



Escola de Engenharia
Curso de Bacharelado em Engenharia de Sistemas

DCC218 - Introdução a Sistemas Computacionais
Relatório do TP 1 - Parte 1

Raphael Henrique Braga Leivas - 2020028101

Belo Horizonte
12 de outubro de 2025

SUMÁRIO

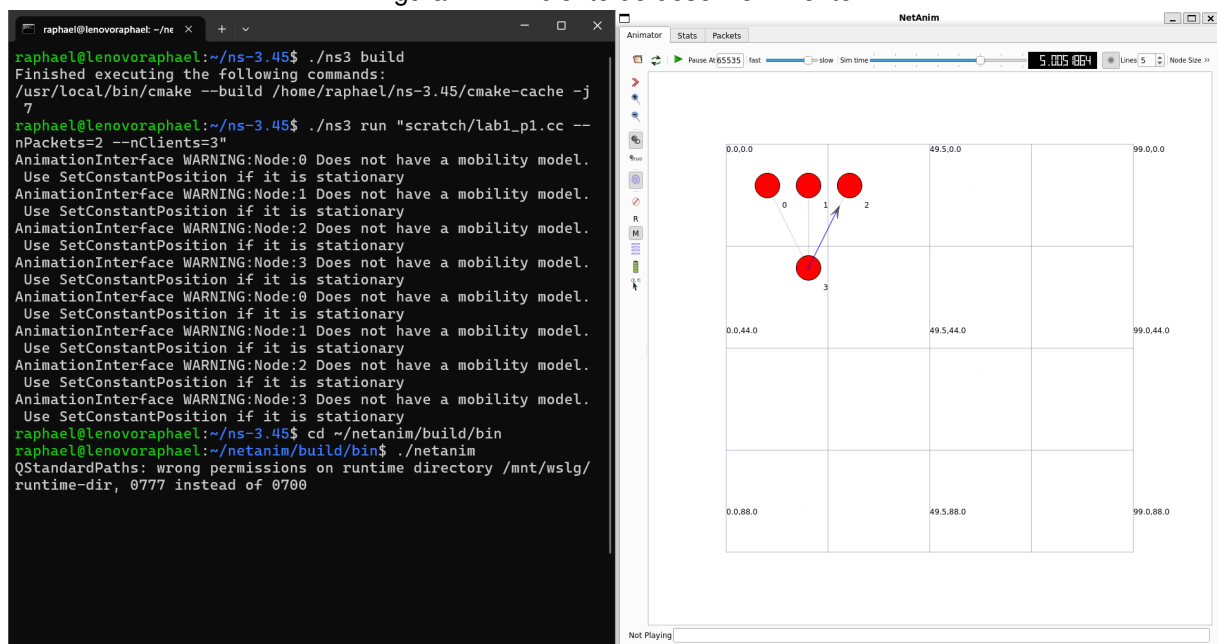
1	AMBIENTE DE DESENVOLVIMENTO	3
2	PARTE 1: COMUNICAÇÃO PONTO-A-PONTO	4
3	PARTE 2: REDE ETHERNET	7
4	PARTE 3: REDE WIFI	9
5	CONCLUSÃO	10
REFERÊNCIAS		11

1 AMBIENTE DE DESENVOLVIMENTO

O projeto é desenvolvido em um notebook Lenovo com processador Intel-Core i5 de 11ª geração, com arquitetura x64. A frequência do processador é 2.4 GHz e possui 4 cores. O processador possui 8 GB de memória RAM disponível. O sistema operacional do computador é Windows 11, mas o ambiente NS-3 é instalado através de WSL (Windows Subsystem for Linux) Ubuntu 22.04.1 LTS.

O ambiente NS-3 é instalado na pasta `home` do WSL. A Figura 1 ilustra o ambiente de desenvolvimento utilizado no projeto.

Figura 1 – Ambiente de desenvolvimento.



Fonte: Elaboração própria.

