QSI-Ferramenta para monitorização de dispositivos móveis
Grupo 1

Bruno Silva[1,2] a71385,
João Bernardo Freitas[1,3] a74814, and
Eduardo Gil Rocha[1,4] a77048

[1] Universidade do Minho, Braga, Portugal
[2] a71385@alunos.uminho.pt
[3] a74814@alunos.uminho.pt
[4] a77048@alunos.uminho.pt

1 Definição do problema

O grupo, para este trabalho prático, decidiu que iria tentar desenvolver uma ferramenta que permita tirar medições activas numa rede, nomeadamente, latência, jitter e perda de pacotes.
O grupo decidiu tentar implementar medições activas entre um cliente e um probe, algures numa rede, visto que, ao contrário das medições passivas, é injectado tráfego na rede de forma a simular uso real o que nos dá maior controlo sobre quando e como é que as medições são tiradas.

2 Introdução

Nos últimos anos temos visto que uma grande parte da largura de banda utilizada está a ser ocupada por serviços de streaming, como *Youtube, NetFlix* e mais recentemente serviços para o *ETrabalho*. Estes serviços usam o protocolo **UDP** na camada de transporte, visto que oferece velocidades mais altas que **TCP** mas não tem controlo de congestionamento nem garante a entrega dos pacotes.
Tendo em conta a natureza desses serviços e a natureza do protocolo utilizado, verificamos que são serviços extremamente sensíveis ao estado da ligação, como tal é necessário a existência de ferramentas que tirem medidas **QOS** de forma a detectar e resolver eventuais problemas nessas ligações.

3 O que estão a tentar medir

Estas medições irão ser tiradas através de um par *cliente->probe*, onde o cliente irá enviar pacotes para o probe, que irá posteriormente devolver. Ao receber de novo os pacotes, o cliente irá poder verificar quanto tempo é que um pacote individual demorou a chegar á probe e quanto tempo demorou para fazer a viagem de ida-e-volta. Ao enviar diversos pacotes irá ser possível verificar se há perda de pacotes bem como tirar medidas relacionadas com os tempos de transmissão.

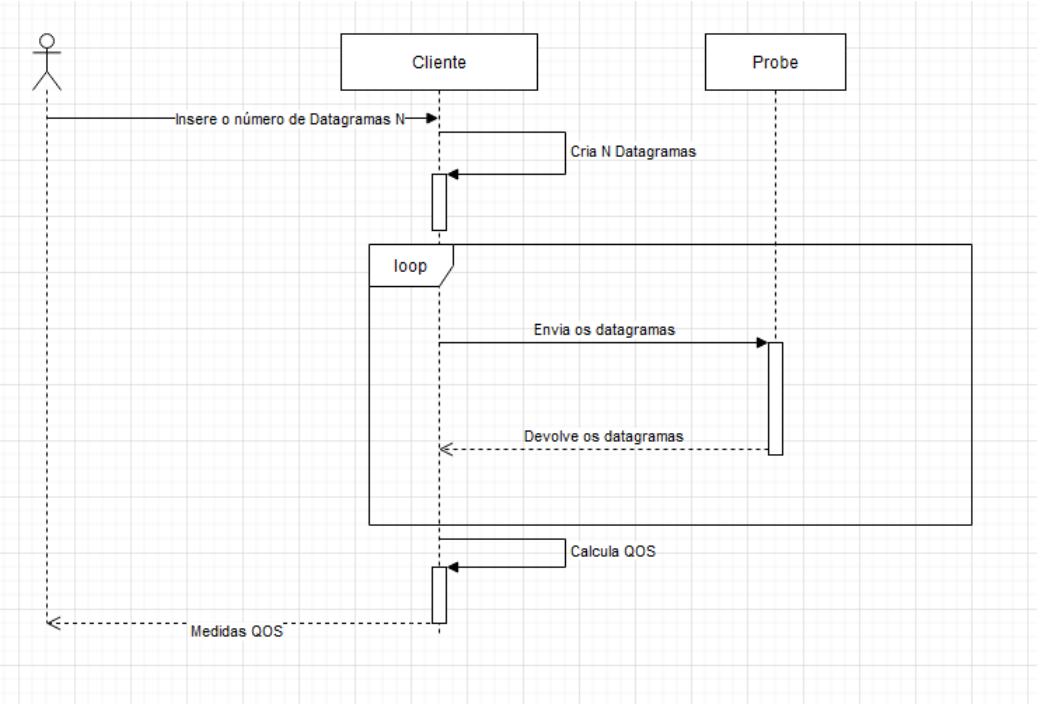

Fig. 1. Diagrama UML do funcionamento da ferramenta

4 Como estão a tentar medir

O grupo irá tirar estas medidas através de uma ferramenta desenvolvida por nós na linguagem **JAVA**, sendo as bibliotecas utilizadas para podermos enviar pacotes entre duas máquinas foram, **DatagramSocket**, **DatagramPacket** e **InetAddress**.
Essa ferramenta irá ser implementada num dispositivo móvel e irá capturar apenas o seu próprio tráfego.
Os pacotes em si são pacotes **UDP** que irão conter um objeto criado por nós ao qual chamamos de **Datagrama**. Este objeto irá conter 4 campos:

- **int id**- Identifica um **Datagrama**
- **long saida**- Timestamp de saida do Cliente
- **long intermedio**- Timestamp de chegada à Probe
- **long chegada**- Timestamp de chegada ao Cliente

Para o Cliente poder enviar e receber pacotes ao mesmo tempo decidimos usar duas **Threads**, uma que envia os pacotes para a **Probe** e outra que recebe as respostas.
Ao comparar o tamanho da lista de **Datagramas** que enviou com a lista dos **Datagramas** que recebeu é possível calcular a perda de pacotes.
Para calcular a latência de um **Datagrama** comparamos os campos **saida** e **chegada**, obtendo então a latência para esse mesmo.
A latência média é obtida através da média das latências de todos os **Datagramas**.
Por fim calculamos o *jitter*, que é obtido ao calcular a média das diferenças entre as latências.

5 Métricas. Porquê estas

Estas métricas foram escolhidas porque eram relevantes para a situação actual, onde várias empresas e universidades foram obrigadas a adoptar soluções de ETrabalho através da internet, soluções estas que funcionam principalmente com base no protocolo **UDP**.

6 Implementação

A topologia que o grupo está a considerar usar para este trabalho é uma rede onde há três ou mais elementos, sendo que um dos elementos é o cliente, um é o probe e os restantes são routers normais.
O cliente e o probe tanto podem estar na mesma rede como podem estar distantes um do outro.
Esta topologia irá ser criada no *CORE*, onde é possível alterar os parâmetros das ligações de forma a simular delays e perdas de pacotes, entre outros.

7 Limitações identificadas

Ao realizar testes o grupo identificou algumas limitações da ferramenta, nomeadamente, para o funcionamento correto desta mesma, é necessário que o cliente saiba, de alguma forma, endereço da Probe para poder enviar os pacotes. Para além disso, a presente implementação da Probe só aceita um cliente de cada vez e se a Probe não recebe primeiro pacote a ser enviado, que indica o número de pacotes que o cliente quer enviar, o programa não irá funcionar correctamente e obrigará a o teste a ser recomeçado.

8 Resultados

Para testar a nossa solução decidimos utilizar o **Core**, visto que este permite-nos criar topologias bem como mudar o estado de uma ligação.
Em todos os testes foram enviados 100 pacotes com o tamanho fixo de 1024 bytes do nó n1 para o nó n3.

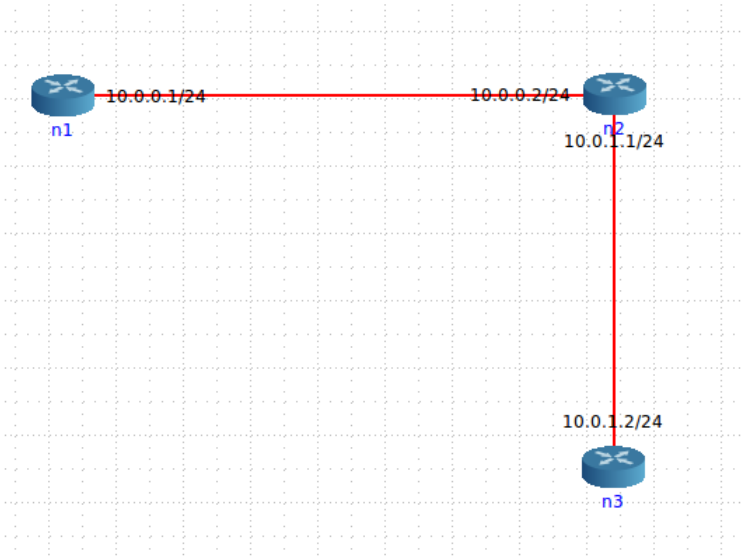

Fig. 2. Topologia controlo

8.1 Controlo

Para este primeiro teste irá servir como **Controlo** para os outros, como tal temos ligações sem delay e sem perdas, logo esperamos que todos os pacotes sejam enviados e recebidos e que as latências baixas.

```
root@n3:/home/core/Desktop/TP3/src# java app/Probe 10.0.1.2
À espera de 100 pacotes
root@n3:/home/core/Desktop/TP3/src#
```

```
root@n1:/home/core/Desktop/TP3/src# java app/Client 10.0.1.2
Insira o número de pacotes a enviar
100


Connected!
Resposta da Probe
Mensagens enviadas
Latência média->1ms
Jitter-> 0.4489796ms
Pacotes perdidos->0
```

Fig. 3. Resultados controlo

8.2 Latência

Para testar a latência alteramos a topologia para que a ligação **n2->n3** tenha um delay de 200ms.

