



Universidade do Minho

Departamento de Informática

UC: Engenharia de Serviços em Rede

Serviço Over-the-top para entrega de multimédia

Alunos:

Eduardo Cunha – PG55939

Guilherme Gonçalves – PG57546

José Pacheco – PG55972

PL7-Grupo N.º 1

Braga, 2024



1. INTRODUÇÃO

Este relatório aborda o desenvolvimento de um serviço Over the Top (OTT) voltado para a entrega de conteúdos multimédia, no contexto da unidade curricular de Engenharia de Serviços em Rede. O objetivo do projeto foi conceber e desenvolver um serviço de streaming eficiente e otimizado, com a aplicação de uma rede de overlay aplicacional.

As atividades realizadas incluíram a criação da topologia overlay, a implementação do serviço de streaming, a monitorização dos servidores de conteúdos, a configuração dos fluxos de transmissão de dados, bem como o desenvolvimento dos Nodes da Overlay, dos Clientes e de um bootstrapper.

O relatório inicia com a descrição da Arquitetura da Solução, seguida pela especificação dos protocolos utilizados. Em seguida, é detalhada a Implementação. O documento continua com a apresentação de alguns testes realizados e seus respectivos resultados. Para concluir, é feita uma análise global do trabalho na seção de Conclusão, além de serem mencionadas sugestões para o trabalho futuro, destacando as melhorias necessárias caso houvesse continuidade na implementação.

2. ARQUITETURA DA SOLUÇÃO

A solução foi desenvolvida com uma arquitetura modular, composta pelos principais componentes: **Cliente**, **OverlayNode**, **Servidor** e **Bootstrapper**, apoiados por várias classes auxiliares, como **VideoStream**, **ServerWorker**, **RtpPacket**, **ClientStream** e **message.py**. Cada componente tem responsabilidades bem definidas, que serão apresentadas em detalhe nas secções seguintes. Nesta secção, é feita uma visão mais abrangente sobre a função de cada elemento principal, enquanto a secção de Implementação irá detalhar as formas específicas como cada funcionalidade foi desenvolvida.

2.1 BOOTSTRAPPER

Após a ativação, o Bootstrapper torna-se conhecedor de toda a topologia através de um ficheiro recebido. À medida que os elementos da topologia são ativados, estes enviam pedidos de registo ao Bootstrapper. Este marca os elementos como ativos e responde com a lista de vizinhos que já estão ativos e pertencem à configuração do elemento.

2.2 SERVIDOR

O servidor, após estabelecer contacto com o bootstrapper e identificar os seus vizinhos ativos, tem como principais responsabilidades o envio periódico de mensagens de *flood*, a gestão do controlo dos vizinhos e a iniciação do streaming.

O envio de mensagens de *flood* é essencial para atualizar as tabelas de encaminhamento de todos os elementos no *overlay*. Esta atualização é crucial para permitir a ativação de uma rota funcional para o streaming.

2.3 OVERLAYNODE

Após estabelecer contacto com o *bootstrapper* e identificar os seus vizinhos ativos, o OverlayNode assume várias responsabilidades principais: gerir as ligações com os vizinhos ativos, propagar mensagens de *flood*, manter uma tabela de encaminhamento (*routing*) atualizada e processar os pedidos, encaminhando-os para o servidor. No caso de atuar como um ponto de presença (*POP*), deve ainda assegurar o contacto direto com o cliente.

2.4 CLIENTE

O Cliente regista-se no Bootstrapper, gere os vizinhos e seleciona a melhor rota com base nas métricas recebidas. A rota escolhida entre as disponíveis é ativada, iniciando-se, a partir desse momento, a interface gráfica completa. Nesta interface, o utilizador pode realizar pedidos ao POP, que são replicados até ao Servidor, permitindo operações como solicitação de vídeo, início, pausa e término do stream.

3. ESPECIFICAÇÃO DOS PROTOCOLOS

A escolha do protocolo aplicacional para o streaming recaiu sobre o **RTP (Real-time Transport Protocol)**, utilizando uma implementação fornecida pelos docentes, que foi adaptada às necessidades do projeto. O RTP opera sobre o protocolo de transporte **UDP**, conhecido pela sua simplicidade e baixa latência, características fundamentais para transmissões em tempo real. Adicionalmente, o RTP é ideal para cenários onde existe tolerância a perdas de pacotes, atendendo de forma eficaz às exigências do nosso projeto.