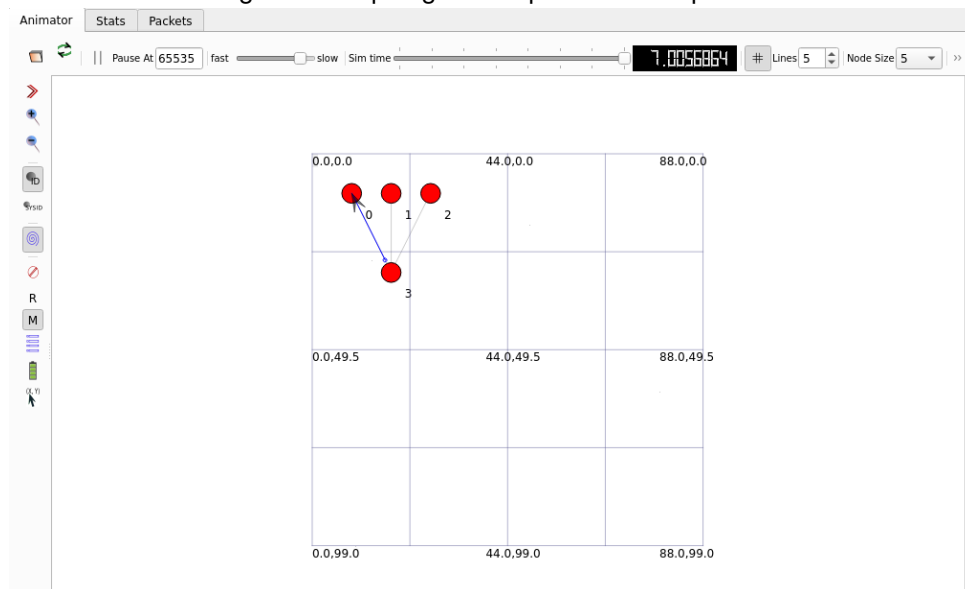
Essencialmente, é adotado um procedimento de duas etapas:

1. O comando `./ns3 build`, seguido de `./ns3 run` com o programa e argumentos de linha de comando desejados, é usado para executar o script C++ com as funções do NS-3. A saída dos comandos será um arquivo XML;
2. O comando `./netanim` é usado para visualizar o arquivo XML exportado na etapa 1, obtendo a animação exibida no lado direito da Figura 1.

2 PARTE 1: COMUNICAÇÃO PONTO-A-PONTO

Para a parte 1, o objetivo é estabelecer conexões ponto-a-ponto entre clientes e um servidor. Usando `nPackets = 2` e `nClients = 3` nos argumentos da linha de comando, é obtido a topologia exibida na Figura 2.

Figura 2 – Topologia do experimento da parte 1.



Fonte: Elaboração própria.

A Figura 3 mostra os envios dos pacotes dos nós clientes para o nó servidor ao longo do tempo, com três clientes e cada um enviando dois pacotes.

É possível observar que o tempo de propagação é muito menor do que o tempo de espera para envio dos pacotes. O tempo de RTT é de 3.69 ms, enquanto 1 segundo é esperado para enviar cada pacote. Em cada envio, o cliente envia para o nó 3 (servidor), e o servidor responde de volta para o nó cliente (0 a 2) que o requisitou.

