AppData%\Roaming\Session\sql\`. Trata-se de uma base de dados SQLite encriptada com SQLCipher, cuja estrutura está resumida na Tabela 3.1.

Tabela 3.1: Resumo da estrutura da base de dados

Tipo de Objeto	Quantidade
Tabelas	19
Índices	25
Triggers	3

3.1.1 Tabelas Principais

Das 19 tabelas identificadas (listagem completa no Anexo A.1), destacam-se as seguintes pela sua relevância forense:

- `messages` — Armazena todas as mensagens enviadas e recebidas, incluindo conteúdo, *timestamps* e estado;
- `conversations` — Regista as conversas e contactos, com identificadores, nomes e *timestamps* de atividade;
- `seenMessages` — Controla as mensagens marcadas como visualizadas;
- `attachment_downloads` — Gere o estado de download de ficheiros anexados;
- `messages_fts` — Índice Full-Text Search (FTS) que permite pesquisa por palavras-chave no conteúdo das mensagens.

Destaca-se que seis tabelas (`messages_fts*`) estão dedicadas à funcionalidade de pesquisa FTS, replicando o conteúdo das mensagens para pesquisa otimizada.

3.1.2 Campos Relevantes para Análise Forense

Na tabela `messages`, os campos com maior relevância forense são:

- `body` — Conteúdo da mensagem em texto claro;
- `type` — Distingue mensagens enviadas (`outgoing`) de recebidas (`incoming`);
- `sent_at` e `received_at` — *Timestamps* em milissegundos Unix;
- `conversationId` — Associa a mensagem a uma conversa;
- `hasAttachments`, `hasFileAttachments`, `hasVisualMediaAttachments` — Indicam presença e tipo de anexos.

Na tabela `conversations`, destacam-se:

- `id` — *Account ID* do contacto (66 caracteres);
- `displayNameInProfile` — Nome público do contacto;
- `isApproved` e `didApproveMe` — Estado do pedido de mensagem;
- `active_at` — *Timestamp* da última atividade.

A estrutura completa das tabelas encontra-se nos Anexos A.3 e A.4.

3.1.3 Armazenamento de Anexos

Os ficheiros anexados às mensagens não são armazenados na base de dados SQLite, sendo guardados numa pasta dedicada:

```
%AppData%\Roaming\Session\attachments.noindex\
```

Encriptação dos Anexos

Os anexos são encriptados individualmente com o algoritmo **XChaCha20-Poly1305** (Jefferys et al., 2024). A chave de encriptação está armazenada na tabela `items`, no registo `local_attachment_encrypted_key`.

Organização no Sistema de Ficheiros

Os ficheiros são organizados em subpastas baseadas nos primeiros caracteres do identificador (ex: `0d/`, `5e/`, `35/`). Os nomes são identificadores alfanuméricos gerados pela aplicação, não preservando o nome ou extensão original.

Relação com a Base de Dados

A tabela `messages` indica a presença de anexos através dos campos booleanos apresentados na Tabela 3.2.

3.1.4 Análise dos Triggers

Foram identificados três *triggers* na base de dados, todos associados à manutenção do índice FTS.

Tabela 3.2: Campos de identificação de tipos de anexo

Tipo de Anexo	hasAttachments	hasFileAttachments	hasVisualMedia
Imagem/Vídeo	1	0	1
Documento	1	1	0
Áudio	1	0	0

Trigger: messages_on_insert

Executado após inserção de nova mensagem:

```
CREATE TRIGGER messages_on_insert AFTER INSERT ON messages
BEGIN
    INSERT INTO messages_fts (rowid, body) VALUES (new.rowid, new.body);
END
```

Listagem 3.1: Trigger messages_on_insert

Copia o conteúdo do campo `body` para o índice FTS, permitindo pesquisas rápidas.

Trigger: messages_on_delete

Executado após eliminação de mensagem:

```
CREATE TRIGGER messages_on_delete AFTER DELETE ON messages
BEGIN
    DELETE FROM messages_fts WHERE rowid = old.rowid;
END
```

Listagem 3.2: Trigger messages_on_delete

Remove o registo correspondente do índice FTS. A existência deste *trigger* indica que a eliminação de mensagens resulta em **remoção física** (DELETE), não marcação lógica.

Trigger: messages_on_update

Executado quando o conteúdo de uma mensagem é alterado:

```
CREATE TRIGGER messages_on_update AFTER UPDATE ON messages
WHEN new.body <> old.body
BEGIN
    DELETE FROM messages_fts WHERE rowid = old.rowid;
    INSERT INTO messages_fts(rowid, body) VALUES (new.rowid, new.body);
END
```

Listagem 3.3: Trigger messages_on_update

Apenas é acionado quando o campo `body` é modificado, atualizando o índice FTS com o novo conteúdo.

Conclusões sobre os Triggers

A análise permite concluir:

1. O Session mantém um índice FTS sincronizado automaticamente com as mensagens;
2. A eliminação resulta em remoção física dos registos;
3. Conteúdo eliminado pode persistir no *freelist* do SQLite ou no ficheiro Write-Ahead Log, sendo potencialmente recuperável;
4. O histórico de edições não é preservado.

3.2 Cenários de Teste

Esta secção documenta os resultados dos nove cenários de teste realizados, apresentando as alterações observadas na base de dados para cada operação e as respetivas conclusões forenses.

3.2.1 Cenário 0: Estado Inicial

Antes de realizar os testes comportamentais, foi documentado o estado inicial da base de dados de uma instalação limpa do Session Desktop. A conta de teste (identificada como "Spider") foi criada especificamente para este trabalho, sem qualquer histórico de conversas ou mensagens prévias.

Contagem de Registos

A Tabela 3.3 apresenta a contagem de registos nas principais tabelas no estado inicial.

Tabela 3.3: Cenário 0: Estado inicial da base de dados

Tabela	Registos
`conversations`	1
`messages`	0
`seenMessages`	0
`attachment_downloads`	0

Análise do Registo Inicial

O único registo na tabela `conversations` corresponde à própria conta do utilizador, criada automaticamente durante a configuração inicial. A Tabela 3.4 apresenta os campos relevantes.

Tabela 3.4: Registo inicial na tabela `conversations`

Campo	Valor
`id`	`0568810de27d370009a8d3d26cb94d05...`
`displayNameInProfile`	Spider
`type`	private
`isApproved`	1
`didApproveMe`	1
`active_at`	NULL

A Figura 3.1 apresenta a execução da query de contagem no DB Browser for SQLite.

SQL 1*

```
1  SELECT 'conversations' AS tabela, COUNT(*) AS total FROM conversations
2  UNION ALL
3  SELECT 'messages', COUNT(*) FROM messages
4  UNION ALL
5  SELECT 'seenMessages', COUNT(*) FROM seenMessages
6  UNION ALL
7  SELECT 'attachment_downloads', COUNT(*) FROM attachment_downloads;
```

	tabela	total
1	conversations	1
2	messages	0
3	seenMessages	0
4	attachment_downloads	0

Figura 3.1: Query de contagem de registos no estado inicial

Observações

O estado inicial revela as seguintes características:

1. A aplicação cria automaticamente um registo na tabela `conversations` para a própria conta do utilizador;
2. Os campos `isApproved` e `didApproveMe` estão definidos como `1` para a conta própria, indicando auto-aprovação;
3. O campo `active_at` com valor `NULL` resulta na data `1970-01-01` quando convertido (Unix Epoch);
4. As tabelas `messages` e `seenMessages` vazias confirmam a inexistência de histórico de comunicações.

Este estado serve como *baseline* para comparação com os cenários subsequentes.

3.2.2 Cenário 1: Criação de Nova Conversa

Este cenário documenta o processo de estabelecimento de uma nova conversa entre dois utilizadores Session, analisando as alterações na base de dados resultantes desta operação.

Procedimento

O Session implementa um mecanismo de *pedido de mensagem* para proteção contra contactos não solicitados. O processo envolveu:

1. O utilizador "Spider" abriu o menu "Nova Mensagem" e inseriu o *Account ID* do utilizador "Elliot";
2. Foi enviada uma mensagem inicial: "Olá Elliot, teste de conversa";
3. O utilizador "Elliot" recebeu a notificação de pedido de mensagem e aceitou;
4. A conversa ficou estabelecida para ambos os utilizadores.

A Figura 3.2 ilustra a interface do Session Desktop após a conclusão do processo.

Alterações na Base de Dados

A Tabela 3.5 apresenta a comparação entre o estado inicial e o estado após a criação da conversa.

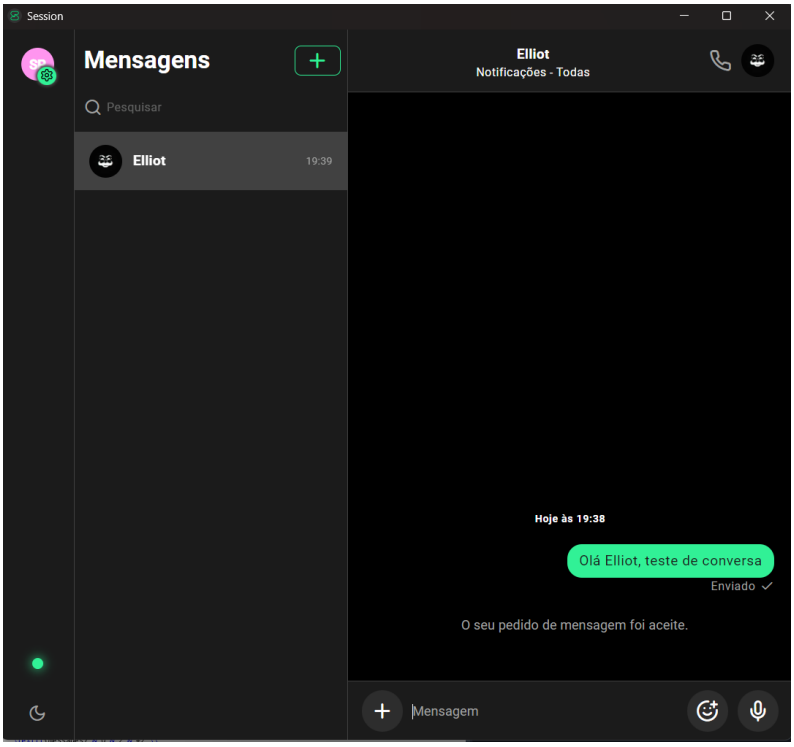


Figura 3.2: Interface do Session Desktop após criação da conversa

Tabela 3.5: Cenário 1: Comparação de registos

Tabela	Antes	Depois	Diferença
conversations	1	2	+1
messages	0	2	+2
seenMessages	0	4	+4
attachment_downloads	0	0	0

Tabela `conversations` Foi criado um novo registo correspondente ao contacto "Elliot", conforme apresentado na Tabela 3.6.

Tabela 3.6: Novo registo na tabela `conversations`

Campo	Valor
`id`	`05214644dadfe36ac7a938a8c9e83efddc36...`
`displayNameInProfile`	Elliot
`type`	private
`isApproved`	1
`didApproveMe`	1
`active_at`	2025-12-20 19:39:20

Os campos `isApproved` e `didApproveMe` com valor `1` confirmam a aprovação mútua entre os utilizadores.