No entanto, para operações de controlo, optámos por implementar um protocolo aplicacional baseado no **TCP**, garantindo a troca segura e fiável de mensagens. O **TCP** é utilizado nas seguintes operações de controlo:

- **Registo no Bootstrapper:** O cliente comunica em TCP para enviar o pedido de ativação e receber os dados dos vizinhos.
- **Pings de controlo:** A troca de mensagens de monitorização entre os nós da rede overlay ocorre em TCP.
- **Propagação de flood:** A disseminação de mensagens de controlo na rede overlay é realizada em TCP.
- **Pedidos de ativação de rota:** O envio de solicitações para ativação de rotas também utiliza TCP.

No que diz respeito aos pedidos de streaming, a comunicação é híbrida:

- O pedido inicial do cliente ao POP é transmitido em **TCP**, enquanto a replicação do pedido até ao servidor utiliza **UDP**, para alinhar com as características do RTP.

Reconhecemos, no entanto, uma falha em relação ao enunciado do projeto, que estipulava que toda a comunicação direta com o cliente deveria ser realizada utilizando **UDP**. Este

erro de implementação resultou, exclusivamente, de uma falha de atenção da nossa parte. Quando identificámos a questão, já tínhamos avançado consideravelmente no desenvolvimento, e a sua correção exigiria alterações significativas em várias estruturas, incluindo o **Bootstrapper**, o **Cliente** e o **OverlayNode**. Dada a extensão das mudanças necessárias, optámos por manter a implementação original, assumindo assim o desvio em relação ao requisito estabelecido.

4. IMPLEMENTAÇÃO

4.1 INICIALIZAÇÃO DOS ELEMENTOS DA TOPOLOGIA

Todos os elementos executáveis da rede overlay recebem como parâmetro um ficheiro JSON de entrada, que especifica os seus próprios IPs, tipos na topologia, portas necessárias e, ainda, a forma de contactar o bootstrapper.

```
{} bootstrapper.json
{} cenariofinal.json
{} cenariotrocarota.json
{} client1.json
{} client2.json
{} client3.json
{} node1.json
{} node2.json
{} node3.json
{} node5.json
{} node6.json
{} pop1.json
{} pop2.json
{} servidor.json
```

Figura 1-Json files

```
{
  "node_ip": "10.0.18.1",
  "node_id": "node6",
  "node_type": "node",
  "rtsp_port": 3000,
  "rtp_port": 3000,
  "control_port": 5000,
  "bootstrapper_ip": "10.0.28.10",
  "bootstrapper_port": 8000
}
```

Figura 2-Nodo6.json

No caso do bootstrapper, este recebe dois ficheiros: além do ficheiro que define vários dos seus parâmetros, recebe também um ficheiro com a topologia overlay completamente definida. Abaixo, segue um excerto do ficheiro JSON que define a topologia:

```
{
  "nodes": [
    {
      "node_id": "server",
      "node_ip": "10.0.18.10",
      "neighbors": [
        {"neighbor_id": "node5", "neighbor_ip": "10.0.19.1"},
        {"neighbor_id": "node6", "neighbor_ip": "10.0.18.1"}
      ]
    },
    {
      "node_id": "node5",
      "node_ip": "10.0.19.1",
      "neighbors": [
        {"neighbor_id": "server", "neighbor_ip": "10.0.18.10"},
        {"neighbor_id": "node1", "neighbor_ip": "10.0.17.1"},
        {"neighbor_id": "node6", "neighbor_ip": "10.0.18.1"}
      ]
    }
  ],
}
```

Figura 3-Bootstrapper.json

4.2 ATIVAÇÃO AO BOOTSTRAPPER

Todos os elementos da topologia, ao serem inicializados, notificam imediatamente o bootstrapper sobre a sua ativação. Este, por sua vez, regista-os como ativos no seu dicionário de elementos da topologia e, em seguida, envia os vizinhos ativos que o elemento recém-inicializado possui.

```
A atualizar vizinhos..
Node node1 | Vizinhos: {'10.0.19.1': {'node_id': 'node5', 'control_port': 5000,
'node_type': '', 'status': 'ativa', 'ping_falhados': 0}, '10.0.13.2': {'node_id'
: 'pop1', 'node_type': 'pop', 'control_port': 5000, 'ping_falhados': 0, 'status'
: 'ativa'}}
```

