Universidade do Minho
Escola de Engenharia

Mestrado em Engenharia Informática

Ano letivo 2024/2025

Engenharia de Segurança

Cofre Digital - Trabalho Prático #1

Desenho da solução

Grupo 04

Eduardo Cunha - PG55939
João Rodrigues - PG57880

Índice

1 Introdução

Este documento surge na continuidade do documento de levantamento de requisitos. Nele é apresentada a solução final desenhada, bem como descrito o protótipo da aplicação já implementado, o qual se encontra numa fase bastante avançada de desenvolvimento.

A metodologia de trabalho adotada para o desenvolvimento da solução esteve fortemente interligada com a definição dos requisitos, uma vez que estes surgiram em parte da própria construção da solução, ao mesmo tempo que a solução foi sendo moldada com base nos requisitos identificados. Esta relação bidirecional reflete a natureza iterativa do processo de desenvolvimento.

No que diz respeito à implementação, seguimos uma metodologia prática baseada numa lista de tarefas (ToDo list), que foi sendo dinamicamente ajustada ao longo do projeto. A lista foi construída de forma orgânica, com tarefas a serem acrescentadas e concluídas de forma contínua, permitindo um acompanhamento claro do progresso e uma gestão flexível do desenvolvimento.

2 Desenho da solução

Nesta secção serão apresentadas e explicadas as soluções relativas aos principais elementos do sistema, bem como os processos das operações que um utilizador pode realizar.

2.1 Processo de Autenticação

A autenticação no sistema será baseada num modelo seguro, garantindo que apenas utilizadores legítimos possam aceder aos seus cofres digitais. O processo será dividido em três áreas principais: autenticação base, gestão de sessões e proteção contra ataques.

Autenticação Base

Para garantir um nível elevado de segurança, a autenticação será feita através de nome de utilizador e palavra-passe. As palavras-passe não serão armazenadas diretamente no servidor, mas sim os seus hashes, utilizando o algoritmo *bcrypt*. Desta forma, mesmo que os dados sejam comprometidos, não será possível recuperar as palavras-passe originais.

Gestão de Sessões

Após uma autenticação bem-sucedida, será gerado um token de sessão único para o utilizador. Este token terá um tempo de expiração, garantindo que o acesso não seja permanente. Quando o token expirar, será exigida uma nova autenticação para continuar a utilizar o sistema.

Proteção Contra Ataques

Para mitigar os riscos de segurança, serão implementadas diversas medidas de proteção contra ataques. Estas incluem a limitação do número de tentativas de login, a introdução de atrasos progressivos após falhas consecutivas, dificultando ataques de força bruta, e a validação rigorosa dos dados introduzidos, de forma a prevenir ataques como a injeção de código malicioso.

2.2 Dados

Client-side

Para garantir a máxima segurança dos dados armazenados no Cofre Digital, a cifragem será realizada no lado do cliente antes de os ficheiros serem enviados para o servidor. Desta forma, o servidor nunca terá acesso ao conteúdo original dos ficheiros.

Cada ficheiro será cifrado individualmente, utilizando uma chave única gerada especificamente para esse ficheiro. Assim, mesmo que um atacante consiga aceder a um ficheiro cifrado, não poderá utilizá--lo para comprometer outros ficheiros armazenados no sistema. A geração das chaves necessárias será feita no cliente, através de um mecanismo de derivação de chaves seguro. Para a cifra dos ficheiros foi utilizado o algoritmo "AES" no modo "GCM"

Para derivar as chaves de cifragem dos ficheiros, será utilizado o algoritmo Argon2. No entanto, as chaves de ficheiro nunca serão enviadas para o servidor em formato legível. Antes de qualquer transmissão, serão sempre cifradas com a chave pública do utilizador (que se encontra associada a si na base de dados) de modo a impedir que o servidor tenha acesso direto a elas.

Server-side

No lado do servidor, será necessário armazenar ficheiros que já tenham sido cifrados pelo cliente, bem como a chave pública associada a cada utilizador.