Tabela `messages` Foram criados dois registos, apresentados na Tabela 3.7.

Tabela 3.7: Registos criados na tabela `messages`

Tipo	Body	Descrição	Timestamp
`outgoing`	"Olá Elliot, teste..."	Mensagem enviada	19:38:37
`incoming`	NULL	Mensagem de controlo	19:39:20

As queries completas encontram-se no Anexo B.2.

Observações

A análise deste cenário permite concluir:

1. A criação de uma conversa gera um novo registo na tabela `conversations` com o *Account ID* do contacto como chave primária;
2. O campo `active_at` é atualizado com o *timestamp* da última atividade;
3. O processo de pedido de mensagem gera uma mensagem de controlo (`type=incoming`, `body=NULL`) que representa a aceitação do pedido;
4. A tabela `seenMessages` regista múltiplas entradas para gestão do estado de leitura;
5. Uma simples troca de mensagens gera múltiplos artefactos forenses que permitem reconstruir a cronologia da comunicação.

3.2.3 Cenário 2: Receção de Mensagem

Este cenário documenta a receção de uma mensagem enviada por outro utilizador, analisando como a base de dados regista mensagens recebidas em comparação com mensagens enviadas.

Procedimento

Após o estabelecimento da conversa no cenário anterior, o utilizador "Elliot" enviou uma resposta através da aplicação Session Mobile:

1. O utilizador "Elliot" abriu a conversa com "Spider" no dispositivo móvel;
2. Foi enviada a mensagem: "Olá Spider, mensagem recebida e pedido aceite";
3. O utilizador "Spider" recebeu a mensagem no Session Desktop;
4. A base de dados foi analisada após encerramento da aplicação.

A Figura 3.3 ilustra a conversa após a troca de mensagens.

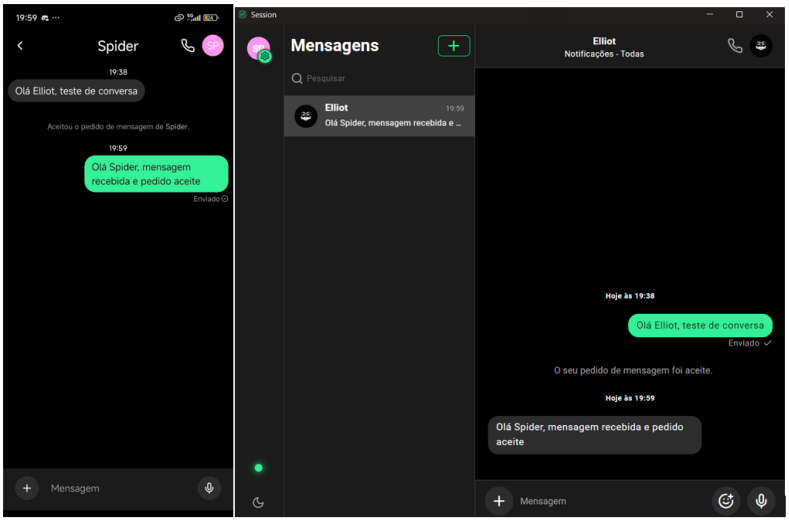


Figura 3.3: Conversa entre Spider (Desktop) e Elliot (Mobile)

Alterações na Base de Dados

A Tabela 3.8 apresenta a comparação entre o estado após o Cenário 1 e o estado após a receção da mensagem.

Tabela 3.8: Cenário 2: Comparação de registos

Tabela	C1	C2	Diferença
`conversations`	2	2	0
`messages`	2	3	+1
`seenMessages`	4	5	+1
`attachment_downloads`	0	0	0

Tabela `conversations` Não foi criado nenhum novo registo, uma vez que a conversa já existia. O campo `active_at` foi atualizado para o *timestamp* da última mensagem recebida (19:59:45).

Tabela `messages` Foi criado um novo registo correspondente à mensagem recebida. A Tabela 3.9 apresenta o estado da tabela após este cenário.

Tabela 3.9: Estado da tabela `messages` após receção

Tipo	Body	Descrição	Hora
`incoming`	"Olá Spider, mensagem..."	Mensagem recebida	19:59:45
`incoming`	NULL	Mensagem de controlo	19:39:20
`outgoing`	"Olá Elliot, teste..."	Mensagem enviada	19:38:37

Comparação: Mensagens Enviadas vs Recebidas

A Tabela 3.10 resume as diferenças entre mensagens enviadas e recebidas.

Tabela 3.10: Comparação entre tipos de mensagem

Campo	Enviada	Recebida
`type`	outgoing	incoming
`source`	*Account ID* próprio	*Account ID* do remetente
`sent_at`	*Timestamp* de envio	*Timestamp* original
`received_at`	*Timestamp* local	*Timestamp* de receção

As queries completas encontram-se no Anexo B.3.

Observações

A análise deste cenário permite concluir:

1. A receção de mensagem cria um novo registo com `type=incoming`;
2. O campo `active_at` na tabela `conversations` é atualizado com o *timestamp* da última mensagem;
3. O conteúdo da mensagem (`body`) é armazenado em texto claro, permitindo leitura direta após desencriptação da BD;
4. É possível distinguir a direção da comunicação através do campo `type`, permitindo reconstruir o fluxo da conversa;
5. A tabela `seenMessages` regista a visualização, podendo determinar se o utilizador viu a mensagem.

3.2.4 Cenário 3: Eliminação de Mensagem

Este cenário documenta o comportamento da base de dados quando uma mensagem é eliminada pelo utilizador. O Session Desktop oferece duas opções de eliminação: "Limpar para mim" (eliminação local) e "Limpar para todos" (eliminação global). Ambas foram testadas para comparar o seu impacto nos artefactos forenses.

Cenário 3a: Eliminação Local ("Limpar para mim")

Esta opção remove a mensagem apenas do dispositivo local, mantendo-a visível para os outros participantes da conversa.

Procedimento

1. Foi enviada uma nova mensagem de teste: "Mensagem teste eliminação local";
2. O estado da base de dados foi documentado antes da eliminação;
3. A mensagem foi eliminada utilizando a opção "Limpar para mim";
4. O estado da base de dados foi documentado após a eliminação.

Estado Antes da Eliminação A Tabela 3.11 apresenta o estado da base de dados antes da eliminação.

Tabela 3.11: Cenário 3a: Estado antes da eliminação local

Tabela	Registos
`conversations`	2
`messages`	4
`seenMessages`	7
`attachment_downloads`	0

Estado Após a Eliminação A Tabela 3.12 apresenta a comparação entre os estados antes e depois da eliminação.

Tabela 3.12: Cenário 3a: Comparação de registos

Tabela	Antes	Depois	Diferença
`conversations`	2	2	0
`messages`	4	3	-1
`seenMessages`	7	8	$+1$
`attachment_downloads`	0	0	0

Análise da Tabela `messages` A mensagem foi **fisicamente removida** da tabela `messages`. A eliminação não utilizou marcação lógica (campo `isDeleted`), mas sim uma operação `DELETE` que removeu o registo.

Análise do Índice Full-Text Apesar da remoção do registo da tabela principal, a análise da tabela `messages_fts` revelou uma descoberta significativa: **o conteúdo da mensagem eliminada permanece acessível** no índice FTS.

A Tabela 3.13 ilustra os registos encontrados após a eliminação.

Tabela 3.13: Excerto da tabela `messages_fts` após eliminação

rowid	body
14	Mensagem teste eliminação local
15	Mensagem teste eliminação local
...	...
20	Mensagem teste eliminação local

Descoberta Forense Crítica

O conteúdo de mensagens eliminadas **persiste na tabela** `messages_fts`, permitindo a sua recuperação através de queries ao índice FTS, mesmo após a eliminação física do registo na tabela principal.

Observações A análise da eliminação local permite concluir:

1. A opção "Limpar para mim" executa eliminação **física** do registo, acionando o *trigger* `messages_on_delete`;
2. O campo `isDeleted` **não é utilizado** — o registo é completamente removido;
3. O conteúdo persiste na tabela `messages_fts`, permitindo recuperação forense;
4. Adicionalmente, o conteúdo pode existir no *freelist* do SQLite e no ficheiro Write-Ahead Log.

As queries completas encontram-se no Anexo B.4.

Cenário 3b: Eliminação Global ("Limpar para todos")

Esta opção remove a mensagem do dispositivo local e envia um pedido de eliminação aos outros participantes da conversa.

Procedimento

1. Foi enviada uma nova mensagem de teste: "Mensagem teste eliminação global";
2. O estado da base de dados foi documentado antes da eliminação;
3. A mensagem foi eliminada utilizando a opção "Limpar para todos";
4. O estado da base de dados foi documentado após a eliminação.

Estado Antes da Eliminação A Tabela 3.14 apresenta o estado da base de dados antes da eliminação.

Tabela 3.14: Cenário 3b: Estado antes da eliminação global

Tabela	Registos
`conversations`	2
`messages`	4
`seenMessages`	10
`attachment_downloads`	0

Estado Após a Eliminação A Tabela 3.15 apresenta a comparação entre os estados antes e depois da eliminação.

Tabela 3.15: Cenário 3b: Comparação de registos

Tabela	Antes	Depois	Diferença
`conversations`	2	2	0
`messages`	4	3	-1
`seenMessages`	10	12	$+2$
`attachment_downloads`	0	0	0

Comparação com a Eliminação Local A Tabela 3.16 resume as diferenças entre as duas opções de eliminação.

Tabela 3.16: Comparação entre opções de eliminação

Aspeto	Limpar para mim	Limpar para todos
Registo em `messages`	Removido	Removido
Variação em `seenMessages`	$+1$	$+2$
Conteúdo no `messages_fts`	Persiste	Persiste
Notificação ao destinatário	Não	Sim

Observações A análise da eliminação global permite concluir:

1. A opção "Limpar para todos" executa eliminação **física** do registo, tal como a opção local;
2. A diferença reside na tabela `seenMessages`, que regista **dois novos registos** ($+2$ vs $+1$), sugerindo o envio de uma mensagem de controlo ao destinatário;
3. Do ponto de vista forense local, ambas as opções produzem resultados idênticos na tabela `messages`;
4. O conteúdo permanece recuperável através da tabela `messages_fts`, independentemente da opção escolhida;
5. O artefacto adicional em `seenMessages` permite identificar que foi utilizada eliminação global.

As queries completas encontram-se no Anexo B.5.

3.2.5 Cenário 4: Envio de Anexo

Este cenário documenta o comportamento da base de dados quando um ficheiro é anexado a uma mensagem, analisando a relação entre a tabela `messages` e o sistema de ficheiros.

Procedimento

1. Foi enviada uma imagem ao utilizador "Elliot" através da conversa existente;
2. O estado da base de dados foi documentado após o envio;
3. A pasta `attachments.noindex` foi inspecionada para identificar os ficheiros armazenados.

Alterações na Base de Dados

A Tabela 3.17 apresenta a comparação entre o estado anterior e o estado após o envio do anexo.