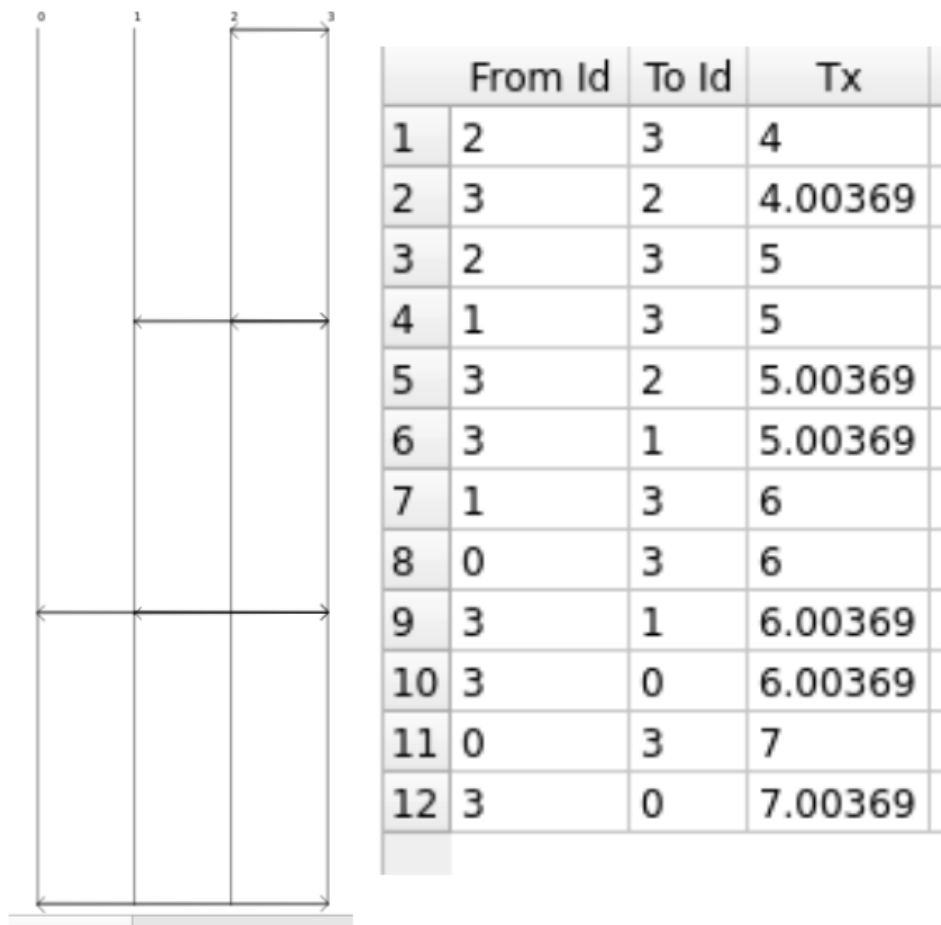
Para tornar o gráfico mais didático, é possível aumentar o tempo de propagação para 200 ms e reduzir a largura de banda para 5 kbps, obtendo o resultado da Figura 4. Vemos que o pacote gasta mais tempo para sair do cliente e chegar ao servidor, e que o servidor demora 1.88 segundos a enviar a resposta ao cliente. Um dos fatores que contribuem para esse tempo é exibido abaixo:

$$T_{prop} + \frac{banda}{tamanho} = 0.2 + \frac{5 \cdot 10^3 \text{ kbps}}{1024 \cdot 8} = 0.81 \text{ s}$$

uma vez que o pacote enviado tem 1024 bytes de tamanho.

