Simulador de Redes

Mauricio Osaki
Yoshio Mori

Ambiente

Sistema Operacional:

- Debian GNU/Linux 8

Hardware:

- 3.20 GHz de processador
- 8 GB de memória

Linguagem de Programação:

- Python3

Host ircc UDP

- Deve ter uma variável global para nome do host, ip, roteador padrão e dns padrão
- Deve ter duas threads, uma para receber mensagens do simulador (TS), outro para receber mensagens de clientes (TC).
- TS deve receber 2 referências para função: uma que recebe mensagens do simulador, outra para enviar mensagens para a interface.
- TC deve receber 2 referências para função: uma para receber mensagens da interface outra para enviar mensagens para enviar.

Objeto Interface do Host

Atributos:

- O objeto interface tem o buffer de entrada e saída
- Para cada um dos buffer, o objeto tem um mutex que será usado para indicar que há alguma thread inserido dados ou lendo e removendo. Outro mutex será usado para indicar que ainda há dados a serem lidos e removidos.

Objeto Interface Host

Métodos:

- Para cada um dos buffers, existe métodos de inserção de dados e leitura e remoção.
- O método de inserção de dados no buffer de entrada e leitura e remoção dos dados no buffer de saída serão usadas pelas Threads do enlace
- O método de inserção de dados no buffer de saída e leitura e remoção dos dados no buffer de entrada serão usadas pelas Threads do host

Thread Router

- Deve receber sua identificação i.
- Deve receber o número de interfaces j
- Deve ter uma tabela T que associa endereço ip a uma das interfaces do roteador ou um ip a outro, essa tabela deve ser visível pela função principal.
- Deve ter uma tabela Conjunto de Interfaces que associa o i a um T, essa tabela deve ser visível pela função principal.
- Cada router deve ter um buffer de entrada.
- Faz isso:

1. Inicializa T como um array
2. push em T um Objeto Interface do router

Thread Router

3. Se o número de objetos em h < j, vá para 2.
4. Adiciona (i, T) na tabela Conjunto de Interfaces.
5. Inicializa uma Condition com o tem_entrada, um predicado que informa se o buffer de entrada tem datagrama
6. Com o condition, aguarda até ter datagrama no buffer de entrada
7. Pop do buffer de entrada
8. Do datagrama, pego o ip destino

Thread Router

9. uso o ip destino na tabela de roteamento com a máscara 255.255.255.0
10. se o valor associado for uma porta, faça o envio nessa porta
11. senão se o valor associado for um ip, faça ip destino ← valor associado e vá para 9.
12. volta para 7.

Objeto Interface do Router

- Deve ser inicializado com um buffer de entrada passado como atributo, que servirá como repositório para todos os datagramas que chegarem pelas interfaces

Atributo:

- Buffer de saída, que será usado para enviar dados pelos enlaces

Métodos:

- tem_saída, usado pelo enlace para saber se tem datagrama no bufer de saída da interface

Thread Enlace

- Uma thread é usada para enviar datagrama da origem para o destino e a outra do destino para a origem
- Cada thread tem uma referência para o método push da fila da máquina destino, uma referência para o método pop da fila da máquina origem e um predicado que informa se chegou novo dado na interface de saída da máquina de origem.
- Deve receber como parâmetro a velocidade de transferência e o tempo em milissegundos do atraso.

Thread Enlace

- Cada uma das threads fica fazendo isso:

1. Inicializa o condition de novo dado da interface da máquina de origem
2. Aguarda a máquina origem ter novo dado:
    a. Com o condition, espere a interface da máquina de origem ter dados para enviar:
        i. Espere D segundos para representar o delay.
        ii. Enquanto tem dados para enviar, faça:
            1. Pop do datagrama da interface de origem
            2. Espere T segundos para representar a taxa de envio
            3. Push do datagrama na interface de destino
3. Volte para 2.

T segundos

- Depende da velocidade de transmissão do enlace
- Depende do tamanho do pacote sendo transferido
- Depende do tempo de simulação em cada iteração, pois em cada iteração da Thread Enlace um pacote é transferido.
- A questão é: Como definir a resolução de tempo?
- Quanto menor a resolução do tempo mais preciso a simulação e maior será o tempo de execução, quanto maior a resolução do tempo menos preciso a simulação e menor será o tempo de execução
- Solução: deixar como parâmetro de execução

Thread Relógio

- O tempo deve ditar o transito dos pacotes, para isso, a Thread Relógio deve esperar um pacote ser processado, quando houver um.
- Deve prover um parâmetro de tempo para as iterações dos laços de todas as threads do enlaces
- Deve receber uma barreira para sincronizar todos os laços de todos as threads com o relógio.

Thread Relógio

- E faz isso:

1. Inicializa o tempo t com 0
2. tempo_por_iteração ← P / M, onde P é o tamanho máximo do pacote
3. t ← t + tempo_por_iteração
4. vá para 3.

- t é um objeto timedelta que guarda valores no formato de tempo

Datagrama no simulador

- Endereço IP de origem 4 B
- Endereço IP de destino 4 B
- Identificação do protocolo 1 B
- Tamanho do cabeçalho IP + Tamanho de todo o resto 2 B
- TLL 1 B

Pacote UDP no simulador

- Porta origem 2 B
- Porta destino 2 B
- Comprimento 2 B

Pacote TCP no simulador

Não implementado