MiniProjeto - Medição Ativa em Redes com o Iperf

Avaliação de Desempenho de Sistemas

Arthur Cadore Matuella Barcella

11 de Abril de 2025

Engenharia de Telecomunicações - IFSC-SJ

Sumário

1. Introdução 3
1.1. Especificação de cenário 3
1.2. Topologia: 3
1.3. Objetivo 3
1.3.1. Fatores e níveis 4
1.3.2. Metrica avaliada 4
1.3.3. Execuções: 4
2. Desenvolvimento 4
2.1. Script Shell 4
2.2. Script Makefile 6
2.3. Resultados obtidos 7
3. Conclusão 7
4. Referências 7

1. Introdução

Este relatório tem como objetivo apresentar o desenvolvimento e os resultados obtidos no mini projeto de medição ativa em redes utilizando o Iperf.

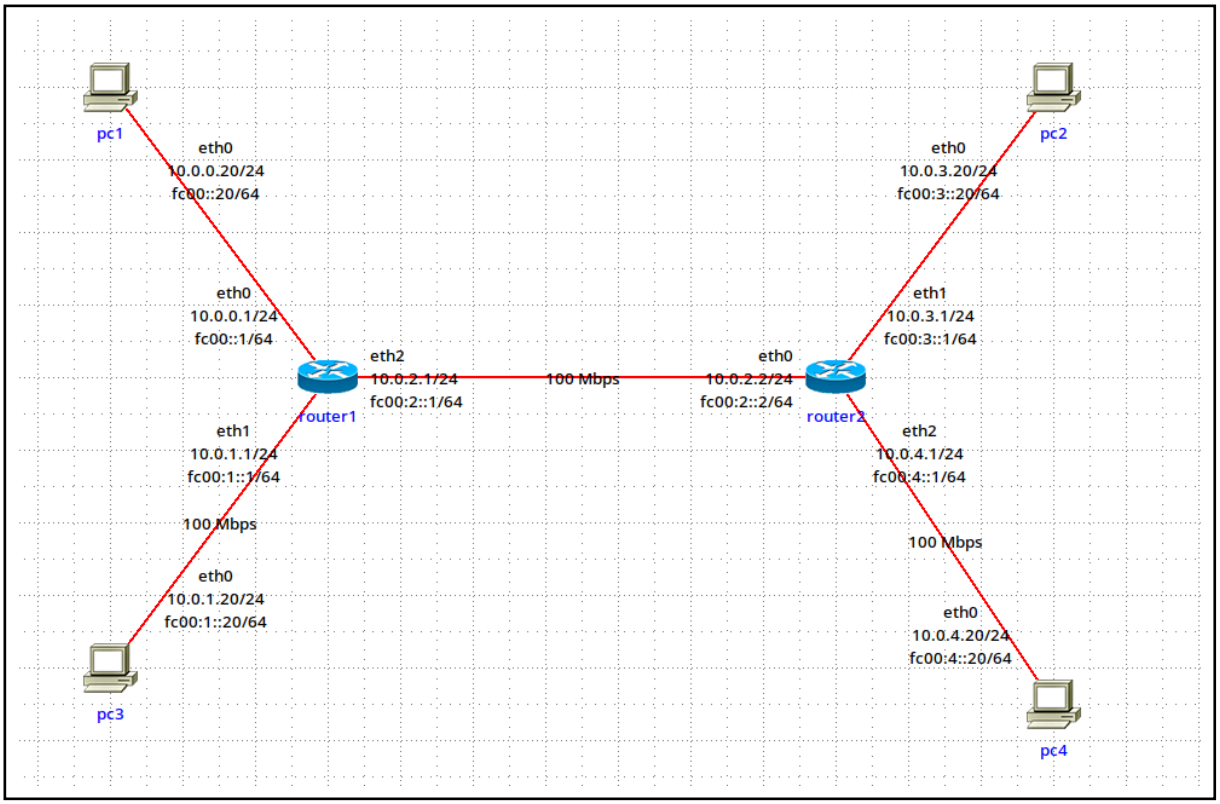
1.1. Especificação de cenário

- Uso da ferramenta iperf;
- Uso da ferramenta imunes;
- Automação de tarefas via scritpt shell e comandos do simulador imunes;
- Conceitos de intervalo de confiança e
- Projeto fatorial 2kr e boas práticas de projeto de experimentos.