Quando um ficheiro for inicialmente armazenado pelo proprietário, a chave única desse ficheiro será cifrada com a chave pública do proprietário.

Para o armazenamento de palavras-passe, será utilizado o "bcrypt". Apenas as chaves de ficheiros cifradas serão armazenadas, nunca em *plaintext*. Numa implementação futura, poderia ser considerado trocar para Argon2, devido à sua maior segurança, flexibilidade e por ser um dos algoritmos mais recomendados atualmente para proteção de palavras-passe.

Com esta arquitetura do servidor, garantimos a "falta de confiança" no mesmo. Ele apenas guardará conteúdo dos ficheiros cifrado, e garantirá o encaminhamento de ficheiros e chaves para os clientes certos. Para além disto, será o servidor a verificar as permissões dos utilizadores relativas aos ficheiros (as permissões são previamente definidas pelo dono do ficheiro).

2.3 Comunicação

Para assegurar uma comunicação segura entre cliente e servidor, todas as interações utilizam o protocolo TLS 1.3, com autenticação mútua baseada em certificados digitais, recurso a algoritmos de cifra robustos e desativação de versões obsoletas dos protocolos. As mensagens trocadas seguem o formato JSON. Todos os dados de entrada são rigorosamente validados antes do processamento, tanto do lado do cliente como, sobretudo, do lado do servidor.

As mensagens de erro adotam um formato uniforme e contêm apenas a informação estritamente necessária, de forma a evitar a exposição de detalhes internos. É aplicada uma política de limitação de taxa por utilizador e por endereço IP, bem como definidos timeouts de ligação para prevenir o consumo excessivo de recursos. Após 10 minutos de sessão ativa, é requerida nova autenticação, dado que o token expira, impedindo a sua reutilização por terceiros não autorizados.

Todas as transferências de dados incluem verificação através de checksums criptográficos e validação da integridade dos ficheiros. Além disso, é implementada a assinatura de mensagens para garantir a sua autenticidade.

2.4 Auditoria

Todas as ações realizadas são registadas com timestamp, identificação do utilizador (email), recurso afetado e tipo de alteração (escrita ou adição), sendo registada qualquer ação que um utilizador possa executar.

Para garantir a integridade dos registos, são utilizadas assinaturas digitais e os logs são armazenados em formato apenas de adição (append-only), impedindo alterações retroativas. São realizados backups

regulares para um local seguro, e todo o conteúdo dos logs é cifrado. O acesso aos registos de auditoria é estritamente controlado, estando os logs guardados num ficheiro cifrado, acessível apenas ao servidor e armazenado num caminho oculto no código, evitando assim a sua presença na base de dados da aplicação. Esta abordagem assegura que, mesmo em caso de comprometimento da base de dados, os logs mantêm a sua validade, uma vez que permanecem isolados e protegidos contra manipulação.

2.5 Controlo de Acessos

O servidor irá abrir uma thread dedicada para tratar cada cliente, aumentando a escalabilidade e melhorando o desempenho geral do sistema. Para garantir um acesso controlado e ordenado a recursos partilhados, são utilizados locks em pontos críticos como a manipulação de logs, gestão de sessões e controlo de tentativas de login. Desta forma, evitam-se problemas como *race conditions*, inconsistências de dados e comportamentos inesperados durante o acesso concorrente.

2.6 Permissões

No sistema de segurança implementado, cada recurso possui um único proprietário (Owner), que detém o controlo absoluto sobre o ficheiro. Este Owner é estabelecido automaticamente durante a criação do recurso.

As permissões são atribuídas inicialmente pelo Owner, que determina que utilizadores podem interagir com o recurso e de que forma. O sistema implementa uma hierarquia clara de permissões que estabelece três níveis progressivos de acesso:

- O nível mais básico é a permissão de **Read**, que concede ao utilizador apenas a capacidade de visualizar o conteúdo do ficheiro, sem poder alterá-lo de qualquer forma.
- Um patamar intermédio é representado pela permissão de **Append**, que permite ao utilizador não só ler o ficheiro, mas também adicionar novo conteúdo ao final do documento existente, preservando a informação original.
- O nível mais elevado é a permissão de **Write**, que confere ao utilizador a capacidade de ler e modificar integralmente o conteúdo do ficheiro, incluindo alterar ou eliminar partes existentes.

