Sistemas Operativos 2016/2017

Trabalho prático – servidor HTTP

João Correia Campos, nº 2013138305
Ana Inês Fidalgo, nº 2013134819

Relatório – Projeto

Neste projeto tínhamos como objetivo desenvolver um servidor web capaz de servir pedidos HTTP, podendo ser páginas estáticas (ficheiro HTML) ou páginas estáticas comprimidas.

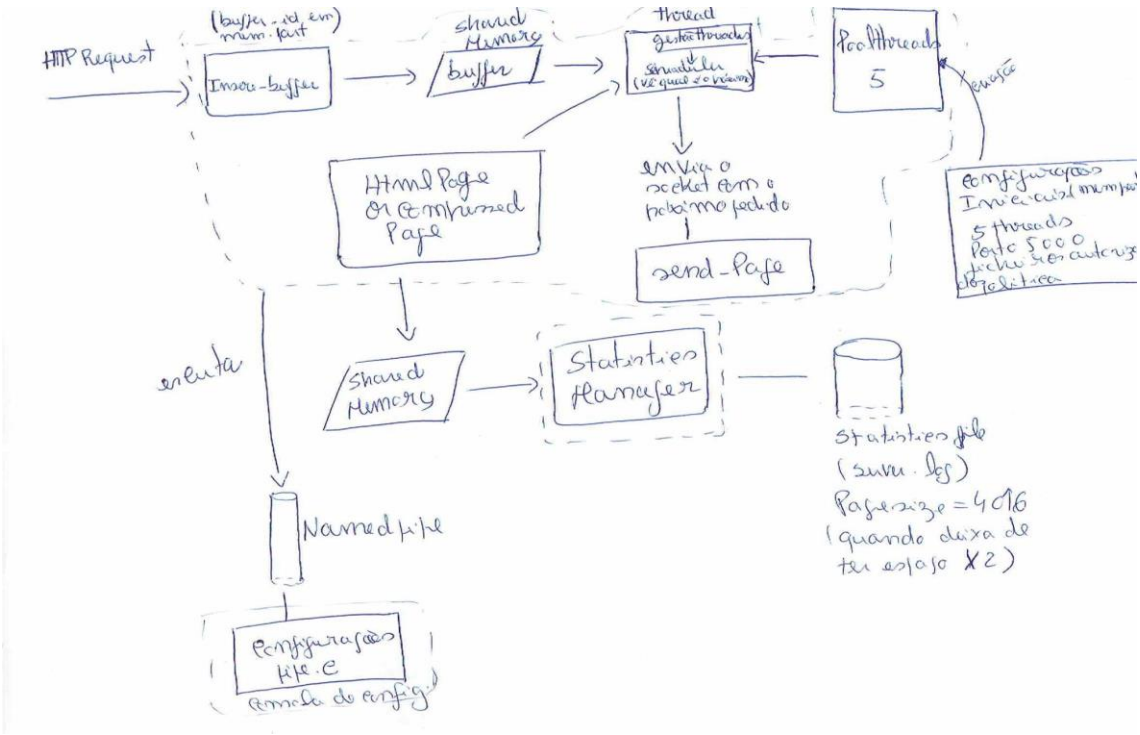

Figura 1 – visão geral do funcionamento do servidor

Quando iniciamos o servidor, este começa por criar um ficheiro de memória partilhada (server.log) para guardar as informações de cada request. Cria também em memória partilhada o buffer (apontador para estrutura de pedidos) onde são armazenados os requests, uma estrutura de configurações, usada no processo de configurações para armazenar a informação do ficheiro, assim como um id para a posição de escrita no buffer. Cria também o named pipe e, assim como os processos.

O servidor por nós implementado utiliza 3 processos:

- Processo de configuração que lê de um ficheiro (configfile.txt) as configurações com qual o servidor vai iniciar, tais como porto ao qual se vai ligar, número de threads a serem criadas, política e nome dos ficheiros estáticos comprimidos autorizados, um por cada linha;
- Processo de estatísticas, que fica à espera de sinais do tipo SIGUSR1 e SIGUSR2 para imprimir no ecrã o resultado das estatísticas e reset das mesmas, respetivamente.
- Processo principal, que é responsável por inicializar os semáforos, criar a pool de threads, receber as ligações http, fazer o escalonamento de pedidos e ficar à escuta de um named pipe;

Após aceitar um pedido no processo principal, vai identificar o cliente e fazer o get_request, guardando o instante em que foi realizado o pedido. Verifica se está autorizado e, em caso afirmativo, vai verificar qual o tipo de ficheiro(estático ou comprimido). Se for comprimido faz uma system cal para descompressão do ficheiro. Depois disso coloca no buffer o pedido e faz sem_post para dar sinal às threads que foi colocado novo pedido no buffer.

Uma das threads do servidor vai chamar a função scheduler que devolve o socket do pedido ao qual vai enviar a página. Esse método vai verificar qual o tipo de política que está na estrutura de configurações. Caso seja um pedido normal vai buscar o pedido mais antigo no buffer, caso seja pedido estático vai percorrer o buffer e ver qual o pedido estático mais antigo devolvendo o socket desse pedido. Caso não encontre nenhum pedido desse tipo no buffer devolve o socket do pedido do tipo comprimido mais antigo. Caso seja comprimido efetua o mesmo processo mas invertido. Envia depois a página respectiva e chama o método escreve_estatisticas que trata de escrever e actualizar estatísticas.

Na função escreve_estatisticas primeiro verificamos se a informação que queremos escrever ultrapassa o pageSize (limite de bytes que alocamos para escrita). Caso seja ultrapassado, antes de escrever damos unmap da memória e voltamos a mapear com um pageSize com o dobro do tamanho anterior. Em seguida, é escrita a informação do pedido no ficheiro e incrementamos os contadores (variáveis globais) de cada uma da informação a reter.

Para alteração das configurações através de uma consola de configuração criamos no processo principal uma thread que vai ficar à escuta, através de um named pipe, de novas configurações introduzidas. As threads que estão a tratar de pedidos finalizam o que estavam a fazer antes de serem canceladas e é criada uma nova pool de threads. O ficheiro pipe.c pede os dados de configuração a alterar e envia-os pelo named pipe.

Se fizermos kill do pid com o sinal USR1 ele chama o método que dá reset dos contadores de estatísticas. Caso o sinal seja USR2 faz as médias devidas e imprime no ecrã os resultados.

Esforço: 50 horas p/aluno.