Engenharia de Telecomunicações — Prof. Emerson Ribeiro de Mello

POO29004 – Programação Orientada a Objetos — mello@ifsc.edu.br

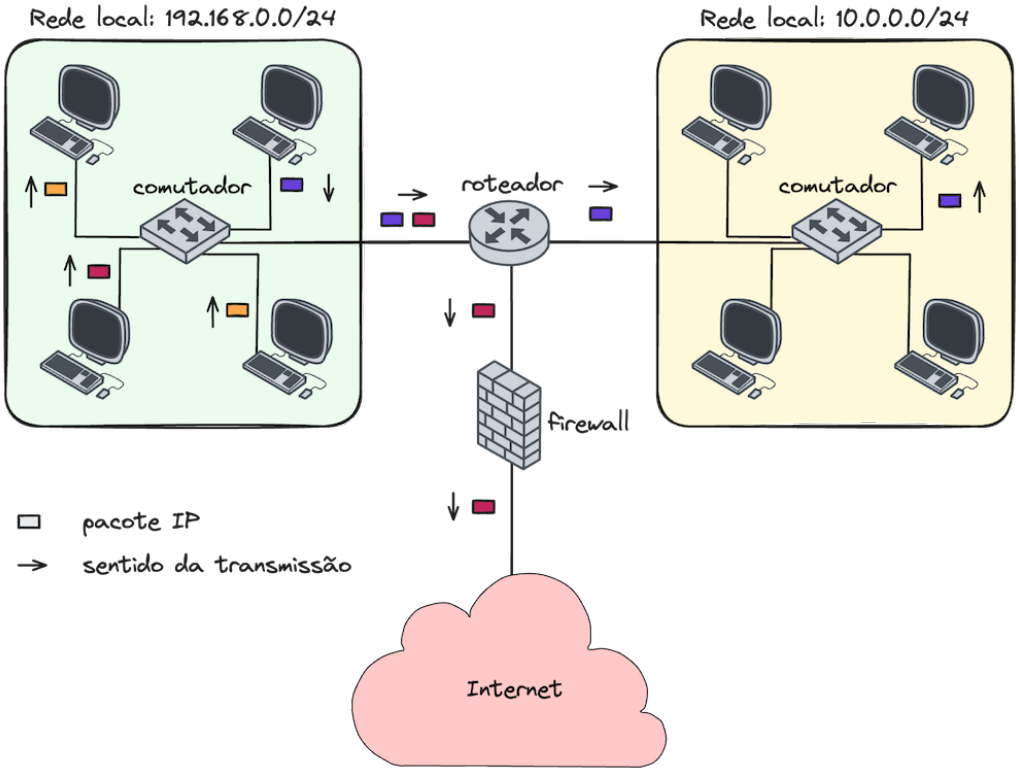
Projeto prático 02: Simulador de dispositivos de rede TCP/IP

05/08/2024

1 Introdução

Nas redes de computadores TCP/IP as informações a serem transmitidas, de uma origem para um destino, são empacotadas em pacotes IP e então transmitidas. Além da informação propriamente dita (*payload*), cada pacote possui um cabeçalho que contém informações como: endereço IP de origem, porta de origem, endereço IP destino, porta de destino, entre outras. Cabe frisar que no caminho entre o dispositivo origem até o destino, podem existir dispositivos como comutadores (*switchs*), roteadores (*routers*) e *firewalls* que podem processar, encaminhar ou descartar os pacotes.

Figura 1: Exemplo de topologia de rede TCP/IP



Na Figura 1 é apresentado um exemplo de topologia de rede TCP/IP. Nesta topologia, existem duas redes locais, cada uma com 4 dispositivos *endpoint* (computadores) e um comutador (*switch*). Os comutadores estão interligados por um roteador (*router*), o qual também está interligado a um *firewall*. Neste exemplo é ilustrado diferentes pacotes IP sendo transmitidos entre os dispositivos. O pacote laranja é trocado entre dispositivos dentro de uma mesma rede. O pacote roxo é trocado entre dispositivos de redes diferentes, sendo encaminhado pelo roteador. O pacote magenta é trocado entre dispositivos de redes diferentes, sendo encaminhado pelo roteador e processado pelo *firewall*, que o encaminha para a Internet.

Ao receber um pacote IP cada dispositivo de rede deve processar o pacote de acordo com as regras de sua camada de rede. O *endpoint* deve processar o pacote de acordo com a aplicação, o comutador deve encaminhar o pacote para o dispositivo correto, o roteador deve encaminhar o

pacote para a rede correta ou descartá-lo caso a rede de destino não seja conhecida e o *firewall* deve processar o pacote de acordo com as regras de filtragem.

2 Objetivo

O objetivo deste projeto é desenvolver um simulador em Java que seja capaz de simular o comportamento de um comutador, roteador e *firewall* ao receber pacotes IP. O comportamento esperado de cada dispositivo é descrito a seguir:

- **Comutador** – deve manter uma tabela de endereços MAC e portas associadas (comutador tem 12 portas). Ao receber um pacote IP, o comutador deve encaminhar o pacote para o dispositivo correto, ou seja, para a porta associada ao endereço MAC de destino do pacote. Caso o endereço MAC de destino não seja conhecido, o pacote deve ser encaminhado para todas as portas, exceto a porta de origem;
- **Roteador** – deve manter uma tabela de rotas com as redes conhecidas e a porta de saída associada. Ao receber um pacote IP, o roteador deve encaminhar o pacote para a rede correta de acordo com o endereço IP de destino do pacote. Caso a rede de destino não seja conhecida, o pacote deve ser encaminhado para a porta de saída associada à rota padrão (geralmente a porta de saída para a Internet). Caso o pacote seja destinado para a Internet, o roteador deve encaminhar o pacote para o *firewall*;
- **Firewall** – ao receber um pacote IP, o *firewall* deve processar o pacote de acordo com as regras de filtragem. Cada regra é composta por um endereço IP de origem, uma porta de origem, um endereço IP de destino, uma porta de destino e uma ação (encaminhar ou descartar). As regras devem ser processadas na ordem em que foram adicionadas. Caso uma regra seja satisfeita, o pacote deve ser encaminhado ou descartado de acordo com a ação da regra. Caso nenhuma regra seja satisfeita, o pacote deve ser descartado.