Esta estrutura hierárquica de permissões proporciona um controlo granular sobre o acesso aos recursos, garantindo que os utilizadores tenham apenas as capacidades necessárias para desempenhar as suas funções, sem comprometer a segurança ou integridade dos dados.

Outra medida implementada consiste em garantir que, caso um utilizador opte por eliminar a sua conta, todas as suas permissões sejam revogadas, bem como todos os ficheiros e pastas dos quais é proprietário sejam eliminados. Adicionalmente, são removidas todas as permissões que outros utilizadores possam ter sobre esses mesmos conteúdos. Naturalmente, todos os dados associados ao utilizador, como o e-mail, ID, palavra-passe e o respetivo cofre, serão também apagados de forma definitiva.

2.6.1 Read

Quando um utilizador solicita a leitura de um determinado ficheiro, o processo inicia-se com o envio de um pedido ao servidor contendo o identificador do ficheiro pretendido e as credenciais do utilizador. O servidor, por sua vez, verifica as permissões de acesso, assegurando que o utilizador tenha permissão de "read" ou superior.

Após confirmar as permissões, o servidor envia um pacote de dados cifrados que inclui o conteúdo cifrado, um "iv", uma tag de autenticação e uma chave cifrada específica para aquele utilizador.

Ao receber estes elementos, o cliente procede a um processo de decifragem em várias etapas. Primeiro, utiliza a sua chave privada para decifrar a chave única do ficheiro. De seguida, emprega esta chave,

juntamente com o "iv", para decifrar o conteúdo do ficheiro utilizando o algoritmo AES no modo GCM. Durante este processo, a tag de autenticação permite verificar a integridade dos dados, alertando caso o ficheiro tenha sido adulterado.

Após a decifragem bem-sucedida, o conteúdo é apresentado ao utilizador através da ferramenta "less", que oferece uma interface de leitura sem possibilidade de edição, garantindo uma experiência controlada de acesso à informação.

2.6.2 Write

O processo inicia-se quando o utilizador solicita a modificação de um determinado ficheiro. O cliente envia um pedido ao servidor contendo o identificador do ficheiro e as suas credenciais . O servidor, antes de qualquer operação, verifica se o utilizador possui permissões de escrita para o ficheiro solicitado.

Após confirmação positiva das permissões, o servidor disponibiliza um pacote de dados que inclui o conteúdo cifrado atual, o "iv", a tag e a chave cifrada específica para o utilizador.

Ao receber estes elementos, o cliente executa um processo de decifragem semelhante ao do processo de "read"(explicado anteriormente).

Uma vez decifrado, o conteúdo é apresentado ao utilizador através da ferramenta "vipe", que proporciona uma interface de edição em terminal. Esta ferramenta permite que o utilizador modifique o texto conforme necessário, mantendo um ambiente controlado para a edição.

Após a conclusão das alterações, o cliente cifra o conteúdo modificado utilizando a mesma chave do ficheiro, mas gerando um novo "iv" e uma nova tag. Este novo conjunto de dados cifrados é então enviado de volta ao servidor.

O servidor, por sua vez, verifica novamente as permissões de escrita do utilizador e, caso estejam em conformidade, atualiza os registos na base de dados com o novo conteúdo.

2.6.3 Append

O procedimento inicia-se de forma semelhante às outras operações, com o envio de um pedido ao servidor contendo o identificador do ficheiro e as credenciais do utilizador. O servidor, antes de prosseguir, verifica se o utilizador possui permissão de "append" para o ficheiro solicitado.

Após a validação positiva das permissão, o servidor disponibiliza o mesmo pacote de dados criptográficos que na operação de modificação: o conteúdo cifrado atual, o "iv", a tag e a chave cifrada específica para o utilizador.