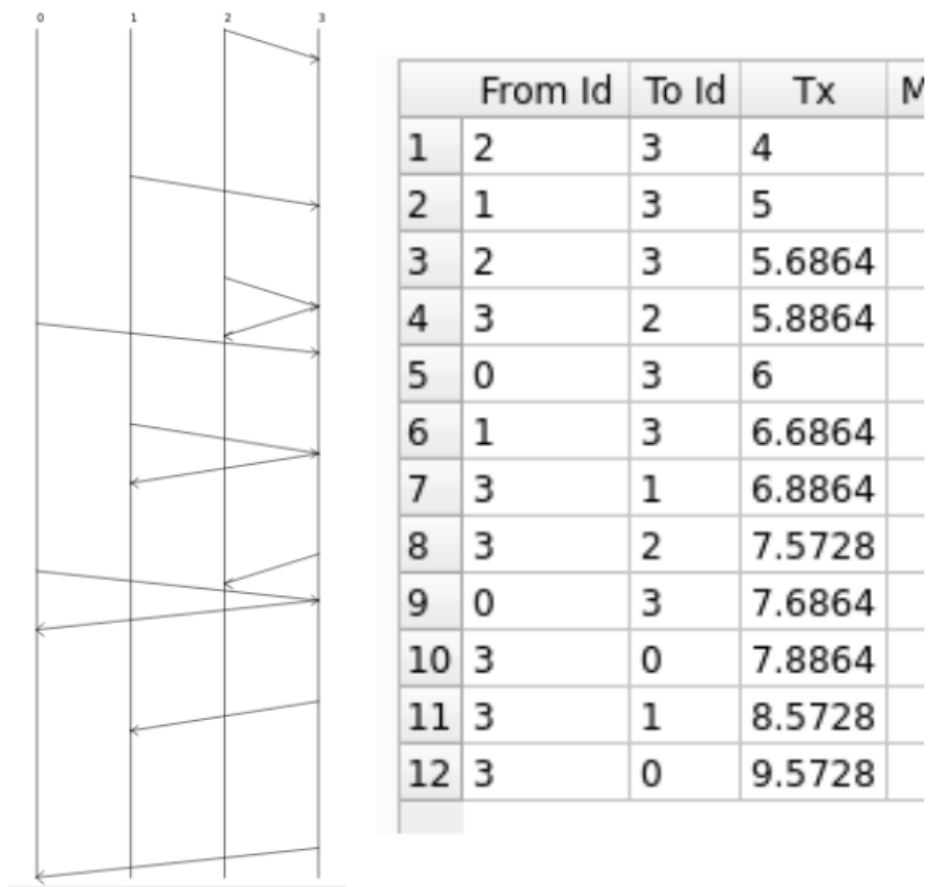
Na pasta compactada no Moodle, está presente um teste com 5 clientes e cada um enviando 4 pacotes, bem como o print com as linhas de comando e arquivo txt.

Figura 3 – Resultado do experimento da parte 1.



Fonte: Elaboração própria.

Figura 4 – Resultado do experimento da parte 1 - tempo de propagação 200 ms e largura de banda 5 kbps.

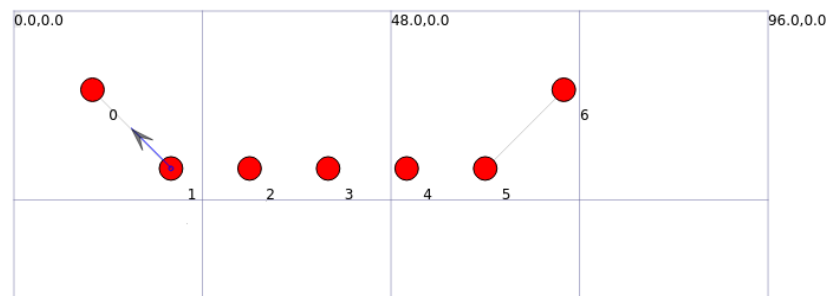


Fonte: Elaboração própria.

3 PARTE 2: REDE ETHERNET

A topologia implementada nesse relatório está exibida na Figura 5. Basicamente, o nó 0 quer se comunicar com o servidor (nó 6). Para isso, ele usa 4 nós CSMA intermediários que enviam a requisição ao servidor final. Nós CSMA utilizam o protocolo Carrier-Sense Multiple-Access (CSMA) que reduz o risco de colisões entre os pacotes enviados simultaneamente por vários nós no mesmo meio.

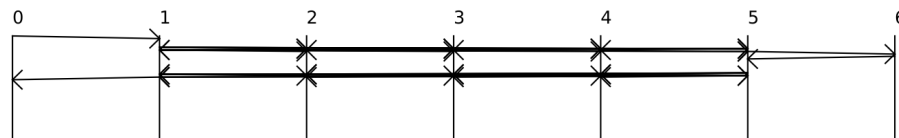
Figura 5 – Topologia da parte 2.



Fonte: Elaboração própria.

Utilizando 4 nós CSMA intermediários enviando 10 pacotes, a sequência de envio dos pacotes em função do tempo está exibida na Figura 6.

Figura 6 – Envio dos pacotes na topologia da parte 2.



