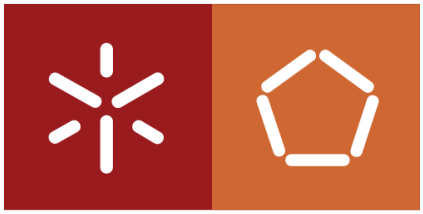

Universidade do Minho
Escola de Engenharia

Segurança de Sistemas Informáticos
TP3 - Autorização de Operações ao nível do Sistema de Ficheiros

Gabriela Martins A81987
Rui Costa A79947
João Vilaça A82339

Janeiro 2020

Conteúdo

1 Introdução

Nos tempos que correm, cada vez mais valorizamos e protegemos a informação que guardamos nos sistemas informáticos, principalmente nos que fazem parte de organizações visto que constituem uma parte fundamental do negócio. Com esta valorização da informação, cresce também o interesse na obtenção dela por parte de atacantes. Assim, medidas de protecção são fundamentais para aumentar a segurança e evitar eventuais riscos.

Com isto, neste trabalho pretendemos reforçar o nível segurança dos mecanismos de controlo de acesso de um sistema de ficheiros tradicional implementando assim um mecanismo adicional de controlo de acesso a ficheiros baseado em *libfuse* enquadrando-se numa politica de segurança moderna, a *Multi-Factor Authentication* que consiste em utilizar mais que um processo para autenticar o utilizador.

Implementou-se, por isso, um sistema que gera um código aleatório, associando-o com um *QR code*. O utilizador tem que validar os dois métodos, tendo 30 segundos para introduzir o código de acesso que lhe permite aceder ao ficheiro.

2 Modelo do Sistema

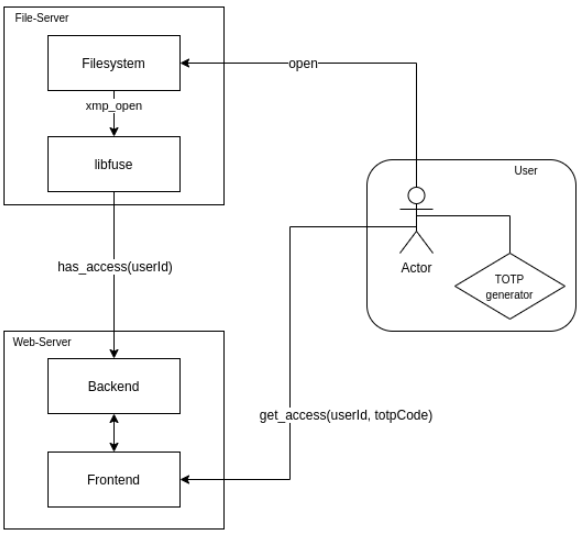

Figura 1: Modelo de Dados

3 Implementação

3.1 Arquitetura

Tendo em conta os objetivos do sistema de ficheiros a implementar, tal como sugerido, teve-se por base a biblioteca *libfuse*.

Foram implementadas algumas alterações ao *xmp_ open* original.

Estas consistem num conjunto de medidas de modo a verificar as permissões a que o utilizador tem acesso. Nesta função é chamado o método *auth_ verification*, que comunica com o servidor de modo a verificar se o utilizador tem permissões para aceder ao ficheiro pedido.

3.2 Verificação do Utilizador

No sistema implementado, utilizou-se um ficheiro de texto de modo a guardar todos os utilizadores permitidos no sistema e respetivos segredos. Assim, quando o programa inicia, a primeira verificação é se o utilizador logado esta no sistema de ficheiros. Em caso afirmativo, o programa trata o segredo associado ao utilizador, para verificar a autenticação de dois fatores. Caso o utilizador nao seja encontrado no ficheiro, o programa mostra a seguinte mensagem: *Access denied, enter your 2-factor authentication at http://localhost:8000/100.*

3.3 Geração do Segredo

O segredo é a chave associada ao user usada para validar a autenticação de dois factores. Para isso usamos um método incluído na biblioteca do *Google Authenticator*, o *create_secret()*, que gera uma sequência única tanto de números como letras. Para o segundo factor de autenticação optamos por gerar um código QR a partir do segredo obtido anteriormente.