No lado do cliente, o processo de decifragem segue os mesmos passos das operações anteriores, utilizando a chave privada do utilizador para decifrar a chave única do ficheiro e, subsequentemente, o conteúdo do documento.

A diferença fundamental ocorre na fase de edição. Em vez de apresentar o conteúdo existente para modificação, o sistema abre um editor vazio através da ferramenta "vipe", onde o utilizador pode introduzir apenas o novo conteúdo a ser acrescentado.

Após a introdução do novo conteúdo, o sistema concatena o texto original com o texto adicional, criando um documento único e coeso. Este conteúdo completo é então submetido ao processo de cifragem, gerando um novo conjunto de dados cifrados, "iv" e tag.

Estes elementos são enviados de volta ao servidor, que verifica novamente as permissões de "append" do utilizador antes de atualizar os registos na base de dados com o novo conteúdo cifrado.

No protótipo desta fase, a verificação relativa à operação ser "append" e não "write" reside no facto de o utilizador não acessar o conteúdo já existente (apenas escreve novo conteúdo a ser concatenado). No entanto, numa implementação futura, seria ideal realizar algo como uma operação robusta *server-side* que verificasse que ocorreu realmente um "append" e não um "write".

2.7 Hierarquia de pastas

No sistema de organização de ficheiros implementado, existe uma sofisticada hierarquia de pastas que incorpora mecanismos de herança de permissões. Tal como acontece com os ficheiros individuais, cada pasta possui um único proprietário (Owner) que detém controlo administrativo sobre esse elemento específico.

A característica distintiva deste sistema é a propagação vertical de permissões, onde os privilégios de acesso definidos numa pasta são automaticamente transmitidos a todos os ficheiros e subpastas contidos na sua estrutura. Esta cascata de permissões simplifica significativamente a gestão de acessos em estruturas complexas, eliminando a necessidade de configurar individualmente cada elemento.

Quando ocorre a movimentação de um ficheiro entre diferentes pastas, o sistema implementa uma importante regra de adaptação: o ficheiro desvincula-se das permissões da sua localização original e herda automaticamente o conjunto de permissões da nova pasta de destino. Este comportamento assegura que os ficheiros mantêm sempre uma configuração de segurança coerente com o seu contexto atual. O mesmo acontecerá no caso de movimento de pastas.

2.8 Processo de partilha de ficheiro

Para a partilha de um ficheiro, será utilizada a cifragem por envelope para partilhar as chaves do ficheiro entre utilizadores. A chave do ficheiro será cifrada com a chave pública do utilizador a quem se pretende dar permissão de acesso ao ficheiro.

Armazenamento de chaves de ficheiros
Se A for o proprietário (owner), inicialmente estará no servidor a chave única do ficheiro cifrada com a chave pública de A. Quando A partilhar esse ficheiro com B, passará a estar também no servidor a chave única do ficheiro cifrada com a chave pública de B.

No diagrama seguinte, é exemplificado o fluxo da partilha de um ficheiro do cliente A com o cliente B, utilizando o método de cifragem por envelope.

