Engenharia de Serviços em Rede - TP2

Júlio Gonçalves[PG50537], Henrique Alvelos[PG50414], and Duarte Cerquido[PG50469]

Departamento de Informática, Universidade do Minho, Braga, Portugal

Abstract. Este documento sumariza o desenvolvimento de um projeto no âmbito da unidade curricular de Engenharia de Serviços em Rede. Com este, pretendemos desenvolver um serviço para entrega de multimédia em tempo real.

Keywords: OTT, RTP Protocol, Redes de Comunicação

1 Introdução

Ao longo do tempo, é inevitável observar uma mudança de paradigma na partilha e transmissão de informação nas redes. Essa mudança é devida aos serviços Over The Top (OTT), que resolvem a entrega massiva de conteúdos e oferecem serviços específicos desenhados sobre a camada aplicacional.

Como objetivo neste trabalho prático, é nos proposto o desenvolvimento de um protocolo para a entrega de áudio, vídeo e texto em tempo real através de um serviço Over the Top(OTT), promovendo a eficiência e otimização de recursos para uma melhor qualidade da experiência por parte do utilizador.

2 Diretrizes do Projeto

Em seguida, serão expostas algumas diretrizes presentes no enunciado prático, sobre quais o desenvolvimento do projeto deve seguir:

- A partir de um servidor de conteúdos deve ser possível enviar áudio/texto/vídeo para um conjunto de N clientes;
- Um conjunto de nós pode ser usado no reenvio dos dados, como intermediários, formando entre si a rede de overlay aplicacional;
- Quando um nó da rede overlay começa a sua execução, precisa de conhecer pelo menos um outro nó qualquer da rede a quem se ligar;

Os seguintes etapas deverão ser abordados durante o desenvolvimento do projeto de modo a assegurar o correto funcionamento do serviço. É sobre a natureza destas etapas que o nosso protocolo será desenvolvido:

- **Construção da Topologia Overlay**;
- **Serviço de Streaming**;
- **Monitorização da Rede Overlay**;
- **Construção de Rotas para a Entrega de Dados**;
- **Ativação e Teste do Servidor Alternativo**;

3 Arquitetura da Solução

A linguagem Java foi a escolhida para o desenvolvimento deste serviço devido ás inúmeras bibliotecas que esta possui no que toca a transmissão de dados;

O nosso protocolo será desenvolvido tendo em conta o protocolo de transporte UDP, uma vez que oferece uma melhor velocidade de transmissão.

Os intervenientes do nosso serviço podem ser distinguidos entre **Servidor** (nodo que efetua a transmissão de conteúdos), **Cliente** (nodo que faz o pedido de transmissão de conteúdos), **Nodo** (nodo integrante da rede, cuja função é encaminhar mensagens).

4 Especificação do Protocolo

Para existir uma correta comunicação entre os vários nodos, de forma a garantir a integridade dos vários serviços diretos e indiretos que o nosso projeto suporta, foi necessária a criação e diferenciação de diversos tipos de pacotes de dados.

4.1 List Neighboors Request Packet

Este pacote tratado do pedido da lista de vizinhos aquando da inicialização de um nodo.

- **OPCODE** - Identificador do Tipo de Pacote;
- **idOverlay** - Endereço do servidor bootstrapper;

List Neighboors Request Packet	
OPCODE	1
idOverlay	String

4.2 Download Request Packet

Este pacote tratado do pedido de transmissão de conteúdos.

- **OPCODE** - Identificador do Tipo de Pacote;
- **Filename** - Identificador do ficheiro a pedir ao Servidor;

Download Request Packet	
OPCODE	2
Filename	String

4.3 Data Packet

Este pacote permite fazer a transmissão de dados.

- **OPCODE** - Identificador do Tipo de Pacote;
- **BlockN** - Númerador do Pacote;
- **Length**- Tamanho do Pacote;
- **Data** - Informação a ser transmitida;

Data Packet	
OPCODE	3
BlockN	int
Length	short
Data	byte[]

4.4 Acknowledgment Packet

Este pacote permite fazer a confirmação da receção de pacotes.