Figura 4-Atualização dos vizinhos

4.3 CONTROLO DE VIZINHOS

Todos os elementos da topologia (exceto o bootstrapper) enviam pings a cada 20 segundos para todos os seus vizinhos, com o objetivo de monitorizar o bom funcionamento da rede. Caso uma falha no envio do ping ocorra, a tentativa é contabilizada como falhada. Se houver três tentativas consecutivas falhadas para o mesmo vizinho, esse vizinho é marcado como inativo no dicionário do próprio nó. Por outro lado, se o vizinho responder com um **PING_RESPONSE**, ele é marcado como ativo e o contador de pings falhados é reiniciado.

```
PING enviado para: node5
PING_RESPONSE de: node5
Ping falhado para o node: node5: [Errno 111] Connection refused
Ping falhado para o node: node5: [Errno 111] Connection refused
Ping falhado para o node: node5: [Errno 111] Connection refused
Vizinho node5 tornado inativo
```

Figura 5-Vizinho Desativado

4.4 FLOOD

O servidor envia uma mensagem de **flood** a cada 30 segundos. Esta mensagem é encaminhada para todos os vizinhos ativos, propagando-se por toda a topologia. O objetivo é atualizar as tabelas de rotas em cada nó, definindo todas as árvores de distribuição possíveis na topologia. É importante destacar que o envio de uma mensagem de flood não ativa automaticamente nenhuma rota para a transmissão de vídeo.

Segue-se um exemplo de uma tabela de rotas atualizada após a receção de uma mensagem de flood:

```
MensagemFlood recebida
Flood recebido vindo do: 10.0.19.1

=====
Source IP      | Stream ID | Source ID | Ctrl Port | RTSP Port | Status
| Métrica
=====
10.0.19.1      | filme.Mjpeg | node5     | 5000      | 3000      | inativa
| 1
10.0.19.1      | movie.Mjpeg | node5     | 5000      | 3000      | inativa
| 1
=====

Mensagem de Flood reencaminhada para: pop1 | Numero de Saltos: 2
```

Figura 6- Atualização das tabelas de rotas

Como se pode observar, o nó recebeu uma mensagem de flood e atualizou a sua tabela de rotas com base no endereço IP do remetente. Duas rotas possíveis foram criadas para a propagação de conteúdos de dois filmes diferentes. Além disso, foram registadas as portas de controlo e RTSP do nó de onde recebeu a mensagem. O estado das rotas, como mencionado anteriormente, permanece **inativo** até que seja recebido um pedido de ativação de rota.

Por fim, é possível verificar que a mensagem de flood foi reencaminhada para o nó **pop1**, que, no momento da captura, era o único vizinho ativo disponível.

4.5 ATIVAÇÃO DE ROTA

Desde a sua inicialização, o cliente realiza, a cada 20 segundos, uma procura por uma rota, com o objetivo de identificar e ativar a melhor rota disponível para o filme solicitado. Assim que uma rota válida é encontrada, é enviado um pedido de ativação ao **POP**, que é posteriormente replicado até ao servidor. Este processo torna a rota ativa e pronta para a propagação do serviço de streaming.

Caso um cliente solicite um filme para o qual já exista uma rota ativada, ele será rapidamente alocado a essa rota. No primeiro nó onde a rota já estiver ativa, será exibida a mensagem "**JÁ EXISTE ROTA!!!!**". Vale destacar que, se o cliente solicitar um filme que já está em transmissão, ele começará a visualização exatamente no ponto em que a transmissão se encontra naquele momento.

Exemplo de pedido de ativação de rota para um filme já existente:



Figura 7-Pedido de ativação de rota já existente

Como pode ser visto na imagem, o cliente 2 já estava a visualizar o filme. Quando o cliente 3 solicita o mesmo filme, o **POP1** exibe a mensagem **"JÁ EXISTE ROTA!!!!"** e imediatamente aloca o cliente 3 à mesma rota. A partir desse momento, o cliente 3 pode iniciar a visualização do filme exatamente no ponto em que o cliente 2 estava a assistir.

