Análise do tempo de transferência e tamanho de buffer em um servidor web

Renan Rodrigues Sousa [1], **Rodrigo José Zonzin Esteves**[1]

[1]Curso de Ciência da Computação – Departamento de Computação
Universidade Federal de São João del Rei (UFSJ)

1. Introdução

A internet representa é compota pela interconexão de inúmeras subredes de computadores, buscando a comunicação eficiente entre dispositivos distribuídos globalmente. A fim de se estabelecer eficiência e confiabilidade no tráfego dos dados, é imprescindível a existência e adoção de protocolos específicos que definam e padronizem a forma como essas redes se comunicam.

De modo didático, a Internet é dividida em camadas que desempenham funções distintas, mas interligadas: a camada de aplicação, transporte, rede, enlace e física. Cada uma dessas camadas opera com protocolos específicos que asseguram a integridade, segurança e eficiência das comunicações, ao menos de forma parcial.

A camada de transporte, por exemplo, é responsável por gerenciar a comunicação entre os dispositivos e opera por meio dos protocolos TCP (Transmission Control Protocol) e UDP (User Datagram Protocol). Enquanto isso, a camada de rede utiliza o protocolo IP (Internet Protocol) para rotear os dados na rede global, determinando a alcançabilidade e o melhor caminho para a transmissão dos dados.

Na camada de aplicação, são definidos protocolos como o HTTP (Hypertext Transfer Protocol), FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol) e outros, que possibilitam a interação entre os usuários e os serviços disponíveis na internet. É nesta camada que se estabelece o modelo cliente-servidor, fundamental para a requisição e transmissão de arquivos.

A construção de um modelo cliente-servidor na camada de aplicação envolve a utilização dos recursos das camadas inferiores, unindo o transporte confiável de dados oferecido pelo protocolo TCP e a intercomunicabilidade de redes oferecida pelo protocolo IP. Nesse sentido, a interação entre cliente e servidor se dá através da pilha de protocolo TCP/IP, permitindo tanto a solicitação quanto a resposta eficiente aos pedidos de informação.

O presente trabalho visa implementar um sistema baseado no modelo cliente-servidor, implementando tanto a transmissão unidirecional de arquivos, quanto a comunicação bidirecional do tipo requisição-resposta. Essa implementação explora a aplicabilidade dos conceitos teóricos das camadas da internet ao mesmo tempo que evidencia a importância dos protocolos na construção e funcionamento de sistemas de comunicação eficientes.

2. Implementação

2.1. Servidor

O servidor envia arquivos para clientes que se conectam a ele através de um socket TCP/IP. Ele aceita conexões de clientes, recebe o nome de um arquivo e envia o conteúdo desse arquivo de volta para o cliente. O programa requer dois argumentos na linha de comando ao ser executado:

porta_servidor: O número da porta em que o servidor será executado.

tam_buffer: O tamanho do buffer utilizado para enviar os dados para o cliente.

O servidor cria um socket para comunicação em IPv4 usando o protocolo TCP. Esse socket é então associado a uma porta específica usando $bind()$, permitindo que o servidor escute conexões nessa porta. Após a associação do socket à porta, o servidor aguarda conexões de clientes usando $listen()$. O servidor foi definido para atender a apenas um cliente por vez

Quando um cliente se conecta ao servidor, $accept()$ é chamado para aceitar essa conexão. Isso cria um novo socket, *cliente_socket*, dedicado exclusivamente a essa conexão com o cliente. Após a conexão ser estabelecida, o servidor recebe o nome do arquivo enviado pelo cliente através do socket utilizando $recv()$. Ele verifica se há erros durante o recebimento e, em caso positivo, fecha a conexão e o socket. Com o nome do arquivo recebido, o servidor tenta abrir o arquivo em modo de leitura *("r")*.

O servidor lê o conteúdo do arquivo em blocos do tamanho especificado pelo buffer usando $fread()$ e envia esses blocos para o cliente usando $send()$. O envio é feito repetidamente até que todo o arquivo tenha sido enviado ou ocorra um erro. Após a conclusão do envio do arquivo, o servidor fecha o arquivo aberto e os sockets.

O tempo gasto durante a transferência é calculado utilizando $gettimeofday()$ no início e no fim do processo de envio do arquivo. Isso permite calcular a diferença de tempo para determinar a duração total da transferência.

