Segurança de Sistemas Informáticos

Licenciatura Engenharia Informática

Equipa de Trabalho:

Ana Margarida Sousa Pimenta a100830

Miguel Tomás Antunes Pinto a100815

Pedro Miguel da Costa Azevedo a100557

May 12, 2024

Contents

1. Introdução

No âmbito do projeto relativo à UC de Segurança de Sistemas Informáticos, foi-nos proposta a implementação de um serviço de conversação entre os utilizadores locais de um sistema Linux.

Neste relatório, iremos explorar e abordar os métodos adotados no nosso processo de realização do trabalho, bem como explicar as soluções encontradas e utilizadas nesta tarefa.

De modo a desenvolver este programa, recorremos aos conceitos lecionados e a informação obtida a partir de pesquisa efetuada pelo grupo.

Assim sendo, visando a concretização e o alcance do objetivo pretendido, iremos recorrer ao material reunido pelo grupo, promovendo sempre a procura por novos métodos, técnicas e conhecimento.

2. Solução Desenhada

Quanto à solução desenhada, para este projeto decidimos, seguindo o enunciado, dividir o nosso projeto de modo a que as concordias e o **deamon** comuniquem entre si. O daemon é responsável por receber, processar e responder às mensagens recebidas das concordias, processo que explicaremos de forma mais detalhada posteriormente.

Para este ponto, o pensado foi recorrer a **FIFOS** como estrutura intermediária na comunicação entre as concórdias e o daemon de forma a facilitar o meio de comunicação entre ambos.

O que acontece aqui, é que o utilizador executa uma tarefa (concordia), e essa tarefa é enviada para o FIFO, ao qual o deamon posteriormente vai aceder e ler a tarefa que tem de executar.

Quanto à solução desenhada, a nossa principal preocupação foi ter em conta as permissões ideias para garantir que apenas utilizadores autorizados possam aceder a diretorias ou ficheiros específicos. Relativamente a este processo, dado que o daemon é o ponto principal deste programa, e dado que este tem acesso a tudo, é importante garantir que este é executado com a maior segurança.

Para isso, tivemos em conta os seguintes aspetos:

- **Proprietário:** Ao alterar o proprietário do ficheiro executável do daemon para um utilizador, no nosso caso o **mydaemon**, definimos que este tem controlo sobre o mesmo. Isto é importante para garantir que o daemon tenha acesso aos recursos necessários e não seja executado com privilégios desnecessários.

- **Permissões de Leitura, Escrita e Execução:** Ao conceder permissão de execução para o proprietário, grupo e outros utilizadores, permitimos que o daemon seja executado conforme necessário e ideal. Já as permissoes de escrita, apenas concedemos ao dono do ficheiro, e de leitura, concedemos a todos.

- **Segurança e Proteção:** As permissões adequadas garantem a segurança do sistema, limitando quem pode executar o daemon e quais ações ele pode realizar. Deste modo, reduz-se o risco de acesso não autorizado ou exploração de vulnerabilidades.

3. Arquitetura do Sistema

Explorando agora a arquitetura do sistema, tal como abordamos, este projeto consiste em estabelecer um serviço de conversação.

Tendo já as principais introduções ao deamon e à sua aplicação no nosso projeto, podemos destacar então que este é inicializado quando o sistema é ligado e continua em execução para aguardar a chegada de mensagens das concordias. Assim que uma mensagem é recebida, o daemon processa-a e executa as operações que estão associadas à tarefa recebida.

Quanto à interoperabilidade, a comunicação entre o daemon e as concordias é estabelecida através de FIFOS, tal como abordamos previamente.

Estes FIFOs funcionam como canais de comunicação bidirecionais, permitindo assim que o sistema das concordias envie mensagens para o daemon e que este as receba. O formato das mensagens trocadas entre o daemon e o sistema concordias é definido por um protocolo de comunicação acordado entre os dois sistemas. Este método garante assim que as mensagens sejam compreendidas e processadas corretamente por ambas as partes.

