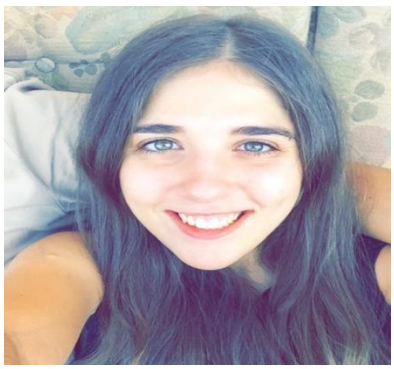
Sistemas Operativos
Trabalho Prático
2015/2016

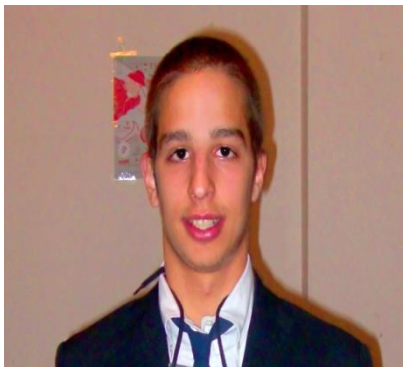
Sistema de backup de ficheiros

MIEI – 2º ano – 2º semestre

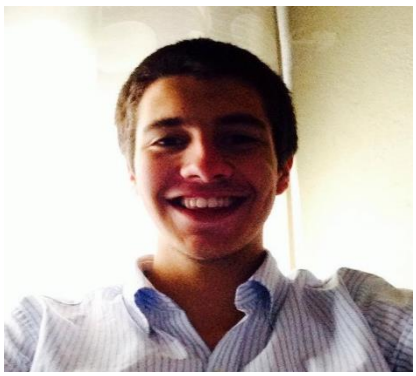
Grupo 13



Sara Pereira
A73700



Manuel Maria
A67713



José Silva
A74576

Índice

Introdução

No âmbito da unidade curricular de Sistemas Operativos, foi-nos proposto criar um sistema eficiente de cópias de segurança, de maneira a salvaguardar ficheiros de um utilizador.

O sistema pode realizar duas opções principais em relação ao utilizador, sendo estas a operação de backup e restore dos ficheiros para a pasta original.

Tal como foi referido no enunciado, foi acrescida dificuldade a este projeto na medida em que foi apelado tanto à eficiência como à privacidade. Com eficiência, entende-se pela minimização do espaço ocupado em disco pelo backup e com a concorrência de operações a serem efetuadas no servidor. Com privacidade, será necessário utilizar uma arquitetura que impeça o acesso direto a pasta de backup e gestão de ficheiros por parte do cliente.

Por forma a cumprir estes requisitos fomos logo capazes de ver que uma tarefa que inicialmente parecia uma simples cópia de ficheiros se torna um pouco mais complicada.

Arquitetura da Aplicação

A aplicação é constituída por por dois programas principais, o programa Servidor *sobusrv* e o programa Cliente *sobucli*.

O servidor *sobusrv* tem como tarefa receber os pedidos por parte dos clientes e gerir os mesmos em concorrência, não mais que 5 de cada vez devido á sobrecarga do sistema e realizar as operações que os mesmos indicaram bem como comunicar o resultado no final ao cliente que efetuou o pedido de tal operação por sinais sendo o sucesso ou erro da mesma o output comunicado.

O programa Cliente *sobucli* tem a possibilidade de realizar o backup dos ficheiros que desejar, e restaurar os mesmos através do comando *restore*, bem como eliminar um ficheiro do servidor onde estão alojados e fazer a limpeza das entradas que não estejam linkadas simbolicamente a nenhum ficheiro.

Funcionalidades

Backup

A funcionalidade **backup** é a principal e o objetivo de todo o trabalho tendo como função o envio de ficheiros que um cliente contém para um servidor a pedido do mesmo.

Após verificação da existência dos ficheiros, estes são enviados para o servidor em blocos numa estrutura que permita um rápido tratamento dos dados á chegada ao servidor.

No servidor estes blocos de ficheiro são tratados e caso o digest do ficheiro por parte do sha1sum indique que algum ficheiro com aquele conteúdo já se encontra na pasta data a cópia do ficheiro não é efetuada e é apenas efetuado um link para o nome do ficheiro em metadata evitando assim cópias de conteudo duplicado.

Não sendo esse o caso o ficheiro é reconstruído até que o ultimo bloco indique que o envio do ficheiro terminou, onde logo após o término deste processo de envio o ficheiro é comprimido pelo programa gzip para uma mais eficiente salvaguarda dos ficheiros em termos de memória. De seguida o nome é alterado para o código do digest do seu conteúdo obtido com o programa sha1sum e um link simbólico para o seu nome original é criado na pasta metadata.

No fim do processo de cópia é enviado um sinal para o cliente com o resultado do backup ter sido ou não bem sucedido.

Restore

O comando **restore** é uma funcionalidade obrigatória neste trabalho para que o cliente tenha forma de reaver os ficheiros dos quais efetuou o backup quando assim o requisitar.

Quando um cliente pede ao servidor que efetue o restore de um ficheiro, envia a este o nome dos ficheiros que pretende reaver através de uma estrutura Ficheiro.

Recebendo a informação do comando o servidor verifica a existência dos ficheiro em metadata e caso seja confirmada vê através do link simbólico que este contem qual o ficheiro a que corresponde em data, onde se encontra efetivamente o ficheiro. Ai após cópia por uma questão de segurança do processo de restauração e descompressão da mesma, o ficheiro é enviado por um pipe respetivo ao cliente que pediu a operação repartido em blocos e enviados numa estrutura igual á que envia o ficheiro do cliente para o servidor na funcionalidade backup, a estrutura Ficheiro. O cliente adquire assim o ficheiro na pasta em que o comando restore foi efetuado.