4.6 ESCOLHA DO MELHOR POP

Caso o cliente tenha a possibilidade de escolher entre dois PoPs para se conectar e receber o vídeo, ele avaliará as métricas de cada rota disponível. A escolha do melhor Point of Presence (PoP) assegura que o cliente se conecte ao PoP mais adequado para uma entrega eficiente do fluxo de vídeo. A métrica utilizada para essa escolha é o **número de saltos**, que, como sabemos de antemão, não é a métrica ideal. A métrica de saltos é calculada através das mensagens de flood, nas quais, a cada replicação, é acrescentado um valor ao contador **"metrica"**, que representa unicamente o número de saltos.

Considerou-se a adição de métricas relacionadas à latência, e essa melhoria será abordada na seção de trabalho futuro, no final do relatório.

4.7 TROCA DE ROTA

Em relação à troca de rotas, temos implementada uma funcionalidade que permite a atualização da rota caso a atual não esteja ativa. Ou seja, se a rota ainda não tiver sido ativada e surgir uma rota mais eficiente para o filme solicitado, ela será automaticamente atualizada. No entanto, se a rota já tiver sido ativada, a troca por uma nova rota mais eficiente não será realizada. Temos consciência dessa limitação e discutimos uma possível solução para este problema na seção de **trabalho futuro**.

4.8 PEDIDO DE STREAMING

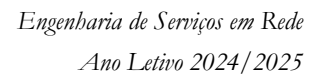
Os pedidos de streaming são realizados pelo cliente após a inicialização da interface gráfica. Esses pedidos são enviados diretamente ao **POP**, que, por sua vez, os encaminha através dos seus vizinhos ativos até ao servidor. Antes de iniciar a reprodução, é necessário configurar o vídeo (**setup**) e aguardar que o cliente exiba a mensagem "**Bind to RTP port**". Após este passo, pode-se enviar o pedido de **play** para iniciar a visualização. Durante a reprodução, o cliente pode alternar entre os comandos **pause/play** ou realizar um pedido de **pause/teardown** para encerrar a sessão.

4.9 MENSAGEM.PY

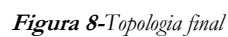
Para facilitar a troca de mensagens entre os elementos da topologia, foi desenvolvida a classe **mensagem.py**, responsável por estruturar todas as mensagens enviadas na rede overlay. Este ficheiro define três classes principais:

- **MensagemControlo**: responsável pela gestão de mensagens de controlo, como os pings.
- **MensagemFlood**: utilizada para a propagação de mensagens de flood na rede.
- **VizinhoInfo**: fornece detalhes sobre os vizinhos na rede.

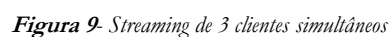
As mensagens de controlo e **flood** desempenham um papel crucial no funcionamento da rede overlay, transportando informações essenciais para a sua operação. Cada mensagem é identificada por um **type**, que define a sua funcionalidade. No caso das mensagens de controlo, os tipos incluem, por exemplo: "**REGISTER**", "**VIZINHOS_UPDATE**" e "**PING**". Já nas mensagens de flood, os tipos são "**FLOODING_UPDATE**" e "**ATIVAR_ROTA**". Além disso, cada mensagem contém atributos como o IP do node remetente, o tipo de node e a porta de controlo associada, assegurando uma comunicação eficiente e organizada entre os elementos da rede.



5.1 TOPOLOGIA FINAL



5.1.1 STREAMING 3 CLIENTES (2 FILMES DISTINTOS) (TOPOLOGIA)



```

vcmd
Ei o filme.Mjpeg em JDP para ADDRESS : 10.0.18.1
Ei o movie.Mjpeg em JDP para ADDRESS : 10.0.18.1
Ei o filme.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 279 do video movie.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 161 do video filme.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 280 do video movie.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 162 do video filme.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 281 do video movie.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 163 do video filme.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 282 do video movie.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 164 do video filme.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 283 do video movie.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 165 do video filme.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 284 do video movie.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 166 do video filme.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 285 do video movie.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 167 do video filme.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 286 do video movie.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 168 do video filme.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 287 do video movie.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 169 do video filme.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 288 do video movie.Mjpeg em JDP para ADDRESS : 10.0.18.1
Enviar pacote 170 do video filme.Mjpeg em JDP para ADDRESS : 10.0.18.1

```

Figura 10-Filmes distintos

5.2 TOPOLOGIA DE TESTES

Definimos a topologia anterior como a topologia final do nosso trabalho. No entanto, com base na métrica que utilizamos (número de saltos), criamos uma outra topologia, apresentada a seguir, apenas para testar a correcta implementação da escolha do melhor POP.