- **OPCODE** - Identificador do Tipo Pacote;
- **BlockN** - Número do Pacote confirmado;

Acknowledgment Packet	
OPCODE	4
BlockN	int

4.5 Probe Packet

Este pacote permite fazer a monitorização periodicamente das condições da rede.

- **OPCODE** - Identificador do Tipo de Pacote;
- **idServidorRemetente** - Endereço do servidor que difundiu a mensagem;
- **numHops** - Número de nodos percorridos até à receção do pacote;
- **timstampEnvio** - Tempo em que o pacote foi enviado;
- **stringRotaPercorrida** - Endereços dos nodos percorridos até à receção do pacote;
- **atrasoMs** - Diferença de tempos entre o envio e a receção do pacote;

Probe Packet	
OPCODE	5
idServidorRemetente	String
numHops	int
timestampEnvio	Timestamp
stringRotaPercorrida	String
atrasoMs	long

4.6 RTP Packet

Este pacote permite a comunicação em tempo real, seguindo o protocolo Real-time Transport Protocol.

- **Version** - indica a versão do protocolo RTP que está sendo usado.
- **Padding** - indica se o preenchimento foi adicionado ao pacote RTP.
- **Extension** - indica se o pacote RTP contém um cabeçalho de extensão.
- **CC** - indica o número de identificadores CSRC incluídos no pacote RTP.
- **Marker** - específica variantes do protocolo.
- **Payload Type** - indica o tipo de carga transportada no pacote RTP.
- **Sequence Number** - identifica pacotes RTP individuais e permite detectar pacotes perdidos ou fora de ordem.
- **Timestamp** - permite sincronizar o fluxo de mídia e calcular o tempo decorrido entre a transmissão de dois pacotes RTP.
- **Ssrc** - permite identificar a origem de um pacote RTP.
- **Header** - refere os campos no cabeçalho RTP que contêm informações sobre o pacote RTP.
- **Payload_size** - tamanho do cabeçalho
- **Payload** - dados transportados em um pacote RTP, excluindo o cabeçalho RTP.

RTP Packet	
Version	int
Padding	int
Extension	int
CC	int
Marker	int
Payload Type	int
Sequence Number	int
Timestamp	int
Ssrc	int
Header	byte[]
Payload_size	int
Payload	byte[]

4.7 StreamRequestPacket Packet

Este pacote permite informar .

– **OPCODE** - Identificador do Tipo Pacote;

StreamRequestPacket Packet	
OPCODE	6

4.8 EndStreamPacket Packet

Este pacote permite informar .

– **OPCODE** - Identificador do Tipo Pacote;

EndStreamPacket Packet	
OPCODE	9

5 Implementação

A implementação do nosso serviço de streming foi desenvolvido de forma faseada e gradual. A mesma seguiu as seguintes etapas, ordenadas cronologicamente:

5.1 Primeiros Passos & Construção da Topologia Overlay

Nesta primeira fase do projeto, começamos por construir um programa simples que permitia a comunicação entre dois nodos diferentes.

Havendo comunicação, começamos por implementar a construção da rede overlay. A estratégia utilizada é baseada num controlador. Um nodo da topologia funciona como bootstrapper, tendo conhecimento de toda a topologia da rede. Quando um nodo se conecta a topologia, faz um pedido ao bootstrapper a pedir informações sobre os seus vizinhos de modo a formar conexões.

5.2 Serviço de Streaming

Para inicializarmos o serviço de Streaming na rede, o cliente deve efetuar um pedido ao Servidor. Uma vez que os diferentes nodos da rede já sabem qual o melhor caminho até ao servidor, estes encaminham a mensagem para o mesmo, ativando e guardando as rotas ativas para o qual os conteúdos devem ser transmitidos. Assim que o pedido chega ao servidor, o mesmo começa a enviar pacotes de streaming. Quando o cliente decide não receber mais informação da transmissão, envia um pacote ao servidor vizinho, de modo a que a rota de streaming construída para o cliente seja ”destruída”.