Relativamente ao funcionamento do serviço, focando no **daemon** este é fundamentalmente baseado na manipulação de strings e no processamento de mensagens recebidas pelo FIFO.

Seguidamente faremos uma descrição mais focada em cada uma das principais funções do nosso **daemon**.

- **Espera de Mensagens:** O daemon entra em um loop infinito enquanto espera a chegada de mensagens no FIFO. Isso é feito através da função `read` que fica bloqueada até que novos dados estejam disponíveis no FIFO.
- **Processamento de Mensagens:** Quando uma mensagem é recebida, ela é passada para a função `process_message` para ser processada. Esta função divide a mensagem em diferentes partes com base no delimitador ":" e interpreta o comando, o remetente, o destinatário e outras informações adicionais, se estas estiverem presentes e pela ordem em que estas são enviadas da concordia para o FIFO.
- **Interpretação dos Comandos:** Com base no comando recebido na mensagem, o daemon executa diferentes ações. Por exemplo, se o comando for "concordia-ler", o daemon marca a mensagem como lida. Se for "concordia-responder", o daemon processa a resposta associada à mensagem. E assim sucessivamente.

Quanto às dependências e aos componentes, para além do meio de comunicação entre ele e as concordias, já abordado e explicado, este está também dependente do módulo `message_handling.h`. Recorremos a este módulo para recorrer a funções que são utilizadas para processar e manipular mensagens recebidas pelo daemon. Assim sendo, módulo é uma dependência interna do daemon e é essencial para o processamento adequado das mensagens que são recebidas.

4. Segurança do Serviço

De modo a garantir a segurança do serviço, decidimos estabelecer variadas regras.

Uma das primeiras decisões tomadas, focou-se no intuito da **segurança** e do **controlo de acesso aos ficheiros** e, para isso, decidimos **associar o nosso daemon a um utilizador específico**, neste caso, um que criamos especificamente para isso, o **mydaemon**.

Esta estratégia **limita assim as permissões do daemon estritamente ao necessário** para as suas funções operacionais. Isso reduz o risco de determinadas ações em caso de uma falha ou ataque, mas também previne o abuso de permissões que poderiam ser exploradas para fins maliciosos.

Ao executar o daemon sobre um utilizador único, ele **opera isoladamente dos outros processos do sistema**. Este isolamento **minimiza as interferências e eleva a segurança geral**, já que **qualquer comprometimento afetará apenas as permissões disponíveis para aquele utilizador específico**.

Na nossa Makefile, é onde definimos que o utilizador do daemon (mydaemon) é explicitamente designado como o proprietário do executável do daemon, o daemonexe. Isso é feito através do comando `sudo chown mydaemon $(BIN_DIR)/daemonexe`, e as permissões são configuradas como `sudo chmod 755 $(BIN_DIR)/daemonexe`. Estas permissões permitem então que o daemon **seja lido e executado por outros**, mas **apenas pode ser modificado pelo proprietário**, protegendo assim o executável contra alterações não autorizadas e **garantindo a integridade do programa**.

Quanto à criação de diretorias, a pasta concordia, com as informações relativas a utilizadores, grupos e mensagens, pertence também ao daemon, pois é ele quem as cria. Deste modo restringimos então o acesso e aumentamos a segurança prevenindo a alteração de ficheiros. Apenas o daemon está autorizado a escrever e alterar essas pastas, enquanto os grupos podem ler e escrever, e os outros podem apenas executar.

Para além disso, fizemos configurações acedendo ao `sudo visudo` de modo a garantir que algumas funcionalidades como o `groupadd` e `usermod`, por exemplo, fossem possíveis de executar. Para isso, adicionamos então a seguinte linha:

`mydaemon ALL=(ALL) NOPASSWD:/usr/sbin/groupadd, /usr/sbin/usermod, /usr/bin/groupdel`.