Como é facilmente observável, a única diferença entre esta topologia e a anterior é a remoção da ligação entre o node2 e o pop1, além da desativação dos nodes 6 e 3. Isso permite testar a correcta implementação da escolha da melhor rota. Nesse caso, o caminho para o pop2 envolve um salto adicional em comparação com o caminho para o pop1, fazendo com que o cliente 2 (ligado a ambos os pops) escolha sempre o pop1, pois apresenta um número menor de saltos.

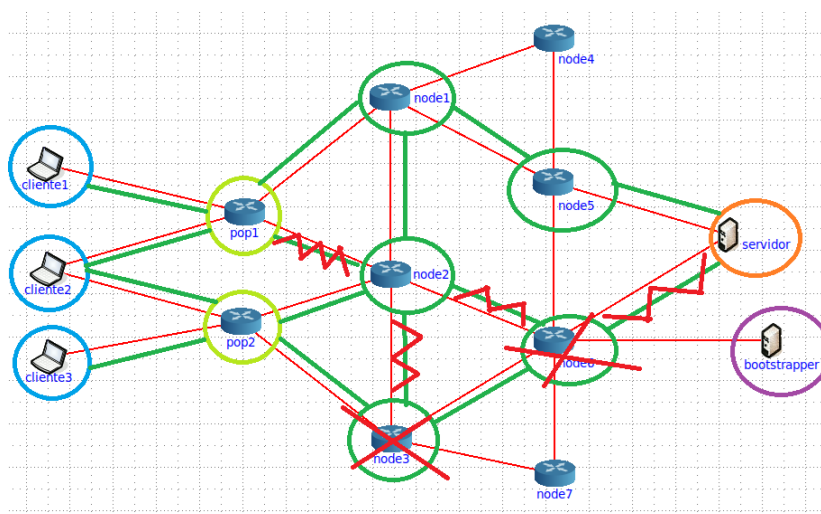
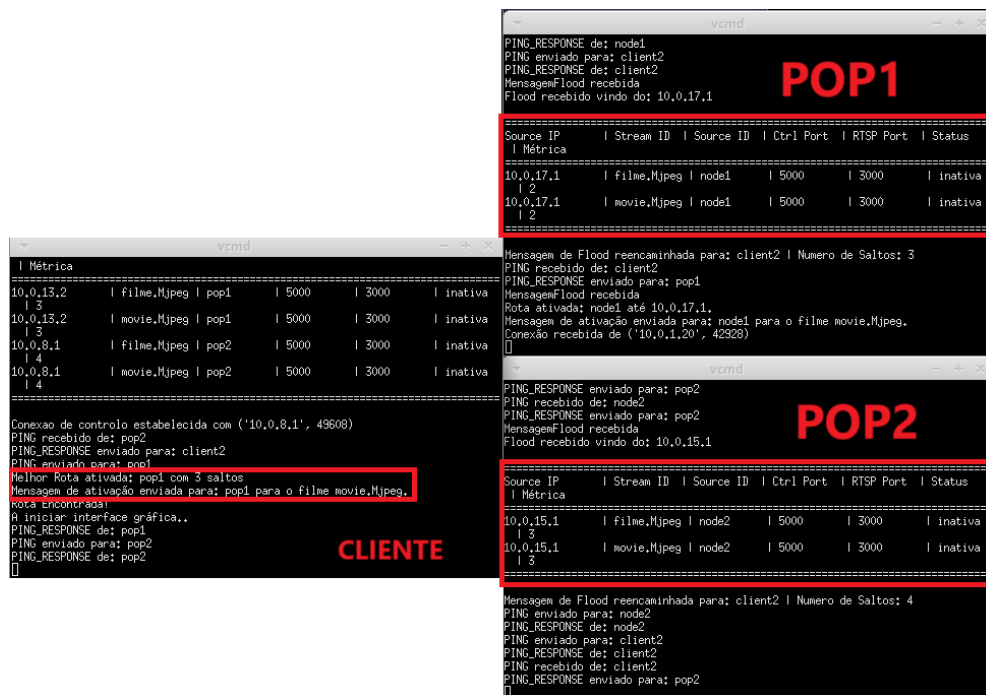