1.2. Topologia:

A topologia utilizada no experimento foi a seguinte:

Figura 1: Elaborada pelo Autor



1.3. Objetivo

Avaliar, por meio de medição ativa com iperf, como a vazão de uma conexão TCP é afetada por dois fatores:

- O número de fluxos TCP paralelos (parâmetro -P no cliente);
- O retardo de rede, emulado com o comando vlink.

Além disso, verificar se há interação entre os fatores na determinação da vazão.

1.3.1. Fatores e níveis

Os fatores e níveis utilizados no experimento foram:

Fator	Nivel Baixo	Nivel Alto
A: Paralelismo TCP (-P)	1 fluxo	4 fluxo
B: Retardo de rede (RTT)	10 ms (ida ⟹ RTT20ms)	100 ms (ida ⟹ RTT200ms)

1.3.2. Metrica avaliada

- Vazão total (em Mbps), somando todos os fluxos, medida no cliente ao final da execução.

1.3.3. Execuções:

Os ciclos de execução utilizados foram os seguintes:

Execução	Fluxos (-P)	Delay (ms)	Repetições
1	1	10	8
2	1	100	8
3	4	10	8
4	4	100	8

2. Desenvolvimento

Abaixo está a descrição do desenvolvimento do experimento, incluindo a configuração do ambiente, o script utilizado para automatizar a execução e o Makefile utilizado para facilitar a execução de múltiplas iterações.

2.1. Script Shell

O script shell desenvolvido para automatizar a execução do experimento foi o seguinte. O script é executado em três passsos:

- Primeiramente, ele inicia o simulador IMUNES em background, utilizando o cenário desejado.
- Em seguida, configura os links de acordo com os parâmetros desejados, utilizando o comando vlink.
- Por fim, inicia o servidor iperf nas máquinas desejadas e executa o cliente iperf, coletando os resultados.

```
#!/bin/bash

TOPOLOGY_FILE=scenario.imn
```

```
BANDWIDTH=100000000
# SCENARIO_ID=i2002
# DELAY=10000
# FLUXES=1

# Create a log directory if it doesn't exist
mkdir -p ./log

echo "============================================================"
echo "Starting IMUNES simulation with scenario ID: $SCENARIO_ID"

# Run IMUNES in background and redirect all output to log
sudo imunes -b -e $SCENARIO_ID $TOPOLOGY_FILE > /dev/null
sleep 2

echo "============================================================"
echo "Simulation started, applying commands..."

# Link configurations
sudo vlink -bw $BANDWIDTH -dly $DELAY router1:pc1@$SCENARIO_ID > /dev/null
sudo vlink -bw $BANDWIDTH -dly $DELAY router1:pc3@$SCENARIO_ID > /dev/null
sudo vlink -bw $BANDWIDTH -dly $DELAY router2:pc2@$SCENARIO_ID > /dev/null
sudo vlink -bw $BANDWIDTH -dly $DELAY router2:pc4@$SCENARIO_ID > /dev/null
sudo vlink -bw $BANDWIDTH -dly $DELAY router2:router1@$SCENARIO_ID > /dev/null
sleep 2

# Check status (optional)
sudo vlink -s router1:pc1@$SCENARIO_ID
sudo vlink -s router1:pc3@$SCENARIO_ID
sudo vlink -s router2:pc2@$SCENARIO_ID
sudo vlink -s router2:pc4@$SCENARIO_ID
sudo vlink -s router2:router1@$SCENARIO_ID
sleep 2

# Start iperf servers
sudo himage pc2@$SCENARIO_ID iperf -s &> /dev/null &
sudo himage pc4@$SCENARIO_ID iperf -s &> /dev/null &

# Generate background UDP traffic (mute output)
sudo himage pc1@$SCENARIO_ID iperf -c 10.0.3.20 -u -t 100000 -b 10M &> /dev/null &
sleep 2


# Run TCP test (this is the one you want to see)
echo "============================================================"
echo "Running TCP test between PC3 and PC4..."
sudo himage pc3@$SCENARIO_ID iperf -c 10.0.4.20 -n 100M -P $FLUXES -i 1

# Stop simulation
echo "============================================================"
echo "Stopping IMUNES simulation with scenario ID: $SCENARIO_ID"
sudo imunes -b -e $SCENARIO_ID > /dev/null
```

```
56 echo "============================================================"
57 echo "Simulation stopped"
```