O processo termina com o cliente a ser informado do mesmo ter terminado ou a ser avisado da não existência de tal ficheiro se tal acontecer bem.

Gc

A funcionalidade do comando gc prende-se com a remoção de ficheiros na pasta data do servidor que não tenham a si associados links simbólicos em metadata, estando os mesmos a ocupar espaço desnecessariamente pois o utilizador não tem forma de os recuperar. Quando o utilizador invoca esta funcionalidade é enviado ao servidor o pedido da operação gc e o servidor começa neste momento a verificação nos ficheiros em data um a um a existência de links por parte de outros ficheiros a eles através do comando find.

Caso nenhum link seja encontrado o ficheiro em data é removido, libertando-se assim a memória do mesmo.

Delete

O comando delete remove o link da pasta metadata de um ficheiro indicado pelo cliente caso o mesmo exista, ficando ao cargo do comando gc fazer a posterior remoção do ficheiro em data caso a este não exista mais nenhum link simbólico associado.

Estrutura de comunicação

Quando o projeto começou a ser pensado deparamo-nos logo com problemas como a rápida identificação do ficheiro em questão por parte do servidor ou a forma de enviar o ficheiro de maneira a não haver uma perda de informação e alertar para o fim da cópia do ficheiro para que a operação pudesse continuar, bem como facilitar a identificação do seu conteúdo.

Para tal foi criada uma estrutura “Ficheiro” que serve de comunicação entre servidor e cliente que contém as seguintes informações:

• ***pid_cliente*** que requisitou a operação para que o mesmo seja informado do resultado da funcionalidade pedida.

• ***comando*** que é um array onde é guardado o comando que o cliente pretende efetuar.

• ***ficheiro*** que é um array com o ficheiro sobre os quais pretende efetuar uma operação caso os mesmos exijam tal informação.

```
#define SIZE 256
#define FILE_SIZE 4096

typedef struct ficheiro {
    pid_t pid_cliente;
    char comando[SIZE];
    char ficheiro[SIZE];
    char codigo[SIZE];
    char conteudo[FILE_SIZE];
    int estado;
    int tamanho;
} *Ficheiro;
```

• ***codigo*** é um array onde é colocado posteriormente o sha1sum para fácil acesso por parte do servidor em todos os pontos ao código do conteúdo do ficheiro.

• ***conteudo*** é um array onde são colocados até 4kbytes do ficheiro que é necessário fazer backup ou restore.

• ***tamanho*** é um inteiro onde é guardado o número de bytes da informação lida num ficheiro.

• ***estado*** onde indicamos se o ficheiro terminou a sua cópia ou ainda se encontra em processo de cópia de dados.

Desta forma todas as funções praticamente apenas necessitam de trabalhar com esta estrutura aproveitando as informações que a mesma contem tornando mais limpo o código dos programas mas também ajudando no acesso a dados.

Comunicação entre Cliente e Servidor

O Cliente comunica com o servidor atrvés de um ***named pipe*** situado na raiz do backup ou seja na pasta */home/user/.Backup.*

O Cliente usa o pipe para escrita enviando informações relativas as operações pretendidas para o servidor que está a correr em background ficando numa espera passiva por dados para que efetue operações.

A comunicação inversa do servidor para o cliente é feita através de sinais, tendo sido usados o SIGUSR1 para comunicar sucesso numa operação e SIGUSR2 para comunicar erro, ficando a cargo do cliente imprimir para cada operação a respetiva mensagem de erro ou suceso.

Para comunicação no comando restore foi necessária a criação de um pipe para o cliente para que este tivesse possibilidade de receber um ficheiro enviado por parte do servidor.

Makefile

Make

O comando make tem a habitual função de compilar o programa gerando os executáveis correspondentes ao Servidor (sobusrv) e ao cliente (sobucli).

Make Clean

Fazendo make clean são limpos os executáveis presentes na pasta onde foram gerados.

Make Installation

Move o executável para a pasta /bin, solução que teve de ser implementada devido a termos como terminal o zsh e não a bash.

Make cleanAll

Remove os executáveis que se encontram na pasta /bin.

Make Kill

Comando para matar o processo do servidor criado, visto este estar a correr em background.

É sugerido ao utilizador para uma utilização correta do programa que proceda a fazer make e de seguida make installation para que possa começar a utilizar o programa.

Conclusão

O desenvolvimento deste projeto contribuiu para que pudéssemos alargar os conhecimentos que tínhamos apreendido na unidade curricular de Sistemas Operativos ao longo do semestre apesar de num contexto totalmente diferente e de uma complexidade bastante mais elevada, pois agora fomos ao encontro de uma junção de todos os pequenos pontos que tínhamos vindo a referir nas aulas praticas.

Trata-se de um trabalho de uma dimensão média, mas de uma complexidade de raciocínio mais elevado face ao que nos tinha sido proposto até hoje no curso, o que levou a criar uma maneira de pensar bastante diferente face a introdução de conceitos como a concorrência de processos e os sinais.

Assim sendo, tornou-se um grande desafio e por isso bastante mais gratificante pois deparamo-nos com alguns obstáculos que conseguimos ultrapassar adquirindo assim uma maior experiência no que toca ao uso destes conceitos aprendidos ao longo do semestre em Sistemas Operativos.