Segurança Informática em Redes e Sistemas

Alameda

Grupo 43

87848, Daniel Oliveira, danielapoliveira@tecnico.ulisboa.pt

91122, Pedro Cipriano, pedrofcipriano@tecnico.ulisboa.pt

**Remote Document Access**

Conteúdo

[Problema 3](#_Toc532308751)

[Requisitos 3](#_Toc532308752)

[Solução 3](#_Toc532308753)

[Básico 3](#_Toc532308754)

[Intermédio 4](#_Toc532308755)

[Avançado 5](#_Toc532308756)

[Resultados 5](#_Toc532308757)

[Avaliação 5](#_Toc532308758)

[Conclusão 6](#_Toc532308759)

[Referências 6](#_Toc532308760)

[Anexos 7](#_Toc532308761)

# Problema

(Given the chosen scenario, why is security necessary? What is the main problem solved? Use around 200 words)

Pretende-se criar um sistema que permita ler, editar e partilhar ficheiros com outros utilizadores (para leitura e/ou escrita). Estes ficheiros deveram ser armazenados num servidor remoto. Este sistema deve disponibilizar as funcionalidades de listar ficheiros, ler e escrever nos ficheiros armazenados, criar ficheiros e fazer upload e download de ficheiros. Para acesso ao sistema deverá ser disponibilizado uma aplicação cliente.

Os ficheiros armazenados pelo servidor só devem poder ser acedidos (para leitura/escrita) por utilizador autorizadas pelo seu dono. Caso alguma entidade tenha acesso ao local de armazenamento destes ficheiros, deseja-se ainda garantir que esta não seja capaz de ler o conteúdo dos ficheiros lá armazenados (confidencialidade), assim como caso esta entidade altere o seu conteúdo seja possível detetar esta adulteração (integridade). Quer-se ainda garantir que um possível adversário não é capaz de espiar ou adulterar as mensagens trocadas entre o servidor e os seus clientes. Para além disso, deseja-se ainda tentar ser resistente a ataques de ransomware.

# Requisitos

(Which security requirements were identified for the solution? Present as list)

* Os documentos não podem ser lidos por pessoas não autorizadas (confidencialidade);
* Os documentos não podem ser alterados por pessoas não autorizadas (integridade);
* Apenas o dono do documento pode alterar as permissões dos utilizadores com acesso ao documento, assim como eliminá-lo;
* O sistema deve tentar ser resistente a ataques de ransomware;

# Solução

(overview with diagram and explanation with around 200 words or less. Describe basic, intermediate, and advanced versions of the solution. Be explicit about keys and how they will be distributed.)

TODO

## Básico

Desenvolvimento de um sistema de partilha de ficheiros com autenticação baseada no par *username* e *password* e autorização baseada nessa autenticação. O sistema deverá ainda comunicar sobre HTTP (*Hypertext Transfer Protocol*) e disponibilizar as seguintes funcionalidades:

* Registo de utilizadores;
* Criação de ficheiros;
* Eliminação de ficheiros;
* Listagem dos ficheiros acessíveis pelo utilizador;
* Leitura e escrita em ficheiros;
* Download e upload de ficheiros;
* Gestão de permissões (leitura/escrita) doutros utilizadores sobre ficheiros;

Toda a informação armazenada pelo servidor deverá ser armazenada numa base de dados.

Desenvolvimento de uma aplicação cliente para acesso ao sistema de partilha de ficheiros.

## Intermédio

Baseado na versão básica que apenas fornece como segurança a autenticação dos utilizadores e a sua autorização no acesso aos ficheiros, introduzir mecanismos de segurança para melhorar a segurança do sistema contra outros ataques.

De forma a proteger a comunicação de ser espiada e/ou adulterada por uma terceira entidade que não o cliente e o servidor, pretende-se substituir a utilização de HTTP por HTTPS (*Hypertext Transfer Protocol Secure*), o qual estende o HTTP com TLS (*Transport Layer Security*), garantindo assim estas duas propriedades. Adicionalmente, isto ainda evita ataques de *man-in-the-middle*, devido a garantir a autenticação do servidor para com o cliente. Para se poder tirar proveito deste protocolo é necessário a geração um par de chaves Pública e Privada para o servidor. Por sua vez, para que a chave pública do servidor possa ser distribuída pelos clientes, deve-se obter um certificado para essa chave, o qual deve ser emitido por uma entidade certificadora.

Para proteger a *password* dos utilizadores de alguma entidade com acesso à base de dados, onde estas estão armazenadas, ao invés de armazenar a *password* passamos a armazenar um *hash* criptográfico da password com um *salt*. Este *salt* serve para evitar ataques de dicionário contra o *hash* armazenado, utilizando *rainbow tables*. O *salt* deverá ser diferente para cada utilizador e armazenado juntamente ao *hash*. Neste caso, será utilizado o argon2 para fazer *password hashing*.