Tabela 3.17: Cenário 4: Comparação de registos

Tabela	Antes	Depois	Diferença
`conversations`	2	2	0
`messages`	3	4	$+1$
`seenMessages`	12	14	$+2$
`attachment_downloads`	0	0	0

Tabela 3.18: Cenário 4: Registo da mensagem com anexo

Campo	Valor
type	outgoing
body	(vazio)
hasAttachments	1
hasFileAttachments	0
hasVisualMediaAttachments	1
data_hora	2025-12-20 22:11:14

Tabela `messages` Foi criado um novo registo correspondente à mensagem com anexo. A Tabela 3.18 apresenta os campos relevantes.

Os campos `hasFileAttachments` e `hasVisualMediaAttachments` permitem distinguir o tipo de anexo: `hasVisualMediaAttachments=1` indica imagem ou vídeo, enquanto `hasFileAttachments=1` indica documentos.

Tabela `attachment_downloads` A tabela `attachment_downloads` permaneceu vazia. Esta tabela é utilizada apenas para gerir downloads de anexos **recebidos**, não para anexos enviados.

Armazenamento no Sistema de Ficheiros

Os anexos são armazenados na pasta `attachments.noindex`, organizada em subpastas baseadas nos primeiros caracteres do identificador. A Figura 3.4 ilustra a estrutura.

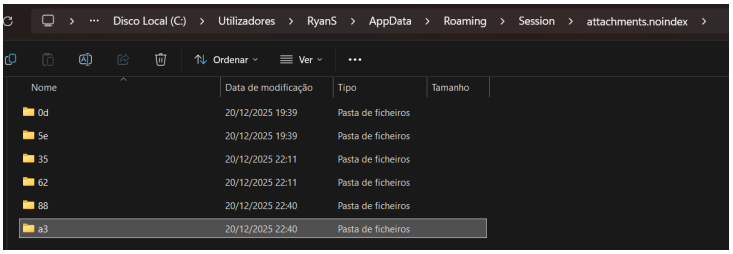


Figura 3.4: Estrutura da pasta `attachments.noindex`

A Tabela 3.19 apresenta os ficheiros identificados.

Tabela 3.19: Cenário 4: Ficheiros na pasta `attachments.noindex`

Subpasta	Identificador	Tamanho	Hora
0d/	0dbca138a21c4f683456...	2 KB	19:39
5e/	5ea9645afae12bf96f0a...	2 KB	19:39
35/	355c50ec07ec17646266...	1 KB	22:11
62/	624e6d3232bcec721bc1...	29 KB	22:11

O ficheiro de 29 KB na subpasta `62/`, com *timestamp* coincidente com o envio (22:11), corresponde ao anexo enviado.

Observações

A análise do envio de anexo permite concluir:

1. Os anexos são armazenados **fora da BD**, na pasta `attachments.noindex`, encriptados com XChaCha20-Poly1305;
2. Os nomes dos ficheiros são identificadores alfanuméricos, não preservando nome ou extensão original;
3. A organização em subpastas facilita a gestão de grandes volumes de anexos;
4. Os campos `hasAttachments`, `hasFileAttachments` e `hasVisualMediaAttachments` permitem categorizar anexos;
5. A tabela `attachment_downloads` é utilizada apenas para anexos recebidos;
6. Do ponto de vista forense, mesmo sem desencriptar os anexos, é possível determinar quantidade, tamanho e data de criação.

As queries completas encontram-se no Anexo B.6.

3.2.6 Cenário 5: Receção de Anexo

Este cenário documenta o comportamento da base de dados quando um anexo é recebido, complementando a análise do Cenário 4.

Procedimento

Foram realizados dois testes de receção:

1. O utilizador "Elliot" enviou uma **imagem** ao utilizador "Spider";
2. O utilizador "Elliot" enviou um **documento PDF** ao utilizador "Spider";
3. Ambos os anexos foram visualizados e descarregados no Session Desktop;
4. O estado da base de dados foi documentado após as operações.

Alterações na Base de Dados

A Tabela 3.20 apresenta a evolução dos registos.

Tabela 3.20: Cenário 5: Comparação de registos

Tabela	Antes	Depois	Diferença
`conversations`	2	2	0
`messages`	4	6	$+2$
`seenMessages`	14	17	$+3$
`attachment_downloads`	0	0	0

Tabela `messages` Foram criados dois novos registos. A Tabela 3.21 apresenta os campos que distinguem os tipos de anexo.

Tabela `attachment_downloads` A tabela permaneceu vazia após receção e download, sugerindo que é utilizada apenas para downloads **em progresso**, sendo os registos removidos após conclusão.

Tabela 3.21: Cenário 5: Comparação entre tipos de anexo recebido

Campo	Imagem	PDF
type	incoming	incoming
hasAttachments	1	1
hasFileAttachments	0	1
hasVisualMediaAttachments	1	0
data_hora	22:39:49	22:45:43

Armazenamento de Anexos Recebidos

Ao contrário dos anexos enviados, os anexos recebidos são guardados na localização escolhida pelo utilizador (ex: pasta `Downloads`).

A Tabela 3.22 resume as diferenças.

Tabela 3.22: Comparação do armazenamento: enviado vs recebido

Aspeto	Enviado	Recebido
Localização	`attachments.noindex/`	Escolhida pelo utilizador
Encriptação	XChaCha20-Poly1305	Desencriptado
Nome	Identificador alfanumérico	Nome original
Persistência	Automática	Depende do utilizador

Observações

A análise da receção de anexos permite concluir:

1. Os anexos recebidos **não são armazenados** na pasta `attachments.noindex` — esta contém apenas anexos enviados;
2. Os anexos recebidos são guardados **desencriptados** com o nome original, na localização escolhida pelo utilizador;
3. Os campos `hasFileAttachments` e `hasVisualMediaAttachments` permitem distinguir tipos;
4. A tabela `attachment_downloads` não mantém histórico de downloads concluídos;
5. A identificação forense de anexos recebidos requer análise combinada da BD e do sistema de ficheiros.

As queries completas encontram-se no Anexo B.7.

3.2.7 Cenário 6: Receção de Áudio e Eliminação Remota

Este cenário combina dois testes: a receção de uma mensagem de áudio e a análise do comportamento da base de dados quando o remetente elimina a mensagem utilizando a opção "Limpar para todos".

Cenário 6a: Receção de Mensagem de Áudio

1. O utilizador "Elliot" gravou e enviou uma mensagem de áudio ao utilizador "Spider";
2. O utilizador "Spider" recebeu e reproduziu o áudio no Session Desktop;
3. O estado da base de dados foi documentado.

Alterações na Base de Dados A Tabela 3.23 apresenta o estado da base de dados após a receção do áudio.

Tabela 3.23: Cenário 6a: Estado após receção de áudio

Tabela	Registos
`conversations`	2
`messages`	7
`seenMessages`	18
`attachment_downloads`	0

Classificação de Mensagens de Áudio A análise revelou que as mensagens de áudio utilizam uma combinação distinta de campos. A Tabela 3.24 apresenta a comparação entre os diferentes tipos de anexo.

Tabela 3.24: Padrão de identificação de tipos de anexo

Tipo	hasAttachments	hasFileAttachments	hasVisualMedia
Imagem/Vídeo	1	0	1
Documento	1	1	0
Áudio	1	0	0

As mensagens de áudio distinguem-se por terem `hasAttachments=1` com ambos os campos `hasFileAttachments` e `hasVisualMediaAttachments` a zero.

Armazenamento do Ficheiro de Áudio Ao contrário dos documentos (que requerem download manual), as mensagens de áudio são armazenadas automaticamente na pasta `attachments.noindex` para permitir reprodução *inline*, comportamento semelhante ao das imagens.

Foi identificado um novo ficheiro na subpasta `26/`:

Tabela 3.25: Cenário 6a: Ficheiro de áudio identificado

Atributo	Valor
Subpasta	`26/`
Identificador	`261393d193f753bee77523e84bb603c0aa...`
Tamanho	23 KB
Data	2025-12-20 23:02

As queries completas encontram-se no Anexo B.8.1.

Cenário 6b: Eliminação Remota ("Limpar para todos" pelo Remetente)

Este cenário analisa o comportamento da base de dados do destinatário quando o remetente elimina uma mensagem utilizando a opção "Limpar para todos".

Procedimento

1. O utilizador "Elliot" (remetente) eliminou a mensagem de áudio utilizando a opção "Limpar para todos";
2. O utilizador "Spider" (destinatário) abriu o Session Desktop para receber a notificação de eliminação;
3. O estado da base de dados do destinatário foi documentado.

Alterações na Base de Dados A Tabela 3.26 apresenta a comparação entre os estados antes e depois da eliminação remota.

Tabela 3.26: Cenário 6b: Comparação de registos

Tabela	Antes	Depois	Diferença
`conversations`	2	2	0
`messages`	7	7	0
`seenMessages`	18	19	+1
`attachment_downloads`	0	0	0

Ao contrário da eliminação local (Cenário 3), a eliminação remota **não remove o registo** da tabela `messages`. O registo é **atualizado** conforme apresentado na Tabela 3.27.

Tabela 3.27: Cenário 6b: Alterações no registo após eliminação remota

Campo	Antes	Depois
`body`	(vazio)	"Esta mensagem foi apagada"
`hasAttachments`	1	0
`hasFileAttachments`	0	0
`hasVisualMediaAttachments`	0	0

Persistência do Ficheiro de Anexo Apesar da eliminação remota, o ficheiro de áudio **permanece** na pasta `attachments.noindex/26/`, com o mesmo tamanho (23 KB) e *timestamp* original. O anexo pode ser potencialmente recuperado através da chave armazenada na tabela `items`.

Descoberta Forense Crítica

A eliminação remota ("Limpar para todos" pelo remetente) executa uma operação `UPDATE` na base de dados do destinatário, não `DELETE`. O registo original é mantido com o campo `body` substituído por um *placeholder*, e os ficheiros de anexo **permanecem no sistema de ficheiros**, permitindo recuperação forense.

Comparação: Eliminação Local vs Remota A Tabela 3.28 resume as diferenças entre os dois tipos de eliminação.

Tabela 3.28: Comparação entre eliminação local e eliminação remota

Aspeto	Local	Remota
Operação SQL	DELETE	UPDATE
Registo em `messages`	Removido	Mantido
Campo `body`	N/A	"Esta mensagem foi apagada"
Campos de anexo	N/A	Limpos (valor 0)
Ficheiro em `attachments.noindex`	Persiste	Persiste

Observações do Cenário 6

A análise combinada da receção de áudio e eliminação remota permite concluir:

1. As mensagens de áudio são identificadas por `hasAttachments=1` com `hasFileAttachments=0` e `hasVisualMediaAttachments=0`;

2. Os ficheiros de áudio recebidos são armazenados na pasta `attachments.noindex` para reprodução *inline*;
3. A eliminação remota **não remove** o registo da BD do destinatário — apenas atualiza o campo `body`;
4. Os ficheiros de anexos **permanecem no sistema de ficheiros** após eliminação remota, sendo recuperáveis com a chave na tabela `items`;
5. A diferença entre eliminação local (DELETE) e remota (UPDATE) representa uma vulnerabilidade de privacidade, mas uma oportunidade para análise forense.

As queries completas encontram-se no Anexo B.8.2.

