Sistemas de Computação Trabalho 2

Nesse trabalho de Sistemas de Computação o objetivo era implementar a interação de um cliente com um servidor para exercitar as conhecidas “system calls” do sistema operacional Linux. Para tal o cliente escrevia comandos, pedindo para que o servidor devolvesse alguma cosia.

No caso, foi utilizado comandos como escrita de arquivo, leitura, criação de diretórios, remoção de diretórios e arquivos, assim como uma listagem de todos os arquivos em um path específico. Como ficou relativamente livre a maneira de fazer as chamadas, segue abaixo o que o cliente pode fazer

createDir,pathToDir,DirName

Fazendo isso iremos criar um DirName do path pathToDir. Ele consegue tratar erros como se, por exemplo, o diretório já existir

deleteDir,pathTo,DirName

Fazendo isso iremos remover o DirName do path pathToDir. Ele consegue tratar erros como se por exemplo o diretório não existir.

read,pathToTheFile,offset,nrBytes

Fazendo isso iremos ler o arquivo localizado em pathToFile a partir da posição offset por nrBytes.

Write,userId,payload,pathToFile,offset,NrBytes,CanOwnerEdit,CanOthersEdit

Fazendo isso iremos escrever como usuario de id userId a mensagem contida em payload no arquivo pathToFile da posição offset ate NrBytes.

Aqui vale ressaltar que canOwnerEdit e canOtherEdit devem ser passadas sempre como parâmetro, mesmo que o arquivo já exista. Se ele já existir, já haverá informações suas no permissions.txt, e eles serão descartados pelo servidor.

Nessa função o userId sera verificado no arquivo permissions.txt para ver se aquele usuário pode ou não editar aquele arquivo.

Vale notar também que se nrBytes = 0 o arquivo é excluído se o usuário

Info,PathToDir,

Fazendo isso iremos listar todos os files/diretórios no PathToDir. Algo semelhante como “ls”.

InfoFile,PathToFile,

Fazendo isso estamos requisitando informações como quem é o dono e o tamanho do arquivo do file de path pathToFile.

Vale ressaltar a importância do ***permissions.txt***

Primeiramente foi pensando em usar uma struct como estrutura auxiliar para saber quem poderia mexer em qual arquivo. Entretanto por ser algo dinâmico que se perderia caso o servidor caísse, optei por usar um arquivo para manter os dados consistentes.

Permissions.txt tem

fileName owner ownerCanEdit othersCanEdit em cada linha

owner significa o id do usuário que criou

ownerCanEdit, 0 ou 1. 1 indicando que o owner pode editar

othersCanEdit, 0 ou 1. 1 indicando que o owner pode editar

Assim, a partir do momento que o arquivo é criado ele passa a ter essas permissões. Se outro usuário (que não sabe o id correto) tentar escrever algo no arquivo ganhará um acesso negado.

Exemplos de testes:

Info../, (lista tods os files no diretório corrente)

createDir,./,meudir (cria diretório meudir no diretório corrente)

deletDir,./,meudir2(delete diretório meudir2 do diretório corrente)

deletDir,./,meudir(delete diretório meudir do diretório corrente)

write,20,OlaGalera,meuTeste.txt,0,9,0,0 (usuário de id 20 escreve OlaGalera no arquivo meuTeste.txt, começando do offset 0, com 9 bytes, não dando permissão para futuras escritas nem pro owner nem pra outros usuários. Será checado no permissions ele pode/está criando o arquivo)

read,meuTeste.txt,0,9(usuário vai ler do arquivo meuTeste.txt,começando da posição offset 0 por 9 bytes)

write,20,OlaGalera2,meuTeste.txt,0,10,1,1 (usuário de id 20 escreve OlaGalera no arquivo meuTeste.txt, começando do offset 0, com 9 bytes, tentando dar permissão para futuras escritas e pro owner nem pra outros usuários Será checado no permissions ele pode/está criando o arquivo)

write,20,OlaGalera2,meuTeste2.txt,0,10,1,1 (usuário de id 20 escreve OlaGalera no arquivo meuTeste.txt, começando do offset 0, com 9 bytes, tentando dar permissão para futuras escritas nem pro owner e pra outros usuários Será checado no permissions ele pode/está criando o arquivo)

write,21,fala,meuTeste2.txt,9,4,1,1 (usuário de id 20 escreve OlaGalera no arquivo meuTeste.txt, começando do offset 0, com 9 bytes, tentando dar permissão para futuras escritas nem pro owner e pra outros usuários Será checado no permissions ele pode/está criando o arquivo)

A maior dificuldade do trabalho foi entender de fato as system calls, seus retornos e conseguir contornar corretamente seus possíveis erros. Foi importante para exercitar e entender melhor como o Linux trata essas chamadas.

Matheus Caldas - 1312760