Para executa-lo, 3 variáveis são necessárias junto com o script:

- SCENARIO_ID: ID do cenário a ser utilizado.
- DELAY: Retardo de rede a ser utilizado.
- FLUXES: Número de fluxos TCP a serem utilizados.

Essas variáveis podem ser configuradas previamente no OS, ou então descomentar as linhas 3, 4 e 5 do script e definir os valores desejados.

2.2. Script Makefile

O script make utilizado para automatizar a execução do experimento foi o seguinte. O objetivo da utilização de um Makefile é passar as variáveis definidas no script shell como parâmetro e então executar o script shell com os parâmetros desejados.

```
1  # Caminho do script shell
2  SCRIPT=./run.sh
3
4  # Caminho do diretório de log
5  LOGDIR=./log
6
7  # Targets
8  all: exec1 exec2 exec3 exec4
9
10 exec1:
11   @echo "ETAPA1: Executando com DELAY=10000 FLUXES=1" | tee -a $(LOGDIR)/output.log
12   @for i in $$(seq 1000 1008); do \
13     echo ">> Execução $$i da ETAPA1" | tee -a $(LOGDIR)/output.log; \
14     DELAY=10000 FLUXES=1 SCENARIO_ID=$$i $(SCRIPT) 2>&1 | tee -a $(LOGDIR)/output.log; \
15   done
16
17 exec2:
18   @echo "ETAPA2: Executando com DELAY=100000 FLUXES=1" | tee -a $(LOGDIR)/output.log
19   @for i in $$(seq 1000 1008); do \
20     echo ">> Execução $$i da ETAPA2" | tee -a $(LOGDIR)/output.log; \
21     DELAY=100000 FLUXES=1 SCENARIO_ID=$$i $(SCRIPT) 2>&1 | tee -a $(LOGDIR)/output.log; \
22   done
23
24 exec3:
25   @echo "ETAPA3: Executando com DELAY=10000 FLUXES=4" | tee -a $(LOGDIR)/output.log
26   @for i in $$(seq 1000 1008); do \
27     echo ">> Execução $$i da ETAPA3" | tee -a $(LOGDIR)/output.log; \
28     DELAY=10000 FLUXES=4 SCENARIO_ID=$$i $(SCRIPT) 2>&1 | tee -a $(LOGDIR)/output.log; \
29   done
30
```

```
exec4:
  @echo "ETAPA4: Executando com DELAY=100000 FLUXES=4" | tee -a $(LOGDIR)/output.log
  @for i in $$(seq 1000 1008); do \
    echo ">> Execução $$i da ETAPA4" | tee -a $(LOGDIR)/output.log; \
    DELAY=100000 FLUXES=4 SCENARIO_ID=$$i $(SCRIPT) 2>&1 | tee -a $(LOGDIR)/output.log; \
  done

.PHONY: all exec1 exec2 exec3 exec4
```

2.3. Resultados obtidos

Parar extrair apenas os logs de interesse, o seguinte comando pode ser utilizado:

```
# comando de filtragem:
awk '/^Running TCP test between/,/^=+/' seu_arquivo_de_log.txt > filtrado.txt
```

3. Conclusão

4. Referências