Esta configuração permite que o utilizador mydaemon execute os comandos especificados (groupadd, usermod, /usr/bin/groupmodel) como qualquer utilizador no sistema (ALL=(ALL)). Isto significa que o mydaemon tem a capacidade de executar esses comandos sobre a identidade de qualquer outro utilizador, não apenas como si mesmo.

Deste modo, garantimos as restriçoes necessárias tanto a nível de execuções como de comandos que cada utilizador pode executar, tentando garnatir e preservar a segurança do sistema.

5. Aspetos Funcionais

Quanto à funcionalidade do sistema, neste ponto conseguimos garantir a funcionalidade de grande parte das concórdias de acordo com o funcionamento e arquitetura definidos acima.

Como já abordado, de modo a realizar o processamento e execução das concórdias, quando uma é executada é enviada uma mensagem para o FIFO, no qual o daemon vai então ler, processar e executar as funcionalidades associadas aos comandos lidos.

Quanto ao método de segurança adotado, consideremos que atribuir um utilizador específico para o daemon é importante, tal como descrevemos acima, mas também, inicialmente não sendo o que tínhamos, conseguimos agora ter a percepção de que faz uma grande a diferença no sistema criado e no seu funcionamento. Para além disso, consideramos que para além de lógico, acaba por ser também necessário após entender o seu conceito.

Relativamente ao conjunto particular dos grupos e utilizadores, na criação das diretorias tivemos em consideração atribuir as respetivas e necessárias permissões ao seu bom funcionamento, tal como demonstraremos no excerto abaixo.

```
int create_user_directories(const char *username) {
    char path[1024];
    snprintf(path, sizeof(path), "/home/%s/concordia", username);
    mkdir(path, 0761);
    snprintf(path, sizeof(path), "/home/%s/concordia/recebidas", username);
    mkdir(path, 0701);
    snprintf(path, sizeof(path), "/home/%s/concordia/tarefas_grupos", username);
    mkdir(path, 0761);
    return 0;
}
```

Tal como podemos observar, cada pasta apresenta a sua permissão, tornando então as permissões mais restritas e por isso mais seguras.

Assim sendo, consideramos que alcançamos a maioria dos objetivos pensados e planeados.

6. Conclusão

Em suma, consideramos que este foi um projeto interessante e simultaneamente desafiador. Durante o seu processo de realização econtramos algumas dificuldades, como a compreender como iriamos "desenhar" o programa ou qual o método mais seguro e eficaz para realizar o nosso plano. Tais problemas acabaram por ser ultrapassados a partir de pesquisa e estudo realizado pelo grupo.

Quanto às falhas do nosso programa, um objetivo que não conseguimos concluir foi na parte da concórdia de enviar, na parte de enviar uma mensagem para grupos. Apesar disso, como demonstramos no código definido acima, definimos as restrições que consideramos mais adequadas e seguras para a pasta. Contudo, para a sua concretização, o nosso plano seria na concordia enviar verificar a presença do utilizador no grupo e se este garantia ou não as permissões necessárias para efetuar as sua atividade.

No que toca às melhorias, temos consciência que outros métodos de segurança poderiam ser adotados e executados.

Outro aspeto, seria uma melhor organização do código, pois como é possível verificar no GitHub do grupo, tinhamos um ficheiro `message_handling.c`, no entanto, devido a uma questão de diferenças de código entre os membros, acabamos por deixar grande parte das funções do processamento das concordias dentro do ficheiro do daemon. Contudo, temos plena consciência que não é o mais legível ou aconselhável, será sem dúvida, um aspeto que iremos melhorar.

De um modo geral, consideramos que este foi um trabalho bastante interativo e motivador apesar das dificuldades encontradas no caminho. Como grupo, demos sempre o melhor e procuramos aprender mais sobre o tema e sobre a segurança dos sistemas informáticos.