3.4 Autenticação do utilizador

Para um utilizador ter acesso ao ficheiro tem que cumprir três requisitos:

- O seu ID tem que estar no ficheiro *users.txt*, de modo a informar que o *user* tem permissão para aceder ao ficheiro;
- Tem que ler o código QR com o auxilio da aplicação do *Google Authenticator* e obter um código de acesso;
- Tem que aceder á página de inserção da chave e inserir o código obtido com a leitura do código QR. Se a chave introduzida, depois da conversão para o respectivo segredo pelo método *verify_code* da livraria do *Google Authenticator*, corresponder ao segredo desse *user*, a permissão de aceder ao ficheiro é-lhe concedida.

3.5 Janela de inserção da chave

Visto que teria de ser introduzida uma chave, criamos uma simples página com o auxilio de *html* e *java script*, de modo a conseguir criar o *design* pretendido e tratar todos os pedidos que lidam com elementos externos. Foi também criada uma página com o intuito de consultar um segredo e um código QR gerados, tendo assim dois *end points*, o *http://localhost:8000/get_access* e o *http://localhost:8000/generate_code.*

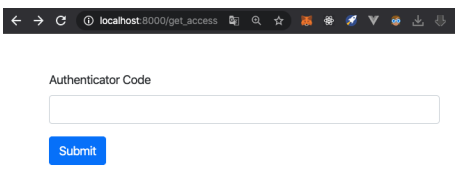

Figura 2: Página de inserção do código

Figura 3: Página de consulta da chave gerada

3.6 Aspetos de Segurança

	Filesystem	libfuse	Backend	Frontend	Users
Spoofing	X				X
Tampering	X		X		
Repudiation	X				
Information disclosure	X				
Denial of Service			X	X	
Elevation of Privilege	X		X		

Existem alguns aspetos que merecem especial atenção.

Em primeiro lugar, é necessário uma alta proteção ao ficheiro *users.txt*. Este ficheiro é um forte vetor de ataque porque, se acedido por um atacante, ele poderia ter acesso a todos os ficheiros.

É também importante referir que o acesso a uma conta de utilizador não permite a abertura dos ficheiros, uma vez que também é necessário possuir o segredo.

Como podemos ver na tabela acima, o *Filesystem* tem muitas vulnerabilidades que, fundamentalmente, são resolvidas com o aumento do controlo de acesso aos ficheiros constituído pela *Two Factor Authentication*.

Na nossa aplicação o *backend* constitui a gestão da lista de controlo de acesso e, por isto, pode sofrer de *Tampering*, se o atacante modificar o ficheiro da lista das permissões. Pode também sofrer de *Denial Of Service* se for feito um elevado número de pedidos simultaneamente. Por fim, pode ser vítima de *Elevation of Previlege* se o atacante tiver acesso ao ficheiro de controlo de permissões e mudar a permissão associada ao seu ID.

3.7 Dependências e Compilação

De forma a automatizar este processo de instalação, foi utilizado o *Docker*. Foram criados dois *containers*, um para o *filesystem* com *libefuse*, outro para o *webserver*. Assim, é apenas necessário executar o *makefile*.

4 Conclusão

A unidade curricular de SSI permitiu um aprofundamento dos conhecimentos dos problemas mais comuns de programação. Tendo isto em conta, este trabalho foi desenvolvido de forma mais consciente, levando sempre em consideração proteger o sistema e não o deixar exposto.

Foi também possível conhecer melhor a biblioteca *libfuse*, bem como as funcionalidades que disponibiliza.

Conclui-se assim que o trabalho desenvolvido de forma positiva, correspondendo ao necessário.