4. Conclusão

Este capítulo encerra o relatório apresentando uma síntese do trabalho desenvolvido. Avalia-se o grau de cumprimento dos objetivos traçados, identificam-se os contributos técnicos e as limitações inerentes ao escopo do projeto. Por fim, são delineadas perspetivas de evolução e trabalhos futuros que visam a continuidade da investigação na área da análise forense a aplicações de mensagens privadas.

4.1 Síntese do Trabalho Desenvolvido

O presente projeto assumiu como objetivo a realização de uma análise forense comportamental à aplicação Session Desktop, uma plataforma de mensagens descentralizada focada na privacidade que utiliza encriptação de ponta a ponta e armazenamento local encriptado com SQLCipher.

A investigação materializou-se através de uma metodologia sistemática composta por nove cenários de teste, abrangendo as operações fundamentais da aplicação: criação de conversas, envio e receção de mensagens, eliminação local e remota, e gestão de anexos (imagens, documentos e áudio). A análise incidiu sobre a base de dados SQLite encriptada e o sistema de ficheiros associado, com particular foco na identificação de artefactos forenses recuperáveis.

Os resultados empíricos validam a hipótese de partida: apesar das funcionalidades de privacidade implementadas pelo Session — incluindo encriptação de ponta a ponta e ausência de servidores centralizados — existem múltiplas vulnerabilidades que permitem a recuperação de informação por um investigador forense com acesso físico ao dispositivo.

4.2 Aferição do Cumprimento dos Objetivos

Esta secção apresenta uma análise crítica do cumprimento dos objetivos estabelecidos no Capítulo **??**. Para cada objetivo proposto, avalia-se o grau de concretização alcançado, cotejando as metas iniciais com os resultados práticos obtidos através da análise forense realizada.

4.2.1 Objetivo Geral

Objetivo: Realizar uma análise forense comportamental à aplicação Session Desktop, identificando artefactos recuperáveis e vulnerabilidades de segurança no armazenamento local de dados.

Status: *Integralmente Alcançado*

A análise forense foi concluída com sucesso, identificando múltiplas vulnerabilidades e documentando o comportamento da aplicação em diversos cenários de utilização. Os resultados demonstram que a recuperação de informação é viável mesmo em aplicações focadas na privacidade.

4.2.2 Objetivos Específicos

Objetivo 1: Localização e Acesso à Base de Dados **Status:** *Alcançado*

Foi identificada a localização da base de dados em `%AppData%\Roaming\Session\sql\` e documentado o método de desencriptação através da chave armazenada no ficheiro `config.json`. A vulnerabilidade da chave em texto claro foi identificada e documentada.

Objetivo 2: Mapeamento da Estrutura da Base de Dados **Status:** *Alcançado*

A estrutura completa da base de dados foi mapeada, identificando 19 tabelas, 25 índices e 3 *triggers*. As tabelas principais (`messages`, `conversations`) foram analisadas em detalhe, incluindo a documentação de todos os campos relevantes para análise forense.

Objetivo 3: Análise Comportamental **Status:** *Integralmente Alcançado*

Foram realizados nove cenários de teste cobrindo todas as operações principais: estado inicial, criação de conversa, envio e receção de mensagens, eliminação local (duas variantes), eliminação remota, e gestão de anexos (envio, receção de imagem/documento, receção de áudio).

Objetivo 4: Identificação de Artefactos Forenses **Status:** *Alcançado*

Foram identificados múltiplos artefactos forenses persistentes, incluindo: conteúdo de mensagens eliminadas na tabela `messages_fts`, ficheiros de anexos na pasta `attachments.noindex`, e metadados de comunicação nas tabelas principais.

Objetivo 5: Análise de Mecanismos de Eliminação **Status:** *Integralmente Alcançado*

Foram comparados três tipos de eliminação, identificando diferenças críticas: a eliminação local executa `DELETE` físico mas mantém conteúdo no índice FTS; a eliminação remota apenas atualiza o campo `body` sem remover o registo ou ficheiros de anexo. Esta descoberta representa uma contribuição significativa para a área forense.

Objetivo 6: Documentação de Vulnerabilidades **Status:** *Alcançado*

Foram documentadas as principais vulnerabilidades de segurança: chave de encriptação exposta, mensagens em texto claro após desencriptação, persistência de dados eliminados, e comportamento inadequado da eliminação remota.

4.3 Principais Descobertas

A investigação revelou várias descobertas com relevância forense significativa.

4.3.1 Vulnerabilidades de Armazenamento Local

A análise revelou múltiplas vulnerabilidades no armazenamento local:

- **Chave de encriptação exposta:** A chave de desencriptação da base de dados SQLCipher está armazenada em texto claro no ficheiro `config.json`, permitindo acesso completo à base de dados por qualquer processo com acesso ao sistema de ficheiros;
- **Mensagens em texto claro:** Após a desencriptação da base de dados, o conteúdo das mensagens encontra-se em **texto claro**, sem qualquer camada adicional de encriptação, demonstrando que a proteção de ponta a ponta termina no momento em que a mensagem é armazenada localmente;
- **Metadados acessíveis:** Informações como *timestamps*, identificadores de contactos, estados de aprovação e tipos de anexo estão igualmente acessíveis sem proteção adicional.

4.3.2 Persistência de Dados Eliminados

A análise dos *triggers* e dos cenários de eliminação revelou:

- A eliminação local executa uma operação `DELETE` física na tabela `messages`;
- O conteúdo das mensagens **persiste na tabela** `messages_fts` (índice *full-text*), permitindo a recuperação do texto de mensagens eliminadas;
- Os ficheiros de anexos **permanecem na pasta** `attachments.noindex` mesmo após eliminação das mensagens associadas.

4.3.3 Comportamento da Eliminação Remota

Quando o remetente elimina uma mensagem "para todos", o comportamento no dispositivo do destinatário difere significativamente:

- O registo **não é eliminado** da tabela `messages`;
- O campo `body` é atualizado para "Esta mensagem foi apagada";
- Os ficheiros de anexo **permanecem** no sistema de ficheiros.

4.3.4 Classificação de Tipos de Anexo

A análise identificou um padrão consistente para categorização de anexos, apresentado na Tabela 4.1.

Tabela 4.1: Padrão de identificação de tipos de anexo

Tipo	hasAttachments	hasFileAttachments	hasVisualMedia
Imagem/Vídeo	1	0	1
Documento	1	1	0
Áudio	1	0	0

4.4 Contributos do Trabalho

O projeto aporta valor em múltiplas dimensões:

1. **Técnica:** Disponibiliza uma metodologia de análise forense replicável para aplicações de mensagens encriptadas, documentando queries SQL e procedimentos de acesso à base de dados SQLCipher;
2. **Forense:** Identifica artefactos recuperáveis que podem constituir prova em investigações criminais, demonstrando que a eliminação de mensagens não garante a sua destruição completa;
3. **Segurança:** Expõe vulnerabilidades no modelo de armazenamento local do Session, alertando utilizadores e programadores para as limitações da encriptação de ponta a ponta contra ameaças com acesso físico;
4. **Académica:** O material produzido serve como recurso didático para o ensino de análise forense digital, bases de dados SQLite e criptografia aplicada.

4.5 Limitações e Desafios

A honestidade intelectual obriga ao reconhecimento das limitações do projeto:

- **Plataforma única:** A análise foi realizada apenas na versão Desktop do Session para Windows; o comportamento nas versões móveis (Android/iOS) pode diferir significativamente;
- **Desencriptação de anexos:** Não foi realizada a desencriptação efetiva dos ficheiros de anexos, apenas documentada a sua persistência e o método teórico de recuperação;
- **Recuperação avançada:** A análise não abrangeu técnicas avançadas como análise do *freelist* do SQLite ou do ficheiro WAL para recuperação de registos eliminados fisicamente;
- **Funcionalidades não testadas:** As funcionalidades de chamadas de voz/vídeo e conversas de grupo não foram incluídas na análise;
- **Versão específica:** Os resultados referem-se à versão testada do Session Desktop, podendo diferir em versões futuras caso as vulnerabilidades sejam corrigidas.

4.6 Reflexão e Aprendizagem

O desenvolvimento deste projeto constituiu um vetor de aprendizagem significativo na área da análise forense digital. Do ponto de vista técnico, consolidou-se o domínio sobre bases de dados SQLite, encriptação SQLCipher, e metodologias de análise comportamental.

Evidenciou-se que aplicações focadas na privacidade, apesar das suas garantias de encriptação de ponta a ponta, apresentam frequentemente vulnerabilidades no armazenamento local que podem ser exploradas por investigadores forenses. A principal lição retida é que a "*privacidade*" promovida por estas aplicações protege primariamente contra interceção em trânsito, oferecendo proteção limitada contra análise forense local.

A metodologia sistemática de testes demonstrou-se essencial para uma análise completa, permitindo documentar comportamentos que não seriam evidentes numa inspeção superficial da base de dados.

4.7 Trabalho Futuro

Considerando os resultados obtidos e as limitações identificadas, delineiam-se as seguintes perspetivas de evolução:

1. **Análise das versões móveis:** Investigar o comportamento das aplicações Session para Android e iOS, comparando com os resultados obtidos na versão Desktop e identificando diferenças no armazenamento local;
2. **Recuperação avançada de dados:** Aplicar técnicas de análise do *freelist* do SQLite e do ficheiro WAL para recuperação de registos eliminados fisicamente, expandindo as possibilidades de recuperação forense;
3. **Desencriptação de anexos:** Desenvolver ferramentas ou scripts para desencriptação dos ficheiros na pasta `attachments.noindex` utilizando a chave armazenada na tabela `items`, validando a recuperação completa de conteúdo multimédia;
4. **Análise de chamadas:** Investigar os artefactos forenses gerados pelas funcionalidades de chamadas de voz e vídeo, identificando potenciais metadados sobre a atividade dos utilizadores;
5. **Análise de grupos:** Estender a análise para conversas de grupo e comunidades, verificando diferenças no armazenamento e comportamento face às conversas privadas;
6. **Comparação com outras aplicações:** Realizar análise comparativa com outras aplicações de mensagens focadas na privacidade, como Signal ou Wickr, identificando padrões comuns de vulnerabilidades.

4.8 Considerações Finais

Em suma, o presente trabalho encerra-se com a convicção de que a análise desenvolvida ultrapassa o estatuto de mero exercício académico, constituindo uma contribuição relevante para a área da forense digital aplicada a aplicações de mensagens privadas.

As principais vulnerabilidades identificadas — chave de encriptação exposta, mensagens em texto claro, persistência no índice FTS, manutenção de anexos após eliminação, e comportamento inadequado da eliminação remota — demonstram que a encriptação de ponta a ponta protege eficazmente as comunicações em trânsito, mas oferece proteção limitada contra análise forense com acesso físico ao dispositivo.

Estas descobertas são particularmente relevantes em dois contextos distintos:

- **Investigações forenses:** A recuperação de comunicações supostamente eliminadas ou protegidas pode constituir prova fundamental em investigações criminais;
- **Privacidade dos utilizadores:** Os resultados alertam para as limitações das garantias de privacidade oferecidas por este tipo de aplicações, especialmente contra ameaças com acesso físico ao dispositivo.

Fica estabelecida uma base metodológica sólida que pode servir de referência para análises forenses futuras a aplicações similares, contribuindo para o conhecimento na interseção entre cibersegurança, privacidade e investigação criminal.