5.3 Monitorização da Rede Overlay

A nossa abordagem a este problema teve como solução apenas existirem mensagens de monitorização a partir dos servidores. Para tal, o servidor envia um pacote de monitorização para a rede a cada 10 segundos. Cada nó que recebe este pacote, atualiza na sua tabela de rotas o melhor caminho para chegar ao servidor remetente do pacote caso as métricas do mesmo forem melhores. De seguida, atualiza o pacote com os seus dados e reenvia aos seus vizinhos exceto àquele do qual recebeu o pacote (inundação controlada).

Desta forma, conseguimos nos adaptar a diferentes cenários de rede permitindo sempre que a transmissão de dados percorra as rotas mais rápidas.

5.4 Construção de Rotas para a Entrega de Dados

Para a construção de rotas para a entrega de dados, fazemos uso dos pacotes de monitorização da rede para a construção das rotas até aos servidores remetentes das mensagens. Através destas, construimos tabelas de encaminhamento com informação sobre a Origem/Servidor, Remetente, número de nodos até ao servidor, tempo de envio desde o servidor até ao nodo.

Como fazemos uso de mensagens periódicas para a construção de rotas, asseguramos que as tabelas de encaminhamento estão sempre atualizadas sobre as rotas com prestação de melhores velocidades.

5.5 Ativação e Teste do Servidor Alternativo

Quando o serviço de streaming do servidor é interrompido, um serviço de streaming do servidor alternativo é ativo, garantido a contínua transmissão de dados mesmo em cenários de falha.

Para implementarmos esta solução, fazemos uso novamente dos pacotes de monitorização (Probe Packet). O nodo cliente, assim que deteta uma melhor rota para obter a stream de conteúdo, altera o fluxo da mesma para o novo servidor e manda um novo pedido de streaming.

6 Testes e Resultados

A topologia usada para testes e verificação de resultados é demonstrada na *Figura 1.* O software de suporte à criação e simulação de transmissão de dados foi o *Core.*

Para a construção da topologia, como é possível verificar na *Figura 2*, o nodo O2 e O3 quando iniciados, efetuam um pedido da listinha de vizinhos ao servidor ativando as rotas para os seus vizinhos.

