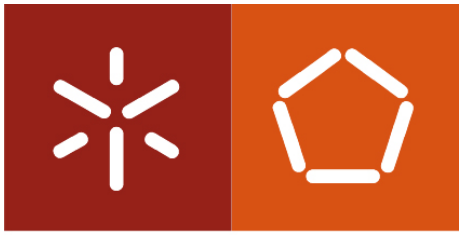

Universidade do Minho

Segurança de Sistemas Informáticos

Trabalho Prático 3

Grupo 7

André Guilherme Nunes Viveiros, A80524

Luís José Rodrigues da Silva Macedo, A80494

Janeiro 2020

Índice

1. Introdução

Este relatório documenta o trabalho desenvolvido no âmbito da Unidade Curricular de **Segurança de Sistemas Informáticos**, do curso de Mestrado em Engenharia Informática da Universidade do Minho, no ano letivo de 2019/2020.

1.1. Contextualização e caso de estudo

O **FUSE** (**Filesystem in Userspace**) é uma interface para os programas que correm no *userspace* exportarem um sistema de arquivos para o **kernel** do **Linux**. O projeto **FUSE** consiste em dois componentes: o módulo kernel do **FUSE** e a biblioteca **libfuse** *userspace*. A **libfuse** fornece a implementação de referência para comunicação com o módulo do *kernel* do **FUSE**.

O mecanismo desenvolvido foi concretizado sob a forma de um novo sistema de ficheiros baseado em **libfuse**. Este mecanismo deverá ser introduzido na *system call* chamada *open*, depois de verificar as permissões do utilizador sobre o ficheiro a ser aberto. O mecanismo autoriza a abertura de ficheiros depois da introdução de um codigo que será enviado por *email* ao utilizador. Para que o mecanismo saiba qual o *email* do utilizador, um administrador da máquina terá que criar e povoar um ficheiro com o nome "*.userEmails*", onde cada linha será um utilizador e terá o formato "<*username*> <*email*>", na diretoria "/".

2. Mecanismo

Para criar este mecanismo foi preciso completar os seguintes passos:

- Encontrar o endereço *email* do utilizador;
- Criar um código de acesso;
- Enviar o código de acesso para o endereço *email* do utilizador;
- E iniciar um servidor para poder ler o código introduzido pelo utilizador.

2.1. Procura do endereço *email* do utilizador

O ficheiro **userEmails** é aberto em modo de leitura e é lido linha a linha. Depois de lida uma linha, esta é analisada, ou seja, verifica-se se a linha pertence ao utilizador e se sim, retira-se o endereço *email*.

A função usada para ler linha a linha é a "**getline**" que recebe um apontador **NULL** e aloca a memória necessária para guardar a linha no apontador recebido. Para analisar cada linha, usou-se a função "**sscanf**", dizendo-lhe o número máximo de carateres a ler para o nome do utilizador (32 carateres) e para o endereço *email* (1023 carateres).

2.2. Criação do código de acesso

Para a criação do código de acesso foi usado um gerador de **uuid**(*universally unique identifier*), que usa o ficheiro **/dev/urandom**. O ficheiro **/dev/urandom** tem mecanismos com intenção de servir como um gerador de número pseudo-aleatório criptograficamente seguro, apesar dele não ser projetado por especialistas em criptografia. Para além do que foi referido, os códigos **uuid** têm tamanho fixo, permitindo alocar memoria fixar e não permitir *buffer overflows*. Para poder usar a biblioteca **uuid** foi necessário instalar a *package* **uuid-dev**.

2.3. Enviar o código

O envio do código foi feito através da biblioteca **libcurl**, que se conecta aos servidores do **gmail**, criando um *email*, com o destinatário, a fonte, o assunto e o código. Para usar o **libcurl** foi necessário instalar a *package* **libcurl4-gnutls-dev**.

Foram estabelecidas três tentativas para enviar o *email*, que após ultrapassadas é recusado o acesso ao ficheiro.

2.4. Servidor/Cliente

Por forma o utilizador a introduzir o código que recebeu, é iniciado um servidor, usando um *socket* que usa protocolos **TCP**, e espera um cliente se conectar.

O servidor inicia um *alarm* de trinta segundos, que quando termina invoca um *signal* que irá executar uma função para fechar o servidor recusando o acesso ao ficheiro. Da mensagem enviada do cliente, é apenas lido trinta e sete carateres (36 carateres para o **uuid** e 1 para o $\backslash 0$). De seguida, é comparado a mensagem recebida com o código gerado.

O Cliente pede o código ao utilizador e apenas lê trinta e sete carateres do *input*, que depois será enviado ao servidor. Para receber mensagens do servidor (erro ou sucesso), foi criado um novo processo, usando a função "**fork**", para tentar ler uma mensagem com um caratere do servidor e interpreta-la.

3. Conclusão

Em conclusão, o mecanismo foi implementado com sucesso, isto é, identifica o endereço *email* do utilizador, através da leitura de um ficheiro especifico, gera um código "aleatório", envia esse código ao utilizador através de um *email* e abre um servidor para o utilizador introduzir o código, através de um programa cliente.

Foram tidos em conta alguns pontos de segurança, como a geração do código de acesso, e na alocação de memoria.

É importante referir que o envio do *email* pode demorar, pois a conexão pode não ser feito na primeira tentativa. Visto que este mecanismo foi implementado e testado numa máquina virtual, este inconveniente pode não se revelar noutra máquina.