Capítulo

Referências

DB Browser for SQLite Development Team. (2024). *DB Browser for SQLite*. Acedido em novembro 27, 2024, de https://sqlitebrowser.org/.

Jefferys, K., Tokarev, M., & Harman, S. (2024). Session: End-To-End Encrypted Conversations With Minimal Metadata Leakage. *arXiv preprint*. https://arxiv.org/pdf/2002.04609.

Session Technology Foundation. (2024). *Session: Send Messages, Not Metadata*. Acedido em novembro 27, 2024, de https://getsession.org/.

Zetetic LLC. (2024). *SQLCipher: Encrypted SQLite Database*. Acedido em novembro 27, 2024, de https://www.zetetic.net/sqlcipher/.

Capítulo

Certificação de Integridade

4.8.1 Prova de Existência e Integridade

Para garantir a integridade e comprovar a data de conclusão deste trabalho, o presente relatório foi submetido ao protocolo **OpenTimestamps**[1], que permite criar provas de existência de documentos ancoradas na blockchain do Bitcoin.

O OpenTimestamps funciona através da criação de um *hash* criptográfico do documento, que é posteriormente incluído numa transação Bitcoin. Uma vez confirmada a transação, a prova torna-se imutável e permanentemente verificável por qualquer pessoa.

Este processo garante:

- **Prova de Existência** — Demonstração criptográfica de que o documento existia numa data específica;
- **Imutabilidade** — O carimbo temporal está permanentemente registado na blockchain;
- **Verificação Independente** — Qualquer pessoa pode verificar a autenticidade do carimbo temporal.

O ficheiro de prova (`.ots`) está disponível no repositório GitHub do projeto[2], juntamente com instruções para verificação.

[1] https://opentimestamps.org/
[2] https://github.com/RyanTech00/session-desktop-forensics

A. Estrutura Detalhada da Base de Dados

Este anexo apresenta a documentação completa da estrutura da base de dados do Session Desktop, complementando a análise realizada na Secção 3.1. São listadas todas as tabelas e índices identificados, bem como a estrutura detalhada das duas tabelas principais: `messages` e `conversations`.

A.1 Lista Completa de Tabelas

A base de dados do Session Desktop contém 19 tabelas, apresentadas na Tabela A.1. Destas, seis tabelas (`messages_fts*`) estão dedicadas à funcionalidade de pesquisa *full-text*, enquanto as restantes armazenam dados operacionais da aplicação, incluindo mensagens, conversas, chaves criptográficas e configurações de rede.

Tabela A.1: Lista completa de tabelas da base de dados

#	Tabela	Descrição
1	`attachment_downloads`	Gestão de downloads de anexos
2	`configDump`	Exportação de configuração
3	`conversations`	Conversas e contactos
4	`encryptionKeyPairsForClosedGroupV2`	Chaves para grupos fechados
5	`guardNodes`	Nós de guarda (*onion routing*)
6	`identityKeys`	Chaves de identidade
7	`items`	Configuração e metadados
8	`lastHashes`	Últimos *hashes* processados
9	`loki_schema`	Versão do esquema da BD
10	`messages`	Mensagens enviadas e recebidas
11	`messages_fts`	Índice *full-text search*
12	`messages_fts_config`	Configuração do FTS
13	`messages_fts_content`	Conteúdo indexado
14	`messages_fts_data`	Dados do índice FTS
15	`messages_fts_docsize`	Tamanho dos documentos
16	`messages_fts_idx`	Índice auxiliar FTS
17	`nodesForPubkey`	Nós por chave pública
18	`openGroupRoomsV2`	Comunidades públicas
19	`seenMessages`	Mensagens vistas

A.2 Lista Completa de Índices

Foram identificados 25 índices na base de dados, criados para otimizar as operações de consulta. A Tabela A.2 apresenta todos os índices organizados pela tabela a que estão associados.

Destaca-se que 18 dos 25 índices (72%) estão associados à tabela `messages`, refletindo a importância central desta tabela na aplicação e a necessidade de otimizar consultas por diversos critérios, como: estado de leitura, presença de anexos, *timestamps* de expiração e *hashes* de mensagens.

Tabela A.2: Lista completa de índices da base de dados

#	Índice	Tabela Associada
1	attachment_downloads_pending	attachment_downloads
2	attachment_downloads_timestamp	attachment_downloads
3	conversation_displayNameInProfile	conversations
4	conversation_nickname	conversations
5	conversations_active	conversations
6	conversations_type	conversations
7	messages_DaR_unread_sent_at	messages
8	messages_conversation	messages
9	messages_conversationId	messages
10	messages_convo_serverID	messages
11	messages_duplicate_check	messages
12	messages_expires_at	messages
13	messages_hasAttachments	messages
14	messages_hasFileAttachments	messages
15	messages_hasVisualMediaAttachments	messages
16	messages_isDeleted	messages
17	messages_receipt	messages
18	messages_serverHash	messages
19	messages_t_messageHash	messages
20	messages_t_messageHash_author	messages
21	messages_t_messageHash_author_convoId	messages
22	messages_unread	messages
23	messages_unread_by_conversation	messages
24	messages_without_timer	messages
25	seen_hashes_per_pubkey	seenMessages

A.3 Estrutura da Tabela `messages`

A tabela `messages` é a tabela central da aplicação Session Desktop, responsável pelo armazenamento de todas as mensagens trocadas pelo utilizador. Com 24 colunas, esta tabela contém não apenas o conteúdo das mensagens, mas também metadados essenciais para análise forense, incluindo *timestamps*, identificadores de conversa, estado de leitura e informação sobre anexos.

A Tabela A.3 apresenta a estrutura completa, incluindo o tipo de dados de cada coluna e a indicação da chave primária.

Tabela A.3: Estrutura completa da tabela `messages`

#	Coluna	Tipo	Descrição
0	`id`	STRING	Identificador único (PK)
1	`json`	TEXT	Dados completos em JSON
2	`unread`	INTEGER	Não lida (1/0)
3	`expires_at`	INTEGER	*Timestamp* expiração
4	`sent`	BOOLEAN	Foi enviada
5	`sent_at`	INTEGER	*Timestamp* envio
6	`conversationId`	STRING	ID da conversa
7	`received_at`	INTEGER	*Timestamp* receção
8	`source`	STRING	Origem da mensagem
9	`hasAttachments`	INTEGER	Tem anexos (1/0)
10	`hasFileAttachments`	INTEGER	Tem ficheiros
11	`hasVisualMediaAttachments`	INTEGER	Tem média visual
12	`expireTimer`	INTEGER	Temporizador expiração
13	`expirationStartTimestamp`	INTEGER	Início contagem
14	`type`	STRING	incoming/outgoing
15	`body`	TEXT	Conteúdo da mensagem
16	`serverId`	INTEGER	ID no servidor
17	`serverTimestamp`	INTEGER	*Timestamp* servidor
18	`serverHash`	TEXT	*Hash* no servidor
19	`isDeleted`	BOOLEAN	Foi eliminada
20	`expirationType`	TEXT	Tipo de expiração
21	`flags`	INTEGER	*Flags* de estado
22	`messageHash`	TEXT	*Hash* da mensagem
23	`errors`	TEXT	Erros ocorridos

A.4 Estrutura da Tabela `conversations`

A tabela `conversations` armazena informação sobre todos os contactos e conversas do utilizador. Com 34 colunas, esta tabela regista não apenas identificadores e nomes, mas também o estado do mecanismo de aprovação de contactos (`isApproved`, `didApproveMe`), configurações de notificações, temporizadores de expiração de mensagens e metadados de perfil.

A Tabela A.4 apresenta a estrutura completa desta tabela, incluindo os valores por omissão definidos para cada coluna.

Tabela A.4: Estrutura completa da tabela `conversations`

#	Coluna	Tipo	Default
0	id	STRING	–
1	active_at	INTEGER	NULL
2	type	STRING	NULL
3	members	TEXT	NULL
4	left	INTEGER	NULL
5	expireTimer	INTEGER	NULL
6	mentionedUs	INTEGER	NULL
7	unreadCount	INTEGER	NULL
8	lastMessageStatus	TEXT	NULL
9	lastMessage	TEXT	NULL
10	lastJoinedTimestamp	INTEGER	NULL
11	groupAdmins	TEXT	"[]"
12	avatarPointer	TEXT	NULL
13	nickname	TEXT	NULL
14	profileKey	TEXT	NULL
15	triggerNotificationsFor	TEXT	"all"
16	isTrustedForAttachmentDownload	INTEGER	"FALSE"
17	priority	INTEGER	"FALSE"
18	isApproved	INTEGER	"FALSE"
19	didApproveMe	INTEGER	"FALSE"
20	avatarInProfile	TEXT	NULL
21	displayNameInProfile	TEXT	NULL
22	conversationIdOrigin	TEXT	NULL
23	markedAsUnread	BOOLEAN	NULL
24	blocksSogsMsgReqsTimestamp	INTEGER	NULL
25	expirationMode	TEXT	"off"
26	lastMessageInteractionType	TEXT	NULL
27	lastMessageInteractionStatus	TEXT	NULL
28	isExpired03Group	BOOLEAN	NULL
29	fallbackAvatarInProfile	TEXT	NULL
30	profileUpdatedSeconds	INTEGER	NULL
31	bitsetProFeatures	TEXT	NULL
32	proGenIndexHashB64	TEXT	NULL
33	proExpiryTsMs	INTEGER	NULL

B. Evidências dos Cenários de Teste

Este anexo apresenta as capturas de ecrã das queries executadas durante os cenários de teste, servindo como evidência da análise forense realizada.

B.1 Estado Inicial

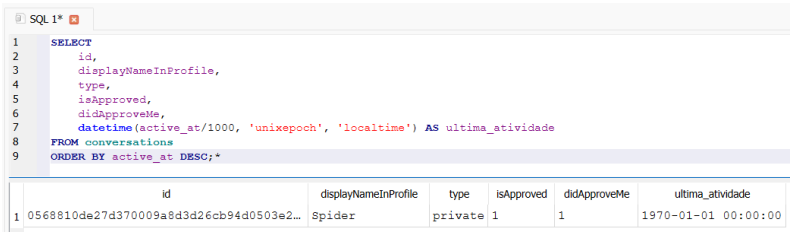


Figura B.1: Query à tabela `conversations` no estado inicial

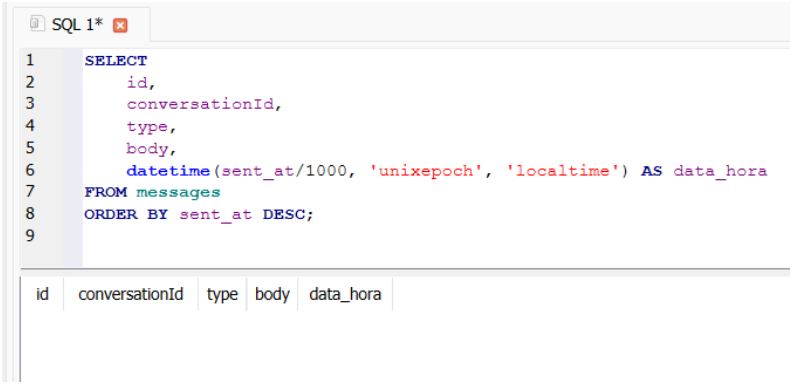


Figura B.2: Query à tabela `messages` no estado inicial

B.2 Cenário 1: Criação de Nova Conversa

As seguintes figuras documentam as queries executadas após a criação de uma nova conversa entre os utilizadores "Spider" e "Elliot".