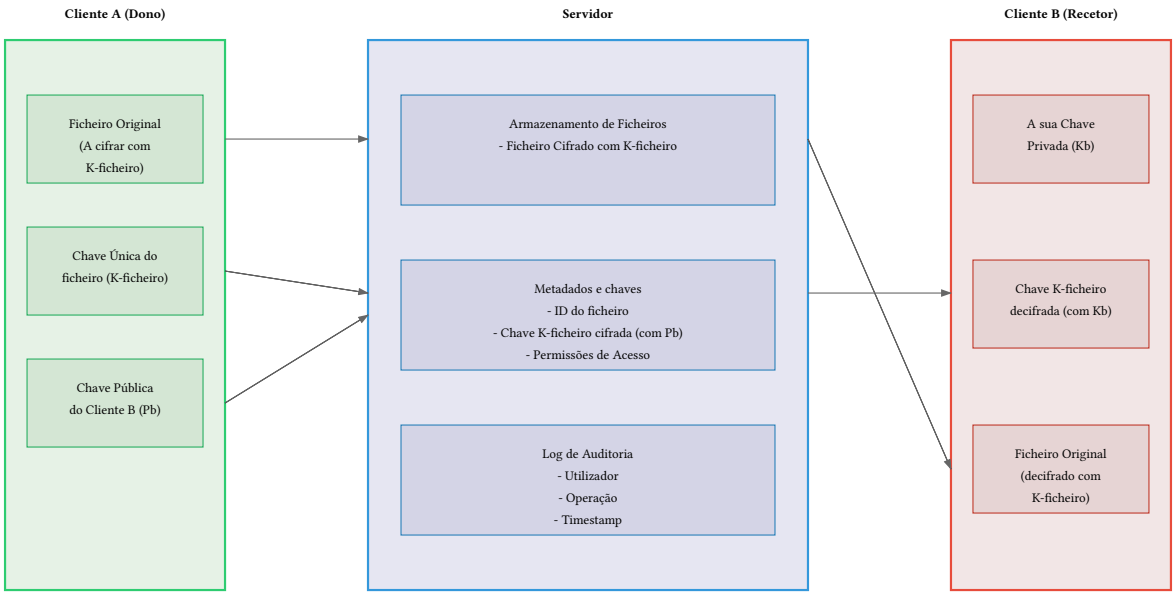

Figura 2: Partilha de um ficheiro do cliente A com o cliente B

3 Prototipagem da solução

A prototipagem da solução foi desenvolvida integralmente em Python, aproveitando um conjunto cuidadosamente selecionado de bibliotecas especializadas que garantem a segurança, eficiência e robustez do sistema.

3.1 Cliente

No lado do cliente, criou-se uma aplicação de terminal interativa organizada em três menus principais. O menu de autenticação apresenta uma interface limpa com validação em tempo real, incluindo processos de login com verificação de credenciais, registo de novos utilizadores (com geração automática de par de chaves RSA). O menu principal permite a navegação hierárquica pelo cofre digital, com visualização intuitiva de pastas e ficheiros. O menu de pastas oferece operações completas de criação, leitura, atualização e eliminação de ficheiros e pastas, incluindo uma interface de partilha com seleção granular de permissões, visualizador/editor integrado (através das ferramentas less e vipe), e funcionalidades de upload.

3.2 Servidor

O servidor foi concebido como um sistema concorrente baseado em threads, com um socket principal em modo listening contínuo e um thread pool para gestão eficiente de múltiplos clientes simultâneos. Implementa uma fila prioritária para alocação inteligente de recursos e mecanismos de timeout automático para conexões inativas. A sua arquitetura divide-se em quatro módulos funcionais principais: o gestor de autenticação, responsável pela validação de credenciais, emissão/revogação de tokens e proteção contra ataques de força bruta; o gestor de ficheiros, que trata do armazenamento seguro em filesystem, verificação de permissões e execução de operações atómicas com capacidade de rollback; o gestor de metadados, que serve de interface com a base de dados e mantém a hierarquia de pastas e políticas de acesso; e o serviço de auditoria, dedicado ao registo cifrado de eventos e manutenção de logs seguros. O servidor limitanda-se a registar no terminal apenas as conexões e desconexões, enquanto todos os erros e informações sensíveis são guardados em logs cifrados, sem exposição de dados críticos em mensagens de erro.

3.3 Base de dados

A aplicação utiliza uma base de dados PostgreSQL, escolhida pela sua robustez, desempenho e suporte avançado a tipos de dados e constraints. Para a interação entre o servidor e a base de dados, foi utilizada a biblioteca psycopg2, que é thread-safe e permite a execução segura de queries com validação de parâmetros, prevenindo injeções SQL e garantindo a integridade das operações.

A base de dados é automaticamente criada e inicializada com um conjunto de tabelas estruturadas para representar utilizadores, cofres, pastas, ficheiros, permissões e chaves cifradas, com uso de UUIDs como identificadores únicos e constraints como chaves estrangeiras e validações de formato para garantir consistência. Foi também criado um tipo enumerado para definir os níveis de acesso, garantindo um controlo rigoroso sobre as permissões.

