



**INSTITUTO
FEDERAL**

Santa Catarina

Câmpus
São José

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.

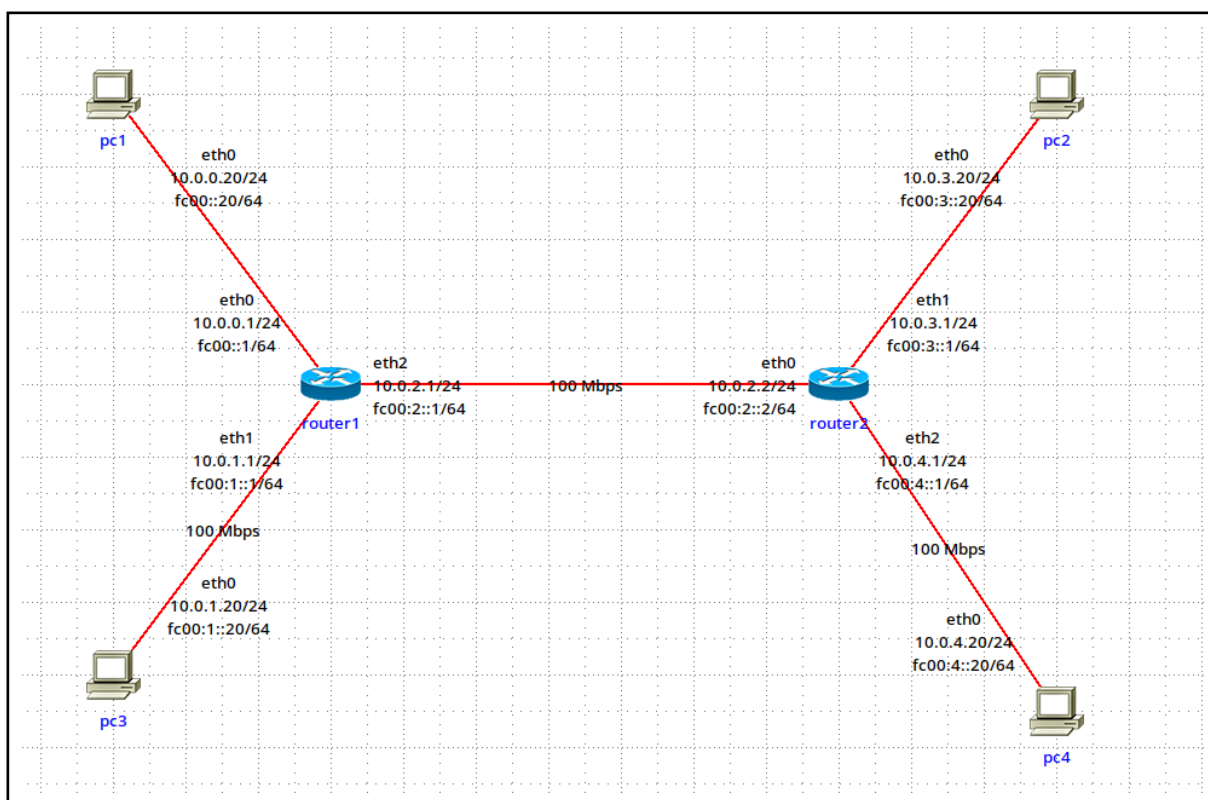
1.1. Especificação de cenário

- Uso da ferramenta iperf;
- Uso da ferramenta imunes;
- Automação de tarefas via script 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	Nível Baixo	Nível Alto
A: Paralelismo TCP (-P)	1 fluxo	4 fluxo
B: Retardo de rede (RTT)	10 ms (ida \Rightarrow RTT20ms)	100 ms (ida \Rightarrow 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 passos:

- 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.

```
1 #!/bin/bash
2
3 TOPOLOGY_FILE=scenario.imn
```

```

4  BANDWIDTH=100000000
5  # SCENARIO_ID=i2002
6  # DELAY=10000
7  # FLUXES=1
8
9  # Create a log directory if it doesn't exist
10 mkdir -p ./log
11
12 echo "=====
13 echo "Starting IMUNES simulation with scenario ID: $SCENARIO_ID"
14
15 # Run IMUNES in background and redirect all output to log
16 sudo imunes -b -e $SCENARIO_ID $TOPOLOGY_FILE > /dev/null
17 sleep 2
18
19 echo "=====
20 echo "Simulation started, applying commands..."
21
22 # Link configurations
23 sudo vlink -bw $BANDWIDTH -dly $DELAY router1:pc1@$SCENARIO_ID > /dev/
null
24 sudo vlink -bw $BANDWIDTH -dly $DELAY router1:pc3@$SCENARIO_ID > /dev/
null
25 sudo vlink -bw $BANDWIDTH -dly $DELAY router2:pc2@$SCENARIO_ID > /dev/
null
26 sudo vlink -bw $BANDWIDTH -dly $DELAY router2:pc4@$SCENARIO_ID > /dev/
null
27 sudo vlink -bw $BANDWIDTH -dly $DELAY router2:router1@$SCENARIO_ID > /
dev/null
28 sleep 2
29
30 # Check status (optional)
31 sudo vlink -s router1:pc1@$SCENARIO_ID
32 sudo vlink -s router1:pc3@$SCENARIO_ID
33 sudo vlink -s router2:pc2@$SCENARIO_ID
34 sudo vlink -s router2:pc4@$SCENARIO_ID
35 sudo vlink -s router2:router1@$SCENARIO_ID
36 sleep 2
37
38 # Start iperf servers
39 sudo himage pc2@$SCENARIO_ID iperf -s &> /dev/null &
40 sudo himage pc4@$SCENARIO_ID iperf -s &> /dev/null &
41
42 # Generate background UDP traffic (mute output)
43 sudo himage pc1@$SCENARIO_ID iperf -c 10.0.3.20 -u -t 100000 -b 10M &> /
dev/null &
44 sleep 2
45
46
47 # Run TCP test (this is the one you want to see)
48 echo "=====
49 echo "Running TCP test between PC3 and PC4..."
50 sudo himage pc3@$SCENARIO_ID iperf -c 10.0.4.20 -n 100M -P $FLUXES -i 1
51
52 # Stop simulation
53 echo "=====
54 echo "Stopping IMUNES simulation with scenario ID: $SCENARIO_ID"
55 sudo imunes -b -e $SCENARIO_ID > /dev/null

```

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

Para executar-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)/
12     output.log
13     @for i in $(seq 1000 1008); do \
14         echo ">> Execução $$i da ETAPA1" | tee -a $(LOGDIR)/output.log; \
15         DELAY=10000 FLUXES=1 SCENARIO_ID=$$i $(SCRIPT) 2>&1 | tee -a
16         $(LOGDIR)/output.log; \
17         done
18
19 exec2:
20     @echo "ETAPA2: Executando com DELAY=100000 FLUXES=1" | tee -a
21     $(LOGDIR)/output.log
22     @for i in $(seq 1000 1008); do \
23         echo ">> Execução $$i da ETAPA2" | tee -a $(LOGDIR)/output.log; \
24         DELAY=100000 FLUXES=1 SCENARIO_ID=$$i $(SCRIPT) 2>&1 | tee -a
25         $(LOGDIR)/output.log; \
26         done
27
28 exec3:
29     @echo "ETAPA3: Executando com DELAY=10000 FLUXES=4" | tee -a $(LOGDIR)/
30     output.log
31     @for i in $(seq 1000 1008); do \
32         echo ">> Execução $$i da ETAPA3" | tee -a $(LOGDIR)/output.log; \
33         DELAY=10000 FLUXES=4 SCENARIO_ID=$$i $(SCRIPT) 2>&1 | tee -a
34         $(LOGDIR)/output.log; \
35         done
```

```
31 exec4:
32   @echo "ETAPA4: Executando com DELAY=100000 FLUXES=4" | tee -a
33   $(LOGDIR)/output.log
34   @for i in $(seq 1000 1008); do \
35     echo ">> Execução $$i da ETAPA4" | tee -a $(LOGDIR)/output.log; \
36     DELAY=100000 FLUXES=4 SCENARIO_ID=$$i $(SCRIPT) 2>&1 | tee -a
37     $(LOGDIR)/output.log; \
38   done
39 .PHONY: all exec1 exec2 exec3 exec4
```

2.3. Resultados obtidos

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

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

3. Conclusão

4. Referências