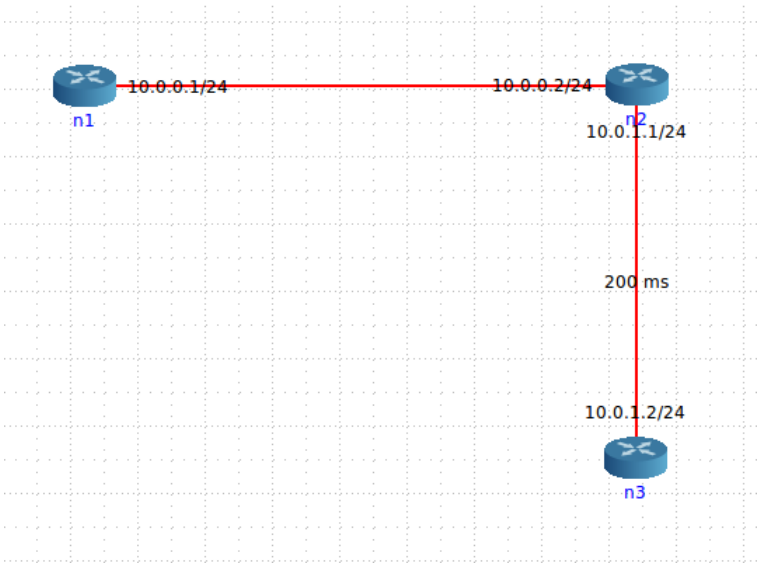

Fig. 4. Topologia delay

Como podemos ver nos resultados, a latência média passou de 1ms para 400ms, 200ms de **n2->n3** e 200ms de **n3->n2**.
Mais uma vez vemos que não houve perda de pacotes.

Fig. 5. Resultados delay

8.3 Perda

Por fim adicionamos perda de pacotes á mesma ligação.

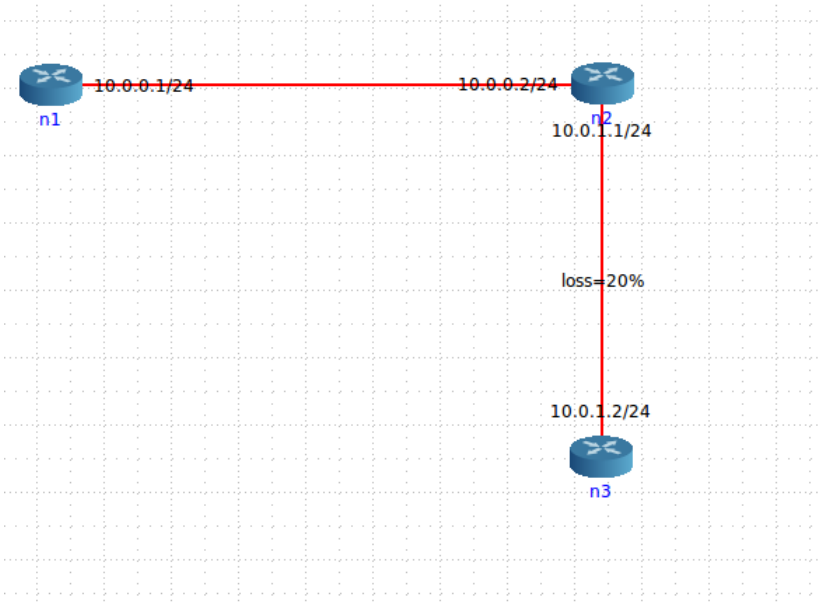

Fig. 6. Topologia perda

Fig. 7. Resultados perda

Como podemos ver a perda detectada **39%** é maior que a definida **20%**. Isto deve-se, mais uma vez, ao facto da perda ser aplicada aos 100 pacotes enviados do cliente para a probe, sendo depois aplicada aos cerca de 80 **Datagramas** que são enviados de volta.

9 Conclusão

Após conclusão deste projeto temos um melhor entendimento de como funcionam e são implementadas este tipo de ferramentas de **QOS**.
Visto que conseguimos implementar um protótipo funcional de uma ferramenta de **QOS**, que permitirá a um utilizador tirar diversas medidas relevantes sobre a sua ligação, consideramos este projeto como um sucesso.