Figura 11-Topologia de testes

5.2.1 SELEÇÃO DO MELHOR POP (TOPOLOGIA TESTE)



```

vcmd
PING_RESPONSE de: node1
PING enviado para: client2
PING_RESPONSE de: client2
MensagemFlood recebida
Flood recebido vindo de: 10.0.17.1

POP1

Source IP | Stream ID | Source ID | Ctrl Port | RTSP Port | Status
| Métrica
=====
10.0.17.1 | filme.Mjpeg | node1 | 5000 | 3000 | inativa
| 2
10.0.17.1 | movie.Mjpeg | node1 | 5000 | 3000 | inativa
| 2

Mensagem de Flood reencaminhada para: client2 | Numero de Saltos: 3
PING recebido de: client2
PING_RESPONSE enviado para: pop1
MensagemFlood recebida
Rota ativada: node1 até 10.0.17.1.
Mensagem de ativação enviada para: node1 para o filme movie.Mjpeg.
Conexao recebida de ('10.0.1.20', 42328)

vcmd
PING_RESPONSE enviado para: pop2
PING recebido de: node2
PING_RESPONSE enviado para: pop2
MensagemFlood recebida
Flood recebido vindo de: 10.0.15.1

POP2

Source IP | Stream ID | Source ID | Ctrl Port | RTSP Port | Status
| Métrica
=====
10.0.15.1 | filme.Mjpeg | node2 | 5000 | 3000 | inativa
| 3
10.0.15.1 | movie.Mjpeg | node2 | 5000 | 3000 | inativa
| 3

Mensagem de Flood reencaminhada para: client2 | Numero de Saltos: 4
PING enviado para: node2
PING_RESPONSE de: node2
PING enviado para: client2
PING_RESPONSE de: client2
PING recebido de: client2
PING_RESPONSE enviado para: pop2

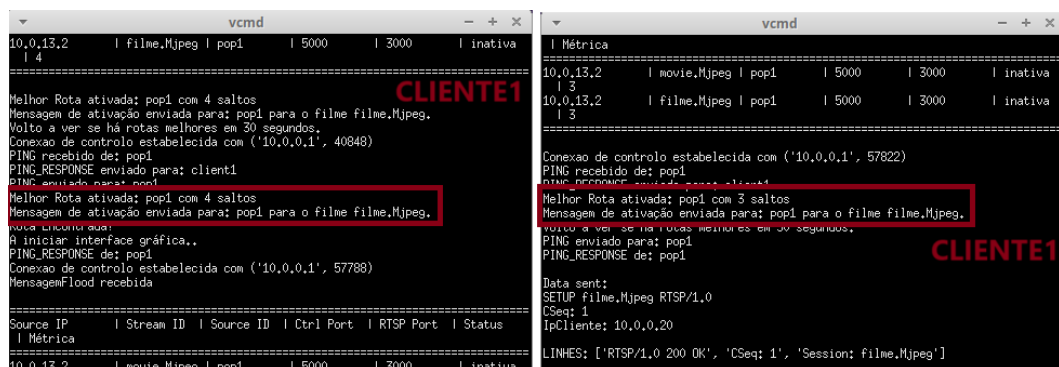
CLIENTE

Conexao de controlo estabelecida com ('10.0.8.1', 43608)
PING recebido de: pop2
PING_RESPONSE enviado para: client2
PING enviado para: pop1
Melhor Rota ativada: pop1 com 3 saltos
Mensagem de ativação enviada para: pop1 para o filme movie.Mjpeg.
Rota encontrada!
A iniciar interface gráfica..
PING_RESPONSE de: pop1
PING enviado para: pop2
PING_RESPONSE de: pop2
  