Todas as tabelas são associadas a um utilizador administrador (adminES) com permissões completas, sendo os dados sensíveis protegidos com *hashing* e os acessos controlados.

3.4 Modelo de segurança

No seguinte diagrama é apresentado um modelo que enumera as diversas caraterísticas de segurança do protótipo realizado.

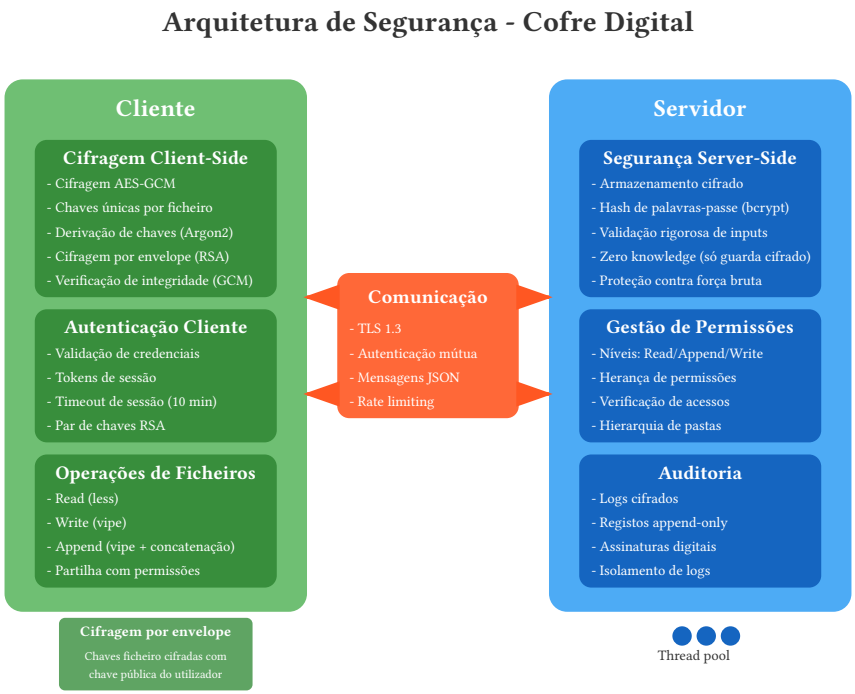

Figura 3: Modelo de segurança

4 Conclusão final

Neste trabalho foi estudada, proposta e (praticamente) desenvolvida uma aplicação de cofre digital seguro, com foco especial na segurança desde a sua conceção até à sua implementação. Esta conclusão é elaborada com base na visão global do projeto e após a finalização dos documentos que o compõem.

Consideramos ter atingido todos os objetivos inicialmente propostos, tanto no que diz respeito ao estudo do caso como à criação de uma solução adequada e segura. Fomos além do exigido, apresentando um protótipo funcional bastante completo, já com diversos mecanismos de segurança integrados. Apesar de alguns entraves iniciais, decorrentes de diferentes interpretações sobre os requisitos e o rumo a seguir, conseguimos consolidar uma solução sólida e bem estruturada.

Este trabalho revelou-se fundamental para o desenvolvimento das nossas competências na área da segurança de aplicações, nomeadamente na identificação de ameaças, análise de risco, definição de requisitos e boas práticas de desenvolvimento seguro. Foi, no entanto, na fase de prototipagem e implementação que mais aprendemos, ao aplicar na prática os conhecimentos adquiridos e aprofundar a nossa compreensão sobre a sua concretização real.

5 Referências

- PostgreSQL Global Development Group. PostgreSQL Documentation. Disponível em: https://www.postgresql.org/docs/
- psycopg. Psycopg Documentation. Disponível em: https://www.psycopg.org/docs/
- Internet Engineering Task Force (IETF). Argon2 Password Hashing Scheme – Draft Specification (draft-irtf-cfrg-argon2-13). Disponível em: https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-argon2-13
- Python Cryptographic Authority (PyCA). bcrypt Python Library – GitHub Repository. Disponível em: https://github.com/pyca/bcrypt/