2.2. Cliente

O programa do cliente se conecta a um servidor remoto e recebe um arquivo, medindo o desempenho da transferência em termos de taxa de transferência e tempo gasto. O programa requer quatro argumentos na linha de comando ao ser executado:

host_do_servidor: O endereço IP do servidor remoto.

porta_servidor: O número da porta no servidor onde a conexão será estabelecida.

nome_arquivo: O nome do arquivo que será transferido do servidor para o cliente.

tam_buffer: O tamanho do buffer usado para a transferência de dados.

O cliente cria um socket para comunicação em IPv4 utilizando o protocolo TCP. O programa tenta estabelecer uma conexão com o servidor remoto usando $connect()$ com o endereço IP e porta fornecidos pelo usuário. Se a conexão for bem-sucedida, o cliente estará conectado ao servidor.

Após a conexão ser estabelecida, o cliente envia o nome do arquivo solicitado para o servidor usando $send()$ através do socket. O cliente abre um arquivo local com o

mesmo nome do arquivo solicitado em modo de escrita *("w+")* para receber os dados do servidor. Se houver falha na abertura do arquivo, o cliente encerra a conexão e finaliza a execução.

O cliente entra em um loop onde recebe os dados enviados pelo servidor usando $recv()$ e escreve esses dados no arquivo local usando $fwrite()$. Esse processo continua até que não haja mais dados para receber ou ocorra um erro na recepção. Após a conclusão da transferência do arquivo, o cliente fecha o arquivo local e o socket de conexão com o servidor.

O tempo decorrido durante a transferência é medido usando $gettimeofday()$ no início e no fim do processo de recebimento do arquivo. Isso permite calcular a diferença de tempo para determinar a duração total da transferência. Ao final da execução, o cliente imprime informações sobre o desempenho da transferência, incluindo o tamanho do buffer, a taxa de transferência (em *kbps*), o total de bytes recebidos e o tempo gasto durante a transferência.

3. Resultados

Para teste, selecionamos quatro tipos de arquivos distintos: uma imagem, um arquivo PDF, um vídeo MP4 e um arquivo de texto. Para cada um desses arquivos, foram realizadas execuções com quatro diferentes tamanhos de buffer: 56, 512, 1024 e 2048 bytes. Isso resultou em um total de 16 observações para análise, que foram agrupados e analisados.

A tabela 1 apresenta a correlação de pearson entre o tempo de transferência e o tamanho do buffer. Espera-se uma correlação negativa.

Arquivo	Tamanho	Correlação
Imagem	39,7 kB	-0.9863
Video	509,5 kB	0.2145
PDF	10,2 MB	0.7426
Texto	129 kB	0.1177

Table 1. Comparação da correlação entre tempo de envio e tamanho de buffer

A figura 1 apresenta a taxa de transferência em função do tamanho do buffer. Conforme o esperado, quanto maior o buffer, maior a taxa de transferência (mais bytes enviados por segundo) e, por consequência, menor o tempo de envio.

As figuras 2 e 4 estão em desacordo com o esperado pois à medida que o tamanho de buffer aumenta, a taxa de transferência diminui. Uma possível explicação é o tamanho do arquivo.

Enquanto a imagem possui apenas 39 kB, os demais dados apresentam tamanhos superiores a 100 kB. Essa discrepância pode ser reflexo de uma taxa maximal de envio, onde o desempenho piora a partir de um limite no tamanho de buffer, considerando fatores computacionais do sistema operacional ou da rede.

4. Conclusão

A implementação de um sistema baseado no modelo cliente-servidor revelou aspectos importantes sobre a transferência de dados na internet. A estrutura em camadas da rede,

