PONTIFÍCIA UNIVERSIDADE DE CAMPINAS PUC - CAMPINAS

CURSO DE ENGENHARIA DE COMPUTAÇÃO

REDES B

CAMADA DE REDE

Professor: Edmar Rezende
Data: 06/10/2015

Luiz Henrique Cruz Junior R.A: 12110797
Matheus F. Nishida R.A: 12212692
Vinicius A. Marques R.A: 12643748

SUMÁRIO

1. INTRODUÇÃO

A **camada de rede** é responsável por controlar a operação da rede de um modo geral. Suas principais funções são o roteamento dos pacotes entre fonte e destino, mesmo que estes tenham que passar por diversos nós intermediários durante o percurso, o controle de congestionamento e a contabilização do número de pacotes ou bytes utilizados pelo usuário, para fins de tarifação.

O principal aspecto que deve ser observado nessa camada é a execução do roteamento dos pacotes entre fonte e destino, principalmente quando existem caminhos diferentes para conectar entre si dois nós da rede. Em redes de longa distância é comum que a mensagem chegue do nó fonte ao nó destino passando por diversos nós intermediários no meio do caminho e é tarefa do nível de rede escolher o melhor caminho para essa mensagem.

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

2.1 COMPOSIÇÃO DOS PACOTES:

```
struct datagram
{
        int tam buffer;
        int no_envio;
        char buffer[100];
        int checksum;
        int controle;
        int offset;
        int id;
        int no_vizinho;
        int no_inicio;
        int no_prox;
        int frag;
        union tabela dados;
};
```

2.2 TABELA DE ROTAS:

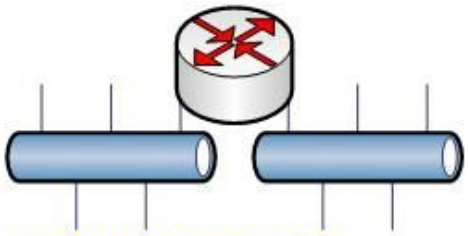
Quando um pacote chega em uma das interfaces do roteador, ele analisa a sua tabela de roteamento, para verificar se na tabela de roteamento, existe uma rota para a rede de destino. Pode ser uma rota direta ou então para qual roteador o pacote deve ser enviado.

```
struct tabela_de_rotas
{
	int no_atual;
	int destino;
	int custo;

};


union tabela
{
	struct tabela_de_rotas tabela_rotas[6];
};
```

2.3 FRAGMENTAÇÃO

A fragmentação de pacotes está relacionada à Maximum Transfer Unit (MTU), que especifica a quantidade máxima de dados que podem passar em um pacote por um meio físico da rede. Caso um pacote tenha tamanho superior ao suportado pelo meio físico da rede, esse é fragmentado, ou seja, dividido.
Os fragmentos resultantes trafegam pela rede e, quando chegam ao seu destino final, são reagrupados, com base em offsets, reconstituindo, assim, o pacote original.

Diagrama:



Roteamento entre LANs
O roteador filtra o tráfego de um segmento de LAN (subnet) para outro para balancear o tráfego e implementar políticas de gerenciamento.

Acesso Remoto
Roteamento são largamente utilizados para acesso remoto convertendo pacotes de LAN em estruturas de dados compatíveis com as redes geograficamente distribuídas (WAN).

3. PLANO DE IMPLEMENTAÇÃO

Para a implementação da camada de rede foram usadas três threads, uma para envio exclusivo da tabela de rotas, outra para receber e atualizar a tabela e outra para envio. Também são disparadas threads produtoras e consumidoras.

O projeto é composto pelos seguintes arquivos:

enlace.c
Código da camada de enlace, para o envio à outra camada de enlace.

garbler.c
Módulo adulterador de pacotes. O adulterador irá introduzir erros.

projeto.c
Contém as threads que iniciam as camadas de enlace, de rede e a de transporte que não está totalmente implementada mas serve de teste.

rede.c
Cria, envia e recebe a tabela de rotas. Envia e recebe datagramas. Fragmenta e desfragmenta.

transporte.c
Serve como teste para a camada de rede.

4. PLANO DE INTEGRAÇÃO E TESTE

Teste 1 - Nós não vizinhos

Fazer envio a um nó não vizinho e consequentemente aprender sua rota.

Teste 2 - Fragmentação

Fragmentar o pacote caso o tamanho seja maior que o MTU.

Teste 3 - Desfragmentação

Desfragmentar o pacote quando o mesmo for recebido.

Teste 4 - Atualizar tabela de rotas

Conferir se a tabela de rotas foi atualizada com novos caminhos.

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.

Os seguintes programas são executados:

```
gcc -c rede.c -o rede.o
gcc -c enlace.c -o enlace.o
gcc -c projeto.c -o projeto.o
gcc -c garbler.c -o garbler.o
gcc -c transporte.c -o transporte.o
```

A compilação é dada por: `gcc projeto.o rede.o enlace.o garbler.o transporte.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 rede tem papel fundamental na hierarquia de redes. Ela recebe serviço da camada de enlace e presta serviço a camada de transporte, é ela que roteia os pacotes, fragmenta e os desfragmenta também. A camada foi extremamente complicada de se implementar, mas foi importante para o melhor conhecimento sobre o funcionamento da mesma.