Segundo projeto: Transferência Confiável de Dados Baseada em Janela com Repetição Seletiva ou Go-Back-N Usando Python

1. Objetivo

O objetivo deste projeto é usar sockets UDP e a linguagem de programação python para implementar um protocolo de transferência de dados confiável.

2. Instruções

Neste projeto, você implementará um protocolo simples de controle de congestionamento baseado em janela (window-based) e construído em cima de um protocolo de repetição seletiva (Selective Repeat) ou de um protocolo Go-Back-N. É da sua escolha com qual protocolo você vai trabalhar. Assim também, é necessária a implementação da operação de soma de verificação (checksum) para garantir a entrega de pacotes confiável. Você deve implementar dois programas: o programa emissor e outro programa receptor.

a. Os dois programas deverão se comunicar usando o protocolo UDP (o qual não garante a entrega dos dados).

b. O programa receptor deverá atuar como uma aplicação cliente, solicitando um arquivo do emissor, o qual atua como um servidor.

c. Na execução do programa receptor, na linha de comando, o programa especificará como argumentos o hostname e o número da porta do emissor, assim como também o nome do arquivo que quer recuperar do emissor. Por exemplo:

```
>receptor <hostname do rementente> <numero de porta do
        rementente> <nome do arquivo>
```

d. O programa emissor, que atua como servidor, deverá especificar o número da porta do serviço como argumento da linha de comandos na sua execução. Exemplo:

```
>emissor  <numero de porta do serviço>
```

e. O receptor primeiro enviará uma mensagem para o emissor, indicando o nome do arquivo que deseja obter. Se o arquivo existir, o emissor irá dividir o arquivo inteiro em vários pacotes e, em seguida, irá acrescentar algumas informações de cabeçalho para cada pacote antes de enviá-los para o receptor. Cabe a você determinar as informações que você deseja incluir no cabeçalho (por exemplo, porta origem, porta de destino). No entanto, dada a fragmentação do arquivo em vários pacotes, considere que você vai precisar de um campo de número de sequência. Você é livre para definir que tipo de mensagens você vai exigir, e o formato das mensagens. Para testar seu programa, você pode criar um arquivo grande para ser solicitado, garantindo assim que o arquivo vai ser transmitido em vários pacotes.

f. Você deve imprimir mensagens quando o servidor ou o cliente está enviando ou recebendo pacotes. Sua mensagem deve incluir informações como: se o pacote sendo transmitido é um pacote de dados ou um ACK, o número de sequência, se ele está corrompido ou perdido.
g. Finalmente, observe que os programas devem atuar tanto como aplicações para a transferência de arquivos pela rede, como também, protocolos de camada de transporte confiável construídos sobre a camada de transporte UDP não confiável.

3. **Tratamento de erros**

A taxa real de perda de pacotes ou corrupção na LAN pode ser muito baixa para você testar seu programa. Portanto, você deve simular a perda de pacotes e corrupção da seguinte maneira:

- **Perda de pacotes:** Com probabilidade P_L ignore os pacotes que chegam. Ou seja, finja com uma probabilidade P_L que você não recebe os pacotes.
- **Corrupção de pacotes:** Com probabilidade P_C marque um pacote que chega como sendo corrompido.

Considere valores apropriados para P_L e P_C, tais como valores no intervalo de 0 a 0,40. Observe também que tanto pacotes que transmitem dados como os seus respectivos pacotes de reconhecimento (acknowledgement) podem ser corrompidos.

4. **Sugestões**

A melhor maneira de abordar este projeto é em passos incrementais. Não tente implementar toda a funcionalidade de uma vez. Por exemplo, a sua implementação, poderia considerar o seguinte:

- Em primeiro lugar, assumir que não existe perda de pacotes ou corrupção. Basta que o emissor consiga enviar pacotes ao receptor, que consiga receber os ACKs, e assim por diante.
- Em segundo lugar, introduzir a corrupção. Isto significa que você deve implementar algumas funcionalidades de retransmissão.
- Em terceiro lugar, introduzir a perda de pacotes. Agora você terá que adicionar um temporizador no lado do emissor para cada pacote.

A avaliação do seu projeto dependerá da implementação das funções necessárias. Se você só terminar parte dos requisitos, você receberá uma pontuação parcial. Assim, faça o projeto de forma incremental.

Para esclarecer o funcionamento dos seus programas, você pode fazer que o tamanho da janela (CWnd), a probabilidade de perda de pacotes P_L e a probabilidade de corrupção do pacote P_C, sejam parâmetros de entrada nos programas de servidor e cliente. Por exemplo:

Programa emissor: `>emissor <número de porta> CWnd Pl PC`
Programa receptor: `>receptor <sender hostname> <sender-porta> <filename> Pl PC`

5. **Submissão do projeto**

- Mesmo Grupo e Repositório do Projeto 1
- Criar pasta projeto2 no repositório
- Escrever documentação descrevendo os resultados em um arquivo GitHub Flavored Markdown
- Criar arquivo relatorio.md dentro da pasta projeto2. Conteúdo do Relatório:
    - Nomes e RAs dos Alunos.
    - Descrição da Implementação (formato de cabeçalho, mensagens utilizadas, os valores dos tempos de espera, protocolo baseado em janela, etc).
    - Dificuldades que você enfrentou e como você as resolveu.
- Prazo Final de Entrega do Código (até 8 dezembro 2015)
- Não serão considerados commits após essa data

BONUS: 10% a mais na nota do projeto 2 – Valendo 110%

- Trabalhar com uma JANELA ADAPTIVA USANDO O ALGORITMO DE CONGESTIONAMENTO DO TCP RENO.