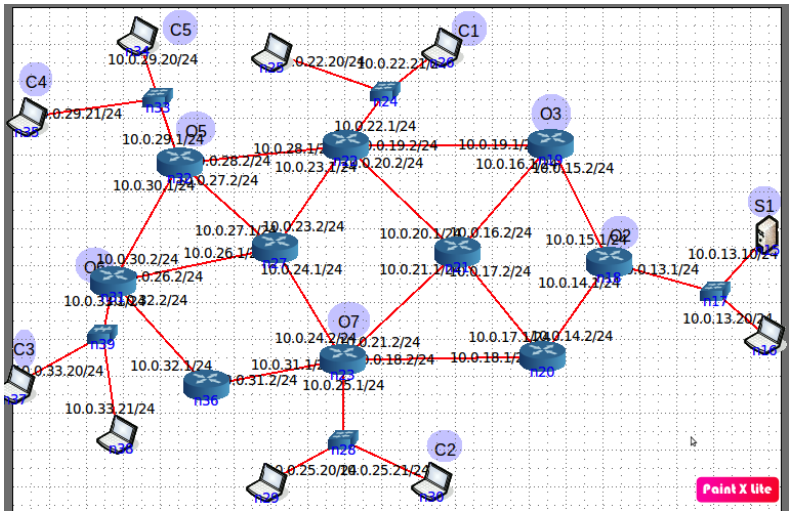

Fig. 1. Topologia da Rede

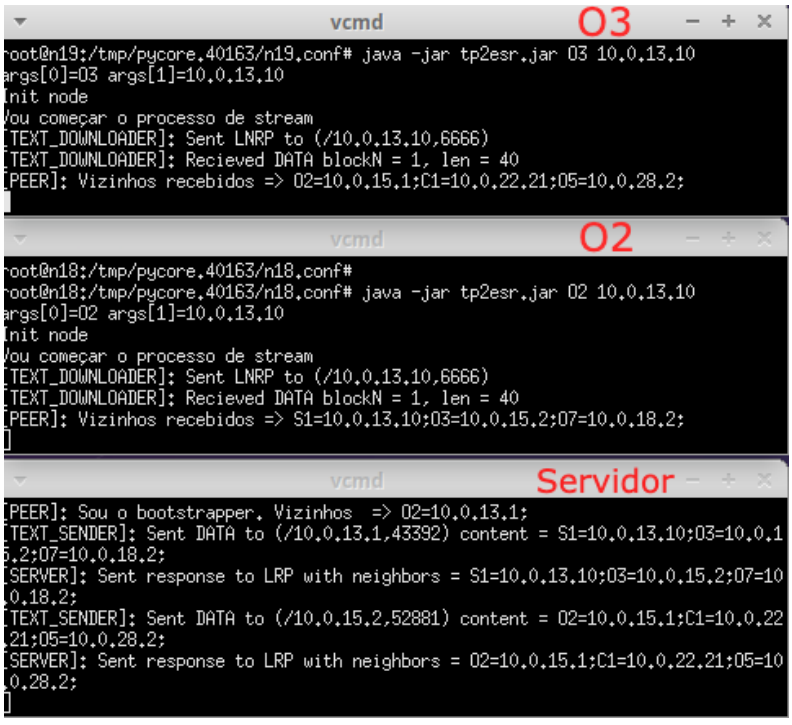

Fig. 2. Construção da Topologia Overlay

A verificação de resultados aquando da transmissão de dados no nosso serviço pode ser confirmada como exposto na *Figura 3*.

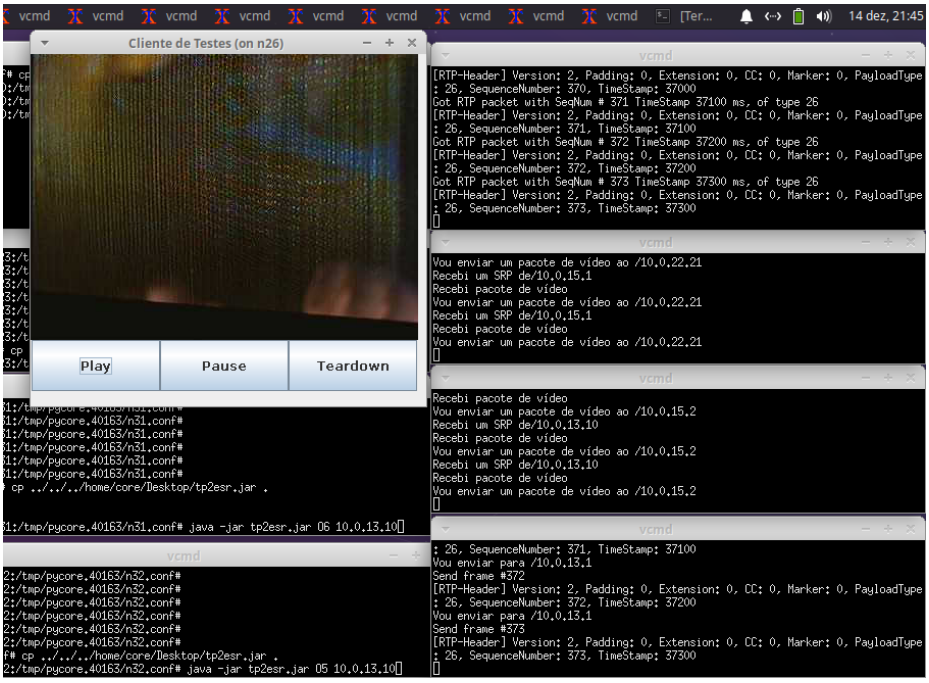

Fig. 3. Streaming no Cliente

Na figura 4 é possível verificar o envio de pacotes de monitorização, e através desses a construção de tabelas de encaminhamento em cada nodo. Nota: Neste teste, foi adicionado um Servidor Alternativo "S2", com ligação ao nodo O5. Esta nova ligação foi adicionada também ao ficheiro bootstrapper.