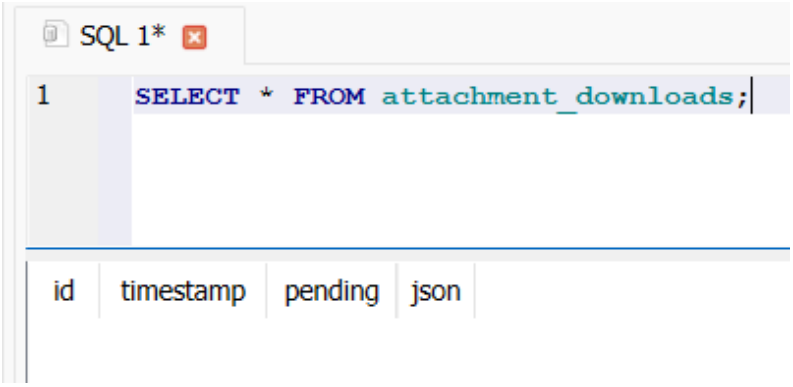


Figura B.3: Query à tabela `attachment_downloads` no estado inicial

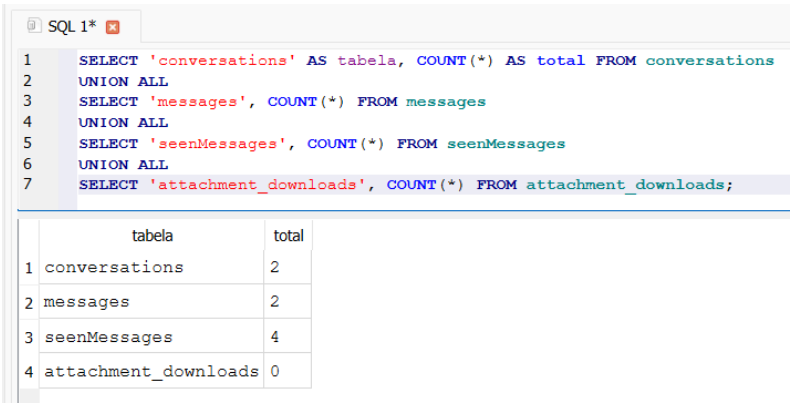


Figura B.4: Query de contagem de registos após criação de conversa

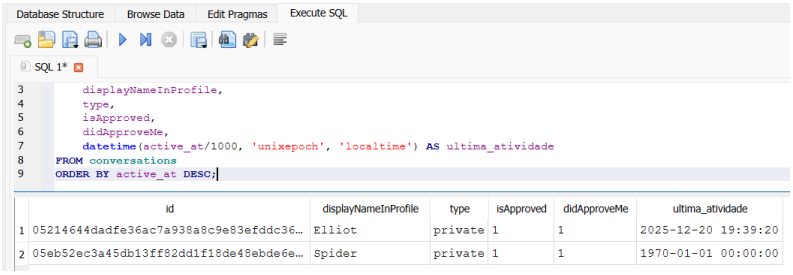


Figura B.5: Query à tabela `conversations` após criação de conversa

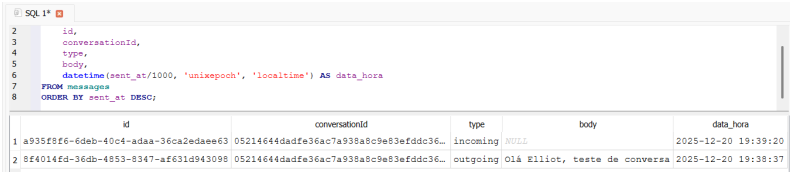


Figura B.6: Query à tabela `messages` após criação de conversa

B.3 Cenário 2: Receção de Mensagem

As seguintes figuras documentam as queries executadas após a receção de uma mensagem do utilizador "Elliot".

SQL 1*

```
1  SELECT 'conversations' AS tabela, COUNT(*) AS total FROM conversations
2  UNION ALL
3  SELECT 'messages', COUNT(*) FROM messages
4  UNION ALL
5  SELECT 'seenMessages', COUNT(*) FROM seenMessages
6  UNION ALL
7  SELECT 'attachment_downloads', COUNT(*) FROM attachment_downloads;
```

	tabela	total
1	conversations	2
2	messages	3
3	seenMessages	5
4	attachment_downloads	0

Figura B.7: Query de contagem de registos após receção de mensagem

SQL 1*

```
1  SELECT
2      id,
3      conversationId,
4      type,
5      body,
6      datetime(sent_at/1000, 'unixepoch', 'localtime') AS data_hora
7  FROM messages
8  ORDER BY sent_at DESC;
```

	id	conversationId	type	body	data_hora
1	0d55671b-c4d5-4e6b-966f-ccc9ac8cf9c1	05214644dadfe36ac7a938a8c9e83efddc36…	incoming	Olá Spider, mensagem recebida e …	2025-12-20 19:59:45
2	a935f8f6-6deb-40c4-adaa-36ca2edaee63	05214644dadfe36ac7a938a8c9e83efddc36…	incoming	NULL	2025-12-20 19:39:20
3	8f4014fd-36db-4853-8347-af631d943098	05214644dadfe36ac7a938a8c9e83efddc36…	outgoing	Olá Elliot, teste de conversa	2025-12-20 19:38:37

Figura B.8: Query à tabela `messages` após receção

SQL 1*

```
1  SELECT
2      id,
3      displayNameInProfile,
4      type,
5      isApproved,
6      didApproveMe,
7      datetime(active_at/1000, 'unixepoch', 'localtime') AS ultima_atividade
8  FROM conversations
9  ORDER BY active_at DESC;
```

	id	displayNameInProfile	type	isApproved	didApproveMe	ultima_atividade
1	05214644dadfe36ac7a938a8c9e83efddc36…	Elliot	private	1	1	2025-12-20 19:59:45
2	05eb52ec3a45db13ff82dd1f18de48ebde6e…	Spider	private	1	1	1970-01-01 00:00:00

Figura B.9: Query à tabela `conversations` com `active_at` atualizado

B.4 Cenário 3a: Eliminação Local

As seguintes figuras documentam as queries executadas antes e depois da eliminação de uma mensagem utilizando a opção "Limpar para mim".

SQL 1*

```
-- Contagem
SELECT 'conversations' AS tabela, COUNT(*) AS total FROM conversations
UNION ALL
SELECT 'messages', COUNT(*) FROM messages
UNION ALL
SELECT 'seenMessages', COUNT(*) FROM seenMessages
UNION ALL
SELECT 'attachment_downloads', COUNT(*) FROM attachment_downloads;
```

	tabela	total
1	conversations	2
2	messages	4
3	seenMessages	7
4	attachment_downloads	0

Figura B.10: Contagem de registos antes da eliminação local

SQL 1*

```
SELECT
    id,
    conversationId,
    type,
    body,
    isDeleted,
    datetime(sent_at/1000, 'unixepoch', 'localtime') AS data_hora
FROM messages
ORDER BY sent_at DESC;
```

	id	conversationId	type	body	isDeleted	data_hora
1	ee54c372-af7a-4faf-9d31-3cc56ed8a2a7	05214644dadfe36ac7a938a8c9e83efddc36…	outgoing	Mensagem teste eliminação local	NULL	2025-12-20 21:10:41
2	0d55671b-c4d5-4e6b-966f-ccc9ac8cf9c1	05214644dadfe36ac7a938a8c9e83efddc36…	incoming	Olá Spider, mensagem recebida e …	NULL	2025-12-20 19:59:45
3	a935f8f6-6deb-40c4-adaa-36ca2edaee63	05214644dadfe36ac7a938a8c9e83efddc36…	incoming	NULL	NULL	2025-12-20 19:39:20
4	5f4014fd-36db-4853-8347-af631d943098	05214644dadfe36ac7a938a8c9e83efddc36…	outgoing	Olá Elliot, teste de conversa	NULL	2025-12-20 19:38:37

Figura B.11: Tabela `messages` antes da eliminação local

B.4.1 Estado Antes da Eliminação

B.4.2 Estado Após a Eliminação

B.5 Cenário 3b: Eliminação Global

As seguintes figuras documentam as queries executadas antes e depois da eliminação de uma mensagem utilizando a opção "Limpar para todos".

B.5.1 Estado Antes da Eliminação

B.5.2 Estado Após a Eliminação

B.6 Cenário 4: Envio de Anexo

As seguintes figuras documentam as queries executadas e a estrutura do sistema de ficheiros após o envio de um anexo.

SQL 1*

```
1  -- Índice FTS (para verificar o trigger)
2  SELECT rowid, body FROM messages_fts;
```

	rowid	body
1	1	Olá Elliot, teste de conversa
2	2	Olá Elliot, teste de conversa
3	3	Olá Elliot, teste de conversa
4	4	Olá Elliot, teste de conversa
5	5	Olá Elliot, teste de conversa
6	6	Olá Elliot, teste de conversa
7	7	Olá Elliot, teste de conversa
8	8	Olá Elliot, teste de conversa
9	9	Olá Elliot, teste de conversa
10	10	*NULL*
11	11	*NULL*
12	12	Olá Spider, mensagem recebida e …
13	13	Olá Spider, mensagem recebida e …
14	14	Mensagem teste eliminação local
15	15	Mensagem teste eliminação local
16	16	Mensagem teste eliminação local
17	17	Mensagem teste eliminação local
18	18	Mensagem teste eliminação local
19	19	Mensagem teste eliminação local
20	20	Mensagem teste eliminação local
21	21	Mensagem teste eliminação local

Figura B.12: Índice `messages_fts` antes da eliminação local

SQL 1*

```
5  UNION ALL
6  SELECT 'seenMessages', COUNT(*) FROM seenMessages
7  UNION ALL
8  SELECT 'attachment_downloads', COUNT(*) FROM attachment_downloads;
```

	tabela	total
1	conversations	2
2	messages	3
3	seenMessages	8
4	attachment_downloads	0

Figura B.13: Contagem de registos após eliminação local

SQL 1*

```
1  -- Mensagens
2  SELECT
3      id,
4      conversationId,
5      type,
6      body,
7      isDeleted,
8      datetime(sent_at/1000, 'unixepoch', 'localtime') AS data_hora
9  FROM messages
10 ORDER BY sent_at DESC;
```

	id	conversationId	type	body	isDeleted	data_hora
1	0d55671b-c4d5-4a6b-966f-ccc9ac8cf9c1	05214644dadfe36ac7a938a8c9e83efddc36…	incoming	Olá Spider, mensagem recebida e pedid…	*NULL*	2025-12-20 19:59:45
2	a935f8f6-6deb-40c4-adaa-36ca2edaee63	05214644dadfe36ac7a938a8c9e83efddc36…	incoming	*NULL*	*NULL*	2025-12-20 19:39:20
3	8f4014fd-36db-4853-8347-af631d943098	05214644dadfe36ac7a938a8c9e83efddc36…	outgoing	Olá Elliot, teste de conversa	*NULL*	2025-12-20 19:38:37