```

Figura 12- Seleção do melhor pop

5.2.2 TROCA DE ROTA (TOPOLOGIA TESTE)

No caso da troca de rotas, conforme mencionado anteriormente, implementámos uma funcionalidade que permite a atualização da rota caso a rota atual ainda não esteja ativa. Embora a ideia esteja definida e funcione conceptualmente, na prática, a implementação ainda precisa de ser concluída para que opere exatamente como esperado.



```

vcmd
10.0.13.2 | filme.Mjpeg | pop1 | 5000 | 3000 | inativa
| 4

CLIENTE1

Melhor Rota ativada: pop1 com 4 saltos
Mensagem de ativação enviada para: pop1 para o filme filme.Mjpeg.
Volto a ver se há rotas melhores em 30 segundos.
Conexao de controlo estabelecida com ('10.0.0.1', 40848)
PING recebido de: pop1
PING_RESPONSE enviado para: client1
PING enviado para: pop1
Melhor Rota ativada: pop1 com 4 saltos
Mensagem de ativação enviada para: pop1 para o filme filme.Mjpeg.
Rota encontrada!
A iniciar interface gráfica..
PING_RESPONSE de: pop1
Conexao de controlo estabelecida com ('10.0.0.1', 57788)
MensagemFlood recebida

Source IP | Stream ID | Source ID | Ctrl Port | RTSP Port | Status
| Métrica
=====
10.0.13.2 | movie.Mjpeg | pop1 | 5000 | 3000 | inativa

vcmd
| Métrica
=====
10.0.13.2 | movie.Mjpeg | pop1 | 5000 | 3000 | inativa
| 3
10.0.13.2 | filme.Mjpeg | pop1 | 5000 | 3000 | inativa
| 3

Conexao de controlo estabelecida com ('10.0.0.1', 57822)
PING recebido de: pop1
PING_RESPONSE enviado para: client1
Melhor Rota ativada: pop1 com 3 saltos
Mensagem de ativação enviada para: pop1 para o filme filme.Mjpeg.
Volto a ver se há rotas melhores em 30 segundos.
PING enviado para: pop1
PING_RESPONSE de: pop1

Data sent:
SETUP filme.Mjpeg RTSP/1.0
CSeq: 1
IpClient: 10.0.0.20
LINES: ['RTSP/1.0 200 OK', 'CSeq: 1', 'Session: filme.Mjpeg']
  
```

Figura 13- Troca de rota

6. CONCLUSÃO/TRABALHO FUTURO

O sistema desenvolvido demonstrou-se funcional, atendendo aos principais requisitos, como a entrega de vídeo em tempo real, a seleção de rotas baseada no número de saltos, entrega multi-filme sem duplicação de rotas e a integração dos elementos da rede, incluindo clientes, servidores, nós intermediários e o bootstrapper. Adicionalmente, funcionalidades importantes, como a ativação de rotas e a escolha dinâmica do melhor PoP, foram implementadas com sucesso e validadas através de testes.

Conforme referido ao longo do relatório, o sistema apresenta oportunidades de melhoria. Seguem-se os ajustes necessários e uma visão conceptual de como poderiam ser implementados:

1. **Substituir TCP por UDP no contacto com o cliente:** De acordo com o enunciado, a comunicação direta com o cliente deveria ser feita via UDP. Para corrigir isso, seria necessário ajustar estruturas como o bootstrapper e o OverlayNode para que, ao detetar que o vizinho tem “*node_type == client*”, as mensagens sejam enviadas por UDP. O cliente, por sua vez, também precisaria ser modificado para receber mensagens via UDP em vez de TCP.
2. **Monitorização Avançada:** Implementar métricas mais robustas, como a latência, agregando-a ao número de saltos para criar uma métrica composta mais fiável (por exemplo, *latência * saltos*). Essa implementação exigiria medições de tempo, que poderiam ser realizadas durante o envio de mensagens de flood ou através dos pings entre os elementos da topologia.
3. **Troca de Rota (após o início do vídeo):** Para permitir a troca de rota após o início da transmissão, seria necessário introduzir um novo tipo de mensagem, como “*DESATIVAR_ROTA*”, que seria propagada ao longo da rota anteriormente ativa. Essa mensagem sinalizaria que a rota não está mais em uso, permitindo a ativação de uma nova rota mais eficiente.

Posto isso, seria eventualmente necessário abordar as etapas complementares que foram deixadas de lado, a fim de tornar o trabalho totalmente completo.