Trabalho Prático: Avaliação de QoS com RTP no Mininet

Este documento descreve os passos para executar os experimentos, coletar dados e gerar análises gráficas sobre perda e jitter usando pacotes RTP.

1. PRÉ-REQUISITOS:

- Mininet instalado (recomendado via WSL ou VM)
- ffmpeg, ffplay e iperf
- tshark (Wireshark CLI)
- Python 3 com matplotlib e pandas
- Vídeo "video.mp4" na mesma pasta dos scripts

2. EXECUÇÃO DO EXPERIMENTO SEM QoS:

Execute:
    sudo python3 experimento.py

3. EXECUÇÃO DO EXPERIMENTO COM QoS:

Execute:
    sudo python3 experimento_qos.py

4. CAPTURA DOS PACOTES RTP:

Execute o comando tcpdump antes do tráfego RTP, no script ou manualmente no terminal:
    h2.cmd('tcpdump -i h2-eth0 udp port 5004 -w /tmp/video_rtp.pcap &')

Após o experimento, copie o arquivo pcap para sua máquina:
    sudo cp /tmp/video_rtp.pcap ~/video_rtp.pcap

5. EXTRAÇÃO DE DADOS COM TSHARK:

Execute no terminal:
     tshark -r ~/video_rtp.pcap -Y "rtp" -T fields -e frame.time_relative -e rtp.seq -E header=y -E separator=, > ~/rtp.csv

6. ANÁLISE E GERAÇÃO DE GRÁFICOS:

Crie o script Python "calc_rtp_stats.py" com o seguinte conteúdo:

---------- INÍCIO DO SCRIPT ----------
```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("/home/$USER/rtp.csv")
df = df.dropna(subset=["rtp.seq", "frame.time_relative"])
df["rtp.seq"] = df["rtp.seq"].astype(int)
df["frame.time_relative"] = df["frame.time_relative"].astype(float)

df = df.sort_values("rtp.seq")
seqs = df["rtp.seq"].to_list()
times = df["frame.time_relative"].to_list()
```

```
jitters = [abs(times[i] - times[i-1]) for i in range(1, len(times))]
avg_jitter = sum(jitters) / len(jitters) if jitters else 0

expected = seqs[-1] - seqs[0] + 1
received = len(seqs)
lost = expected - received
loss_percent = (lost / expected) * 100 if expected else 0

print(f"[4/5] Jitter médio estimado: {avg_jitter:.6f} segundos")
print(f"[5/5]  Pacotes  RTP  esperados:  {expected},  recebidos:  {received},  perdidos:
{lost}")
print(f"      Porcentagem de perda: {loss_percent:.2f}%")

plt.figure(figsize=(10, 4))
plt.plot(jitters, label="Jitter (s)", color="blue")
plt.title("Jitter entre pacotes RTP")
plt.xlabel("Pacote")
plt.ylabel("Jitter (segundos)")
plt.grid(True)
plt.tight_layout()
plt.savefig("/home/$USER/jitter_plot.png")
plt.close()

plt.figure(figsize=(10, 4))
plt.plot(seqs, label="Seq", color="green")
plt.title("Sequência RTP Recebida")
plt.xlabel("Ordem de chegada")
plt.ylabel("Número de sequência RTP")
plt.grid(True)
plt.tight_layout()
plt.savefig("/home/$USER/seq_rtp_plot.png")
plt.close()
------------------------------ FIM DO SCRIPT ------------------------------
```

7. EXECUTE O SCRIPT:

```
--------------------
    python3 calc_rtp_stats.py
```

Ele mostrará na tela:
- Jitter médio
- Pacotes esperados, recebidos e perdidos
- Porcentagem de perda

E salvará dois gráficos em PNG:
- jitter_plot.png
- seq_rtp_plot.png

8. INTERPRETAÇÃO DOS RESULTADOS:

```
-------------------------------
```

- Gráfico de jitter alto indica variação no tempo de entrega (problemas de atraso).
- Gráfico de sequência mostra buracos se houver perdas.

Compare os gráficos com e sem QoS para avaliar a eficácia da priorização de tráfego.