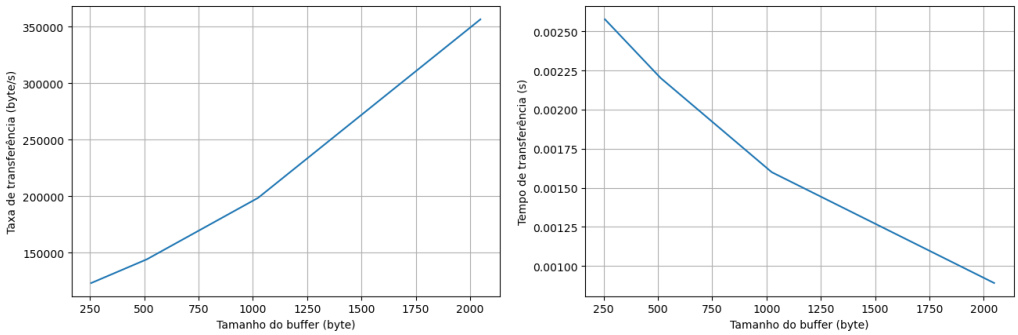

Figure 1. Resultados para o arquivo de imagem

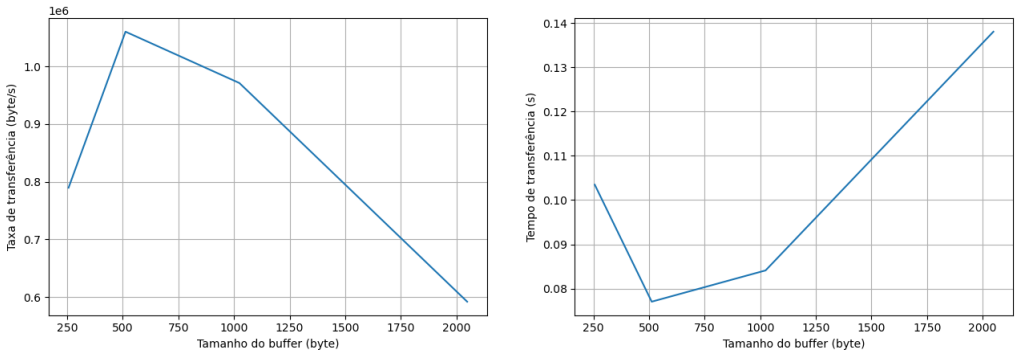

Figure 2. Resultados para o arquivo pdf

com protocolos específicos, demonstra sua relevância na eficiência e na confiabilidade das comunicações. Ao explorar a correlação entre o tempo de transferência e o tamanho do buffer, percebe-se uma relação direta entre esses elementos, embora algumas exceções tenham sido observadas. Enquanto em alguns casos o aumento do tamanho do buffer resultou em uma melhoria da taxa de transferência, em outros, esse comportamento esperado não se confirmou. Essas descobertas apontam para nuances na interação entre protocolos, camadas da rede e o tamanho do buffer, destacando a complexidade do processo de comunicação na internet e a necessidade de considerar diversos fatores para otimizar a transmissão de dados.

References

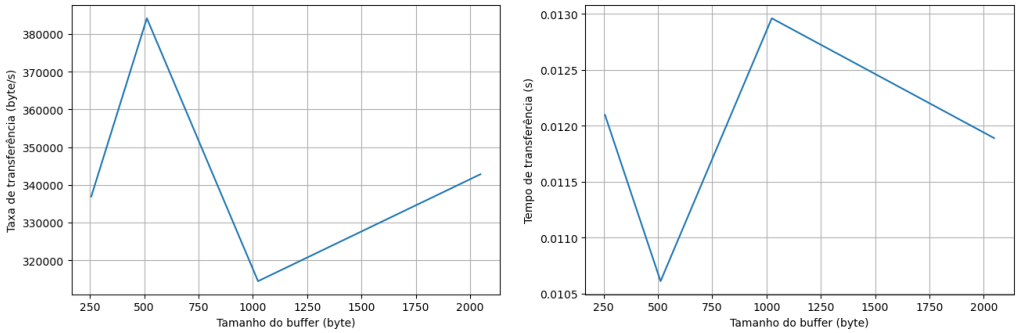

Figure 3. Resultados para o arquivo de vídeo

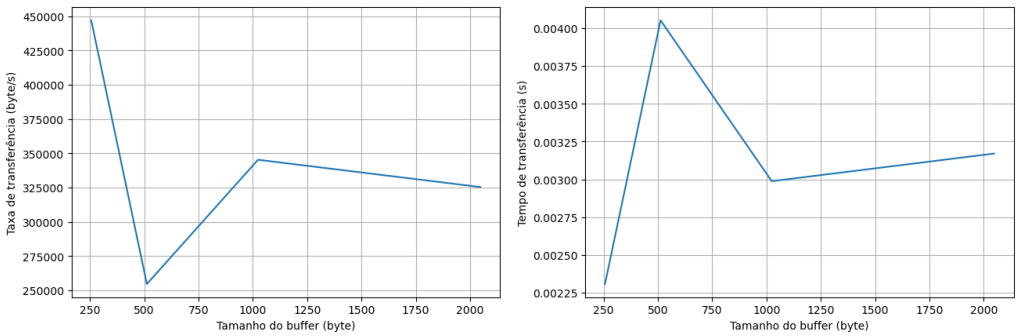

Figure 4. Resultados para o arquivo de texto