PONTIFÍCIA UNIVERSIDADE DE CAMPINAS PUC - CAMPINAS

CURSO DE ENGENHARIA DE COMPUTAÇÃO

REDES B

CAMADA DE ENLACE

Professor: Edmar Rezende

Data: 01/09/2015

Luiz Henrique Cruz Junior R.A: 12110797

Matheus F. Nishida R.A: 12212692

Vinicius A. Marques R.A: 12643748

SUMÁRIO

1. INTRODUÇÃO………………………………………………………………………… 2
2. ESPECIFICAÇÃO DOS PROTOCOLOS……………………………………….. 3
3. PLANO DE IMPLEMENTAÇÃO………………………………………………… 4
4. PLANO DE INTEGRAÇÃO E TESTE…………………………………………… 5
5. COMPILAÇÃO E EXECUÇÃO DO PROGRAMA…………………………. 6
6. CRONOGRAMA E DIVISÃO DE RESPONSABILIDADES……………… 7
7. CONSIDERAÇÕES FINAIS……………………………………………………….. 8

**1. INTRODUÇÃO**

Nesta primeira parte será implementada a camada de ligação de dados, também conhecida como camada de enlace de dados ou camada de link de dados, que é uma das sete camadas do [modelo OSI](https://pt.wikipedia.org/wiki/Modelo_OSI). Esta camada detecta e, opcionalmente, corrige erros que possam acontecer na [camada física](https://pt.wikipedia.org/wiki/Camada_f%C3%ADsica). É responsável pela transmissão e recepção (delimitação) de quadros e pelo controle de fluxo. Ela também estabelece um protocolo de comunicação entre sistemas diretamente conectados.

**2. ESPECIFICAÇÃO DOS PROTOCOLOS**

* Os datagramas são tratados como structs
* O tamanho dos dados é de 1024 bytes
* Usa-se threads e mutex para concorrência e exclusão mútua
* O protocolo de envio é o UDP

Diagrama:



**3. PLANO DE IMPLEMENTAÇÃO**

Para a implementação da camada de enlace foram usadas duas threads, uma para envio e outra para recebimento.

Também são disparadas threads produtoras e consumidoras. O produtor do enlace.c recebe o pacote e o consumidor do enlace envia o pacote, já o produtor do teste.c salva o que será enviado em um buffer e o consumidor recebe e lê o buffer.  
  
O projeto é composto pelos seguintes arquivos:  
**cliente.c**  
Salva a mensagem a ser enviada em um buffer.

**servidor.c**  
Espera as mensagens a serem enviadas na porta 5000

**projeto.c**  
Contém a main(), inicia os mutexes e os enlaces.

**enlace.c**

Possui as threads de envio e recebimento e também as produtoras e consumidoras.

**teste.c**  
Faz o teste de envio e recebimento através do conceito de produtores e consumidores.

**garbler.c**

Tem a função de simular o envio de um pacote corrompido, cuja finalidade é atestar a implementação do checksum.

**4. PLANO DE INTEGRAÇÃO E TESTE**

**Teste 1 - Checksum**  
Realizar a verificação se os bytes enviados são iguais aos recebidos, é importante ressaltar que nesta etapa que ocorre eventuais detecções de erros.

**Teste 2 - Nós vizinhos**  
Fazer envio a um nó vizinho e confirmar o recebimento.

**Teste 3 - Nós não vizinhos**  
Fazer envio a um nó não vizinho e verificar se não é recebido.

**Teste 4 - Comparar MTU**  
Comparar o tamanho do frame com o MTU para evitar que ultrapasse o tamanho máximo.

**5. COMPILAÇÃO E EXECUÇÃO DO PROGRAMA**

O conceito de de programa objeto foi utilizado para a compilação e execução, onde o programa é dividido em partes e depois compilado todo junto.

A compilação é dada por: gcc projeto.o teste.o enlace.o garbler.o -o projeto.exe -lpthread

**6. CRONOGRAMA E DIVISÃO DE RESPONSABILIDADES**

|  |  |  |  |
| --- | --- | --- | --- |
| **INTEGRANTES** | **SEMANA 1** | **SEMANA 2** | **SEMANA 3** |
| **LUIZ HENRIQUE** | Especificação do projeto | Programação | Programação/Testes/Relatório |
| **MATHEUS NISHIDA** | Especificação do projeto | Programação | Programação/Testes/Relatório |
| **VINICIUS MARQUES** | Especificação do projeto | Programação | Programação/Testes/Relatório |

**7. CONSIDERAÇÕES FINAIS**

Conclui-se que a camada de enlace tem importância fundamental na hierarquia de redes. É nela que é verficada a presença/ausência de ligação entre nós vizinhos, também é responsável por enviar quadros e verificar a ocorrência de erros. Durante o desenvolvimento foram encontradas algumas dificuldades, as principais foram a organização do tempo e tarefas entre os membros do grupo, que deverão ser melhores distribuídas na próxima etapa.