Para simplificar a implementação, considere que cada dispositivo possui um endereço MAC, no caso uma letra e um número (exemplo: A1, B2, C3, etc.). O endereço MAC é único para cada dispositivo e é utilizado para identificar o dispositivo na rede local. Além disso, considere que cada dispositivo possui um endereço IP (único por dispositivo), no caso um número de 4 dígitos (exemplo: 1203, 2122, 3412, 6621, etc). O endereço de rede é composto pelo primeiro dígito do endereço IP (exemplo: 1xxx, 2xxx, 3xxx, 6xxx, etc). Considere também que um pacote IP é composto por um endereço IP de origem, uma porta de origem, um endereço IP de destino, uma porta de destino, um endereço MAC de destino e um conteúdo (*payload*). O conteúdo do pacote pode ser uma cadeia de caracteres (exemplo: "Olá, mundo!").

2.1 Aplicação a ser implementada

Ao executar a aplicação, a topologia de rede deve ser configurada conforme a Figura 1. A aplicação deve apresentar um menu com as seguintes opções:

- **Listar dispositivos por tipo** – lista dispositivos da rede agrupados por tipo (comutadores, roteadores e *firewall*). Para cada dispositivo, deve ser exibido o endereço MAC e IP;
- **Listar dispositivos por rede** – lista dispositivos da rede agrupados por rede. Para cada dispositivo, deve ser exibido o endereço MAC e IP;
- **Listar rotas** – exibe a tabela de rotas do roteador. Para cada rota, deve ser exibido o endereço de rede e a porta de saída associada;

- **Listar regras de filtragem** – exibe a lista de regras de filtragem do *firewall*. Para cada regra, deve ser exibido o índice, endereço IP de origem, porta de origem, endereço IP de destino, porta de destino e ação (encaminhar ou descartar);
- **Criar regra de filtragem** – permite criar uma nova regra de filtragem. Deve ser informado o endereço IP de origem, porta de origem, endereço IP de destino, porta de destino e ação (encaminhar ou descartar);
- **Apagar regra de filtragem** – permite apagar uma regra de filtragem. Deve ser informado o número (índice na lista de regras) da regra a ser apagada;
- **Simular processamento de pacote** – O usuário deve informar o endereço IP de origem, porta de origem, endereço IP de destino, porta de destino, endereço MAC de destino e o conteúdo do pacote. A aplicação deve simular o processamento do pacote pelos dispositivos da rede e exibir (imprimir no *console*) o comportamento de cada dispositivo (comutador, roteador e *firewall*) ao receber o pacote IP.
  - **Comutador** – deve indicar se o pacote foi encaminhado para a porta correta, onde encontra-se o *endpoint* destino, ou se foi encaminhado para todas as portas;
  - **Roteador** – deve indicar se o pacote foi encaminhado para uma das redes locais ou se foi encaminhado para a Internet, onde encontra-se o *firewall*;
  - **Firewall** – deve indicar se o pacote foi encaminhado ou descartado e imprimir o índice da regra que foi satisfeita.
- **Sair** – encerra a aplicação.

3 Entregas e requisitos para desenvolvimento da solução

1. Modelagem UML
   - ☐ Diagrama de classes salvo em um arquivo chamado `modelagem.png` na raiz do repositório (ou no Readme.md caso faça com o *Mermaid*).
   - ☐ Notação correta de associação entre classes, representação de atributos e métodos
   - ☐ Herança, devendo representar corretamente classes e métodos abstratos e interfaces
2. Implementação
   - ☐ Encapsulamento, responsabilidade única e divisão de responsabilidades
   - ☐ Uso correto dos conceitos de Herança e Polimorfismo com sobrescrita de métodos
   - ☐ Uso de enumerações e coleções
   - ☐ Comportamento correto das classes modeladas
   - ☐ Comportamento correto da classe que possui método `main`.
3. Projeto Java com Gradle
   - ☐ Arquivo `.gitignore` adequado ao projeto
   - ☐ Possível compilar e executar o projeto via `./gradlew -q run`. Conceito 0 caso não seja possível.
4. Arquivo `Readme.md` na raiz do repositório
   - ☐ Instruções para criar uma nova regra de filtragem, um novo pacote IP e simular o processamento do pacote
   - ☐ Indicar quais funcionalidades foram implementadas e quais não foram

Plágio não é tolerado

Você deve ser o único(a) responsável por fazer a entrega para essa atividade. Todo o código ou texto deverá ser produzido exclusivamente por você, exceto trechos de códigos que possam ter sido fornecidos como parte do enunciado.

Você pode discutir com outros estudantes com o intuito de esclarecer pontos, isso é até incentivado, porém você não poderá copiar trechos de códigos, textos ou soluções de qualquer fonte (e.g. colegas da mesma turma ou de turmas anteriores, repositórios de códigos na Internet ou soluções providas por serviços como Copilot e ChatGPT).