Projeto 2 - Redes de Computadores

João Pedro Assunção Coutinho
18/0019813

João Victor de Souza Calassio
18/0033808

Pedro Luis Chaves Rocha
18/0054635

***Abstract*—This project implements the ARQ protocols in the transport layer.**

I. Introdução

O projeto 2 da disciplina tem como objetivo fazer uma simulação dos protocolos de controle de fluxo e erro na camada de transporte. O grupo escolheu implementar os protocolos Stop-and-Wait e Selective-Repeat. Esses protocolos controlam possíveis erros de transmissão de pacotes no protocolo UDP, utilizando técnicas de confirmação entre o receptor e o emissor.

II. Fundamentação teórica

Para a realização do trabalho, o grupo baseou-se na teoria de um dos protocoloas da camada de transporte mais conhecidos, o RDT (Reliable Data Transfer Protocol), mais especificamente suas variantes, Stop And Wait e o Selective Repeat.

O Stop And Wait é idealizado como um protocolo em que, existem 2 máquinas de estados, uma para o receptor e outra para o emissor, em que a primeira, fica na maioria do tempo no estado esperando receber um dado. Quando o receptor recebe o dado, nas versões mais completas, ele faz uma checagem por erros com um checksum e, caso tudo ocorra certo, ele retorne um ACK, que é uma confirmação de que tudo ocorreu bem, caso contrario, nada será devolvido ao emissor (em versões mais básicas é retornado um NAC). Por parte do emissor a máquina de estados é mais complexa, pois envolve enviar o dado, aguardar o ACK, checar por error e/ou duplicatas setando um valor alternado de 0 e um para os pacotes e ACKs, além de reenviar o dado caso algo tenha dado errado, tendo assim a principal característica do Stop And Wait, sempre enviar o dado e esperar até que o receptor confirme com um ACK que ocorreu tudo certo.

Além do Stop And Wait, no projeto também foi implementado o Selective Repeat que consiste na mesma ideia de enviar e receber os dados de forma a checar por erros e duplicatas, mas com a diferença de que o emissor não espera receber um ACK para enviar o próximo dado, ele simplesmente envia vários dados em sequencia, que não ultrapassem o tamanho da janela de transmissão, e aguarde um ACK, sem para de transmitir, para cada pacote enviado, pois, caso algum pacote se perca, o emissor saberá exatamente qual pacote foi perdido podendo assim, reenviá-lo ao receptor.

Como o Stop And Wait aguarda cada ACK individualmente, ele se tornou inviável nos dias de hoje que é exigido baixa latência para diversos serviços fazendo assim com que se buscasse novas alternativas para esse problema como por exemplo o Selective Repeat que mesmo estando mais sujeito a erros, é mais rápido que o Stop And Wait pelo simples fato que que o tempo de transmissão e melhor aproveitado pela ideia de Pipeline utilizada na implementação do selective repeat.

III. Ambiente experimental e análise de resultados

A. Descrição do cenário

Foi utiliza a linguagem Python 3 para implementação dos protocolos. Utilizando a interface Tkinter, o grupo criou uma interface para inserção dos parâmetros pedidos: vazão, distância, probabilidade de erros e tamanho da janela (no caso do Selective Repeat). Assim, o usuário pode escolher iniciar a conexão com o protocolo desejado, e o receptor e emissor são executados paralelamente. As mensagens enviadas e recebidas em cada um deles são mostradas no terminal.

O grupo escolheu como padrão, os seguintes valores (arbitrários) para ambos os testes com os protocolos:

Vazão 8000 bps;

Distância 10 metros;

10% de probabilidade de erros;

Tamanho da janela com 7 pacotes

Para a execução da simulação, são enviados pacotes com um número aleatório qualquer de 0 a 255, até o programa ser encerrado.

Para o protocolo Stop-and-Wait, um número é enviado com um identificador binário (ex: 0 162), e então o receptor envia o aviso de recebimento relativo a esse mesmo identificador (ex: 0 ACK). Se o emissor receber o aviso de recebimento do pacote 0, ele então envia o próximo pacote, com identificador 1, e assim sucessivamente (0, 1, 0, 1).

Para o protocolo Selective Repeat, são escolhidos N números aleatórios, sendo N o tamanho da janela. Então a janela de N números inteira é enviada para o receptor, pacote a pacote, com o seu identificador de 0 a N. O receptor então deve avisar o recebimento de cada pacote individualmente (ex: 1 ACK, 2 ACK ... N ACK). Apenas quando o emissor receber a confirmação de todos os pacotes da janela atual, ele gera novos N números para enviar na próxima janela.

Ambos foram testados com vazão de 8000 e 16000 bps; distâncias de 10, 100 e 1000 metros; probabilidade de erros de 1, 10 e 90%; janelas com 5, 7 e 10 pacotes.

B. Análise de resultados

A implementação do protocolo Stop-and-Wait funcionou como esperado para vários parâmetros testados, com as mensagens só sendo enviadas após recebimento no cliente. A

principal desvantagem observada foi a demora para se enviar uma grande quantidade de pacotes. Por ter que esperar o recebimento do ACK do pacote, antes de enviar o próximo, o envio de uma grande quantidade de pacotes (grandes arquivos, no caso) se tornaria bem mais lenta.

Com o protocolo Selective Repeat, o funcionamento também foi como esperado para os parâmetros testados. As janelas são respeitadas, e observou-se que era consideravelmente mais rápido que o Stop-and-Wait. No entanto, com uma probabilidade de erros mais alta, ele também se torna muito lento, por ter que esperar os timeouts de cada pacote individualmente.

IV. Conclusão

No laboratório 2, foram implementados, com sucesso, os protocolos Stop-and-Wait e Selective Repeat, que trazem mais confiabilidade ao protocolo UDP. O laboratório permitiu ao grupo conhecer mais profundamente o funcionamento desses protocolos, já que os mesmos foram implementados e simulados.

Segue o link do vídeo: https://youtu.be/O6kjn-pY5gE

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.