O servidor deverá possuir uma chave mestre (*KEK*) para encriptar outras chaves, nomeadamente as chaves para encriptação dos ficheiros (referida em seguida) e para geração dos MACs (referidas abaixo). Esta chave mestre deverá ser uma chave simétrica, neste caso AES de 256 bits, de forma a garantir a maior proteção possível a estas chaves. A chave mestre por sua vez terá de ser armazenada *Key Store*, o qual a encripta e apenas permite aceder-lhe quem possuir uma frase chave.

De modo a evitar que uma entidade com acesso à base de dados possa ter acesso ao conteúdo dos ficheiros armazenados, estes são encriptados com uma chave simétrica AES de 128 bits em modo CBC (Cipher Block Chaining). A qual é também armazenada na base de dados e por sua vez encriptada com a chave mestre do servidor.

De forma a proteger a integridade das entidades guardadas na base de dados, evitando que alguém com acesso à mesma as modifique, estas terão um MAC associado às mesmas. Este MAC é gerado utilizando HMAC com SHA-256 e uma chave de 128 bits, neste caso a mesma usada na encriptação com o AES de 128 bits. No caso dos ficheiros, a chave utilizada para o MAC do seu conteúdo e outras informações (*metadata*) é a mesma que a utilizada para a encriptação do seu conteúdo.

É digno de nota que a solução apresentada não almeja ser resistente a um servidor malicioso e por conseguinte que as chaves por este possuídas sejam obtidas por um atacante.

## Avançado

Com o objetivo de permitir à solução intermédia recuperar de alguma adulteração detetada, deverão ser guardadas cópias das várias versões do ficheiro, permitindo assim que o utilizador possa voltar a uma versão anterior não adulterada e assim tentar recuperar do ataque.

Para além deste versionamento, deverá ainda feita a replicação dos servidores de base de dados em redundância geográfica e em redes diferentes para que possam ser recuperados em caso de um ataque de ransomeware a um dos locais de armazenamento ou à rede.

# Resultados

(What was actually implemented?)

A aplicação cliente proposta concretiza-se numa aplicação de linha de comandos com todos os comandos mencionados, tal como proposto, e ainda numa aplicação web que foi desenvolvida com o propósito de ser mais fácil testar o sistema de partilha de ficheiros. Esta aplicação web possui a maioria das funcionalidades com exceção do download e upload de ficheiros.

Não nos sendo possível a implementação e utilização de todas as características do sistema descritas na solução proposta, optamos pela melhor aproximação possível à mesma. Sendo que as exceções à solução proposta se encontram em seguida.

A base de dados utilizada é uma base de dados SQLite, por simplificação, tendo em conta que a utilização de uma opção real não daria uma melhoria em segurança.

Relativamente à obtenção de um certificado para a chave pública do servidor, emitido por uma entidade certificadora, optou-se por gerar um certificado auto assinado para desempenhar o papel de entidade certificadora (CA) e este assinar o nosso certificado. Para mais informação de como foram gerados os certificados, verificar o anexo Como gerar certificados para a CA e o Servidor. De forma a poder utilizar estes certificados e estes serem considerados válidos, o utilizador deverá adicionar o certificado da nossa CA aos seus certificados de Autoridades de certificação de raiz fidedigna.

Embora na solução proposta a chave mestre seja uma chave AES de 256 bits, na implementação é uma chave assimétrica de RSA de 8192 bits, uma vez que durante a implementação seguimos o descrito na proposta. No entanto, após a implementação concluiu-se que seria mais apropriado esta solução, uma vez que as chaves Pública e Privada nunca seriam utilizadas por outro que não o servidor, havendo, no entanto, um maior peso computacional na criptografia assimétrica, sem outras vantagens.

Referentemente ao versionamento dos ficheiros e replicação da base de dados, optou-se por fazer apenas fazer uma replicação simples da base de dados SQLite como prova de conceito.

# Diferenças entre a solução e a proposta

# Avaliação

(How do you evaluate your solution? Mention strengths and weaknesses. Justify implementation choices.)

# Conclusão

(Ending statements in 1 or 2 paragraphs)

Baseado nos cenários de ataque que considerámos, pode-se considerar que cobrimos os vetores de ataque dos mesmos e por isso conseguimos garantir a segurança do nosso sistema, nomeadamente a sua confidencialidade e integridade.

# Referências

(short list of essential tools and publications with a brief description of how it was used for the project)

* Django
* Argon2

# Anexos

## Como gerar certificados para a CA e o Servidor