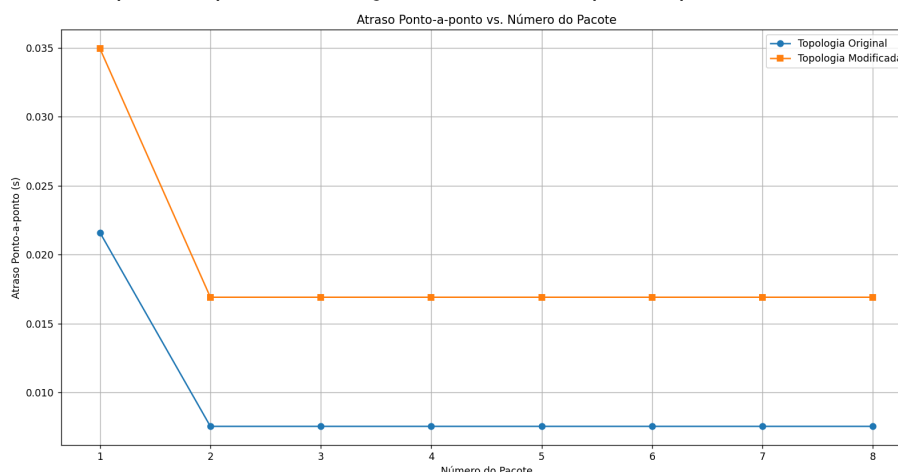
Fonte: Elaboração própria.

O gráfico do atraso de ponto-a-ponto de uma requisição versus o número do pacote está exibido na Figura 7. A topologia original é a presente no exemplo `second.cc` da documentação do NS-3, enquanto a topologia modificada é a exibida na Figura 5 desenvolvida na parte 2. O gráfico foi obtido a partir do seguinte procedimento:

- Modifica os códigos para que eles salvem os atrasos em um arquivo txt com as funções prontas de `TxTrace` e `RxTrace`;
- Carrega os arquivos txt em um arquivo Python e faz os plots com a biblioteca `matplotlib`.

É possível observar que o atraso na topologia original é menor do que na modificada. Isso é conforme esperado, uma vez que a topologia modificada possui um

Figura 7 – Atraso ponto-a-ponto em função do número do pacote para cada uma das topologias.



Fonte: Elaboração própria.

nó a mais, que aumenta o tempo de propagação. O fato de o aumento no atraso ser constante ao longo dos pacotes reforça essa hipótese.

Além disso, o atraso do primeiro pacote é maior do que os demais. Analisando os arquivos PCAP gerados para ambas topologias, vemos que isso é causado pelo tempo de inicialização necessário para estabelecer a conexão, como a criação do socket UDP e conexão com a porta do servidor. A Figura 8 ilustra esse efeito nos arquivos PCAP. Temos um atraso de 0.0349 segundos no primeiro pacote, e 0.0169 segundos nos demais.

Figura 8 – Atraso do primeiro pacote nos arquivos PCAP.

```
raphael@lenovoraphael:~/ns-3.45$ tcpdump -r first_p2p_net-trace-0-1.pcap
reading from file first_p2p_net-trace-0-1.pcap, link-type PPP (PPP), snapshot length 65535
21:00:02.000000 IP 10.1.1.1.49153 > 10.1.3.2.discard: UDP, length 1024
21:00:02.034941 IP 10.1.3.2.discard > 10.1.1.1.49153: UDP, length 1024
21:00:03.000000 IP 10.1.1.1.49153 > 10.1.3.2.discard: UDP, length 1024
21:00:03.016916 IP 10.1.3.2.discard > 10.1.1.1.49153: UDP, length 1024
21:00:04.000000 IP 10.1.1.1.49153 > 10.1.3.2.discard: UDP, length 1024
21:00:04.016916 IP 10.1.3.2.discard > 10.1.1.1.49153: UDP, length 1024
```

Fonte: Elaboração própria.

4 PARTE 3: REDE WIFI

5 CONCLUSÃO

REFERÊNCIAS