Figura B.14: Tabela `messages` após eliminação local

SQL 1*

```
1  -- Índice FTS
2  SELECT rowid, body FROM messages_fts;
```

	rowid	body
1	1	Olá Elliot, teste de conversa
2	2	Olá Elliot, teste de conversa
3	3	Olá Elliot, teste de conversa
4	4	Olá Elliot, teste de conversa
5	5	Olá Elliot, teste de conversa
6	6	Olá Elliot, teste de conversa
7	7	Olá Elliot, teste de conversa
8	8	Olá Elliot, teste de conversa
9	9	Olá Elliot, teste de conversa
10	10	*NULL*
11	11	*NULL*
12	12	Olá Spider, mensagem recebida e …
13	13	Olá Spider, mensagem recebida e …
14	14	Mensagem teste eliminação local
15	15	Mensagem teste eliminação local
16	16	Mensagem teste eliminação local
17	17	Mensagem teste eliminação local
18	18	Mensagem teste eliminação local
19	19	Mensagem teste eliminação local
20	20	Mensagem teste eliminação local

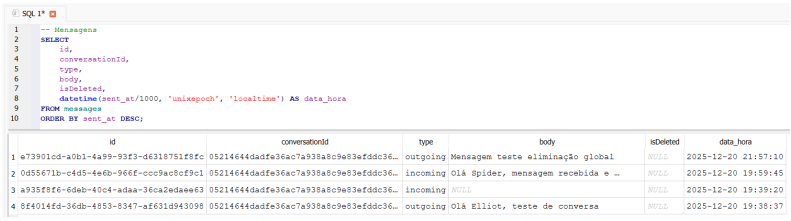
Figura B.15: Índice `messages_fts` após eliminação — conteúdo ainda presente

SQL 1*

```
1  -- Contagem
2  SELECT 'conversations' AS tabela, COUNT(*) AS total FROM conversations
3  UNION ALL
4  SELECT 'messages', COUNT(*) FROM messages
5  UNION ALL
6  SELECT 'seenMessages', COUNT(*) FROM seenMessages
7  UNION ALL
8  SELECT 'attachment_downloads', COUNT(*) FROM attachment_downloads;
```

	tabela	total
1	conversations	2
2	messages	4
3	seenMessages	10
4	attachment_downloads	0

Figura B.16: Contagem de registos antes da eliminação global



```
-- Mensagens
SELECT
    id,
    conversationId,
    type,
    body,
    isDeleted,
    datetime(sent_at/1000, 'unixepoch', 'localtime') AS data_hora
FROM messages
ORDER BY sent_at DESC;
```

	id	conversationId	type	body	isDeleted	data_hora
1	e73901cd-a0b1-4a99-93f3-d6318751f8fc	05214644dadfe36ac7a938a8c9e83efddc36…	outgoing	Mensagem teste eliminação global	NULL	2025-12-20 21:57:10
2	0d55671b-c4d5-4e6b-966f-ccc9ac8cf9c1	05214644dadfe36ac7a938a8c9e83efddc36…	incoming	Olá Spider, mensagem recebida e …	NULL	2025-12-20 19:59:45
3	a935f8f6-6deb-40c4-adaa-36ca2edaee63	05214644dadfe36ac7a938a8c9e83efddc36…	incoming	NULL	NULL	2025-12-20 19:39:20
4	8f4014fd-36db-4853-8347-af631d943098	05214644dadfe36ac7a938a8c9e83efddc36…	outgoing	Olá Elliot, teste de conversa	NULL	2025-12-20 19:38:37

Figura B.17: Tabela `messages` antes da eliminação global

SQL 1*

```
-- índice FTS
SELECT rowid, body FROM messages_fts;
```

	rowid	body
1	1	Olá Elliot, teste de conversa
2	2	Olá Elliot, teste de conversa
3	3	Olá Elliot, teste de conversa
4	4	Olá Elliot, teste de conversa
5	5	Olá Elliot, teste de conversa
6	6	Olá Elliot, teste de conversa
7	7	Olá Elliot, teste de conversa
8	8	Olá Elliot, teste de conversa
9	9	Olá Elliot, teste de conversa
10	10	NULL
11	11	NULL
12	12	Olá Spider, mensagem recebida e …
13	13	Olá Spider, mensagem recebida e …
14	14	Mensagem teste eliminação global
15	15	Mensagem teste eliminação global
16	16	Mensagem teste eliminação global
17	17	Mensagem teste eliminação global
18	18	Mensagem teste eliminação global
19	19	Mensagem teste eliminação global
20	20	Mensagem teste eliminação global
21	21	Mensagem teste eliminação global

Figura B.18: Índice `messages_fts` antes da eliminação global

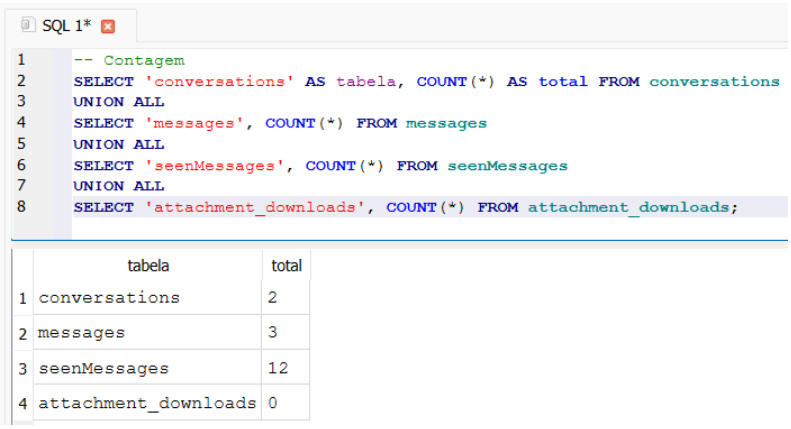


Figura B.19: Contagem de registos após eliminação global

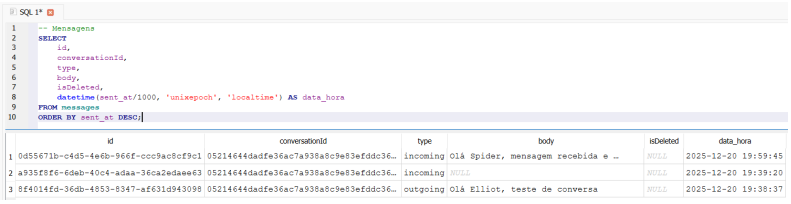


Figura B.20: Tabela `messages` após eliminação global

B.7 Cenário 5: Receção de Anexo

As seguintes figuras documentam as queries executadas após a receção de anexos (imagem e documento PDF).

B.8 Cenário 6: Receção de Áudio e Eliminação Remota

B.8.1 Cenário 6a: Receção de Mensagem de Áudio

As seguintes figuras documentam as queries executadas após a receção de uma mensagem de áudio.

B.8.2 Cenário 6b: Eliminação Remota

As seguintes figuras documentam as queries executadas após a eliminação remota da mensagem de áudio pelo utilizador "Elliot".

SQL 1*

```
1  -- índice FTS
2  SELECT rowid, body FROM messages_fts;
```

	rowid	body
1	1	Olá Elliot, teste de conversa
2	2	Olá Elliot, teste de conversa
3	3	Olá Elliot, teste de conversa
4	4	Olá Elliot, teste de conversa
5	5	Olá Elliot, teste de conversa
6	6	Olá Elliot, teste de conversa
7	7	Olá Elliot, teste de conversa
8	8	Olá Elliot, teste de conversa
9	9	Olá Elliot, teste de conversa
10	10	*NULL*
11	11	*NULL*
12	12	Olá Spider, mensagem recebida e …
13	13	Olá Spider, mensagem recebida e …
14	14	Mensagem teste eliminação global
15	15	Mensagem teste eliminação global
16	16	Mensagem teste eliminação global
17	17	Mensagem teste eliminação global
18	18	Mensagem teste eliminação global
19	19	Mensagem teste eliminação global
20	20	Mensagem teste eliminação global

Figura B.21: Índice `messages_fts` após eliminação global — conteúdo ainda presente

SQL 1*

```
1  -- 1. Contagem
2  SELECT 'conversations' AS tabela, COUNT(*) AS total FROM conversations
3  UNION ALL
4  SELECT 'messages', COUNT(*) FROM messages
5  UNION ALL
6  SELECT 'seenMessages', COUNT(*) FROM seenMessages
7  UNION ALL
8  SELECT 'attachment_downloads', COUNT(*) FROM attachment_downloads;
```

	tabela	total
1	conversations	2
2	messages	4
3	seenMessages	14
4	attachment_downloads	0

Figura B.22: Contagem de registos após envio de anexo

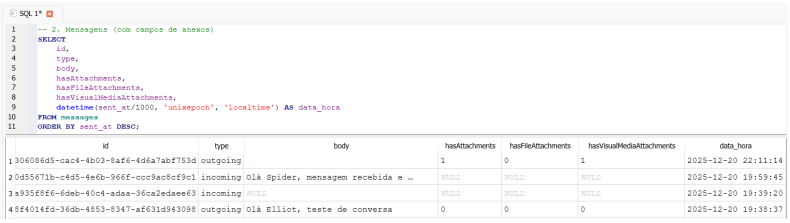


Figura B.23: Tabela `messages` com campos de anexos

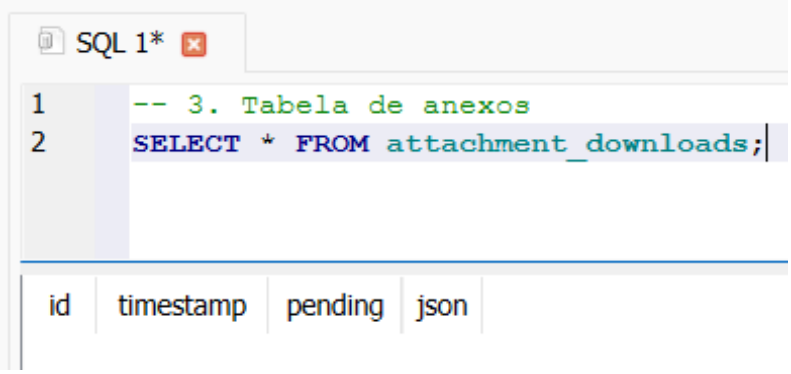


Figura B.24: Tabela `attachment_downloads` (vazia para envios)

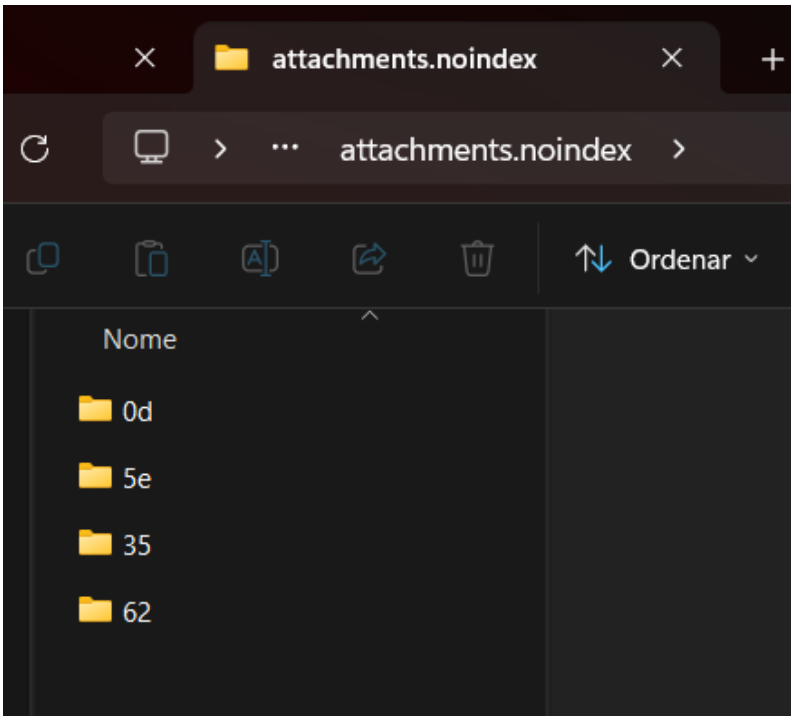


Figura B.25: Estrutura da pasta `attachments.noindex`

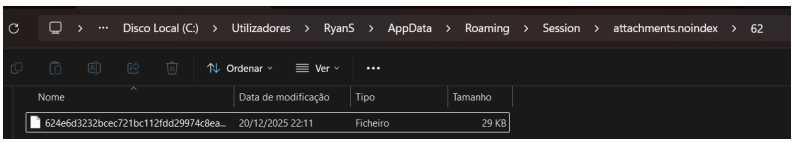


Figura B.26: Ficheiro encriptado correspondente ao anexo enviado

SQL 1*

```
1  -- Contagem
2  SELECT 'conversations' AS tabela, COUNT(*) AS total FROM conversations
3  UNION ALL
4  SELECT 'messages', COUNT(*) FROM messages
5  UNION ALL
6  SELECT 'seenMessages', COUNT(*) FROM seenMessages
7  UNION ALL
8  SELECT 'attachment_downloads', COUNT(*) FROM attachment_downloads;
```

	tabela	total
1	conversations	2
2	messages	6
3	seenMessages	17
4	attachment_downloads	0

Figura B.27: Contagem de registos após receção de anexos

SQL 1*

```
1  -- Mensagens com anexos
2  SELECT
3      id,
4      type,
5      body,
6      hasAttachments,
7      hasFileAttachments,
8      hasVisualMediaAttachments,
9      datetime(sent_at/1000, 'unixepoch', 'localtime') AS data_hora
10 FROM messages
11 ORDER BY sent_at DESC;
```

	id	type	body	hasAttachments	hasFileAttachments	hasVisualMediaAttachments	data_hora
1	242f1992-12e3-4d45-8e05-8884b091e212	incoming		1	1	0	2025-12-20 22:45:43
2	cbf652c0-f5d9-4585-a579-65a5900f1fe1	incoming		1	0	1	2025-12-20 22:39:49
3	306086d5-cac4-4b03-8af6-4d6a7abf753d	outgoing		1	0	1	2025-12-20 22:11:14
4	0d55671b-c4d5-4e6b-966f-ccc9ac8cf9c1	incoming	Olá Spider, mensagem recebida e …	NULL	NULL	NULL	2025-12-20 19:59:45
5	a935f8f6-6deb-40c4-adaa-36ca2edaee63	incoming	NULL	NULL	NULL	NULL	2025-12-20 19:39:20
6	8f4014fd-36db-4853-8347-af631d943098	outgoing	Olá Elliot, teste de conversa	0	0	0	2025-12-20 19:38:37

Figura B.28: Tabela `messages` com anexos recebidos

SQL 1*

```
1  -- Tabela de anexos (agora DEVE ter dados!)
2  SELECT * FROM attachment_downloads;
```

id	timestamp	pending	json

Figura B.29: Tabela `attachment_downloads` (permanece vazia após downloads)

SQL 1*

```sql
-- Contagem
SELECT 'conversations' AS tabela, COUNT(*) AS total FROM conversations
UNION ALL
SELECT 'messages', COUNT(*) FROM messages
UNION ALL
SELECT 'seenMessages', COUNT(*) FROM seenMessages
UNION ALL
SELECT 'attachment_downloads', COUNT(*) FROM attachment_downloads;
```

	tabela	total
1	conversations	2
2	messages	7
3	seenMessages	18
4	attachment_downloads	0

Figura B.30: Contagem de registos após receção de áudio

SQL 1*

```sql
-- Mensagens (ver campos do áudio)
SELECT
    id,
    type,
    body,
    hasAttachments,
    hasFileAttachments,
    hasVisualMediaAttachments,
    datetime(sent_at/1000, 'unixepoch', 'localtime') AS data_hora
FROM messages
ORDER BY sent_at DESC;
```

	id	type	body	hasAttachments	hasFileAttachments	hasVisualMediaAttachments	data_hora
1	ccc95d1a-abf7-4441-ac86-9a6e77acde6c	incoming		1	0	0	2025-12-20 23:02:28
2	242f1992-12e3-4d45-8e05-8884b091e212	incoming		1	1	0	2025-12-20 22:45:43
3	cbf652c0-f5d9-4585-a579-65a5900f1fe1	incoming		1	0	1	2025-12-20 22:39:49
4	306086d5-cac4-4b03-8af6-4d6a7abf753d	outgoing		1	0	1	2025-12-20 22:11:14
5	0d55671b-c4d5-4e6b-966f-ccc9ac8cf9c1	incoming	Olá Spider, mensagem recebida e …	NULL	NULL	NULL	2025-12-20 19:59:45
6	a935f8f6-6deb-40c4-adaa-36ca2edaee63	incoming	NULL	NULL	NULL	NULL	2025-12-20 19:39:20
7	8f4014fd-36db-4853-8347-af631d943098	outgoing	Olá Elliot, teste de conversa	0	0	0	2025-12-20 19:38:37

Figura B.31: Tabela `messages` mostrando a mensagem de áudio

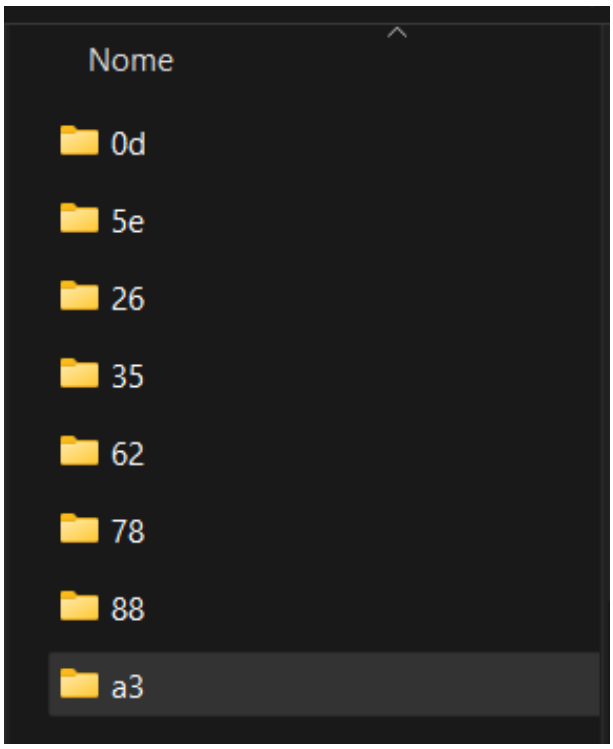


Figura B.32: Pasta `attachments.noindex` com nova subpasta `26/`

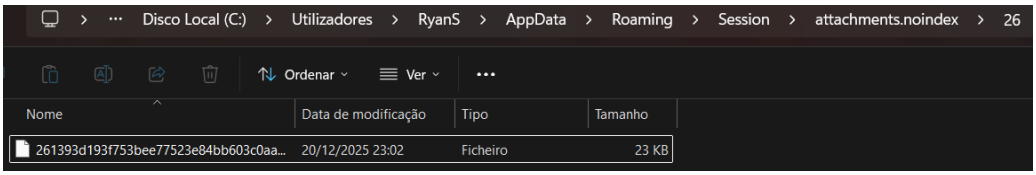


Figura B.33: Ficheiro de áudio encriptado (23 KB)

SQL 1*

```
1  -- Contagem
2  SELECT 'conversations' AS tabela, COUNT(*) AS total FROM conversations
3  UNION ALL
4  SELECT 'messages', COUNT(*) FROM messages
5  UNION ALL
6  SELECT 'seenMessages', COUNT(*) FROM seenMessages
7  UNION ALL
8  SELECT 'attachment_downloads', COUNT(*) FROM attachment_downloads;
```

	tabela	total
1	conversations	2
2	messages	7
3	seenMessages	19
4	attachment_downloads	0

Figura B.34: Contagem de registos após eliminação remota

SQL 1*

```sql
-- Mensagens
SELECT
    id,
    type,
    body,
    hasAttachments,
    hasFileAttachments,
    hasVisualMediaAttachments,
    datetime(sent_at/1000, 'unixepoch', 'localtime') AS data_hora
FROM messages
ORDER BY sent_at DESC;
```

	id	type	body	hasAttachments	hasFileAttachments	hasVisualMediaAttachments	data_hora
1	ccc95d1a-abf7-4441-ac86-9a6e77acde6c	incoming	Esta mensagem foi apagada	0	0	0	2025-12-20 23:02:28
2	242f1992-12e3-4d45-8e05-8884b091e212	incoming		1	1	0	2025-12-20 22:45:43
3	cbf652c0-f5d9-4585-a579-65a5900f1fe1	incoming		1	0	1	2025-12-20 22:39:49
4	306086d5-cac4-4b03-8af6-4d6a7abf753d	outgoing		1	0	1	2025-12-20 22:11:14
5	0d55671b-c4d5-4e6b-966f-ccc9ac8cf9c1	incoming	Olá Spider, mensagem recebida e …	NULL	NULL	NULL	2025-12-20 19:59:45
6	a935f8f6-6deb-40c4-adaa-36ca2edaee63	incoming	NULL	NULL	NULL	NULL	2025-12-20 19:39:20
7	8f4014fd-36db-4853-8347-af631d943098	outgoing	Olá Elliot, teste de conversa	0	0	0	2025-12-20 19:38:37

Figura B.35: Tabela `messages` mostrando "Esta mensagem foi apagada"

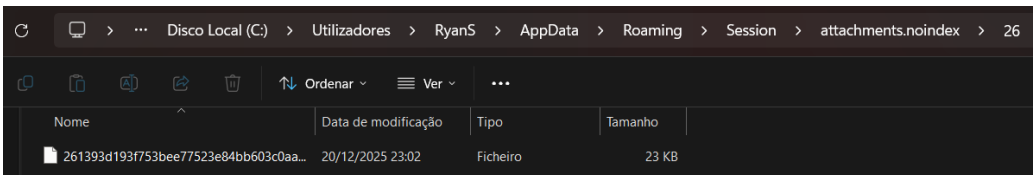


Figura B.36: Ficheiro de áudio persistente na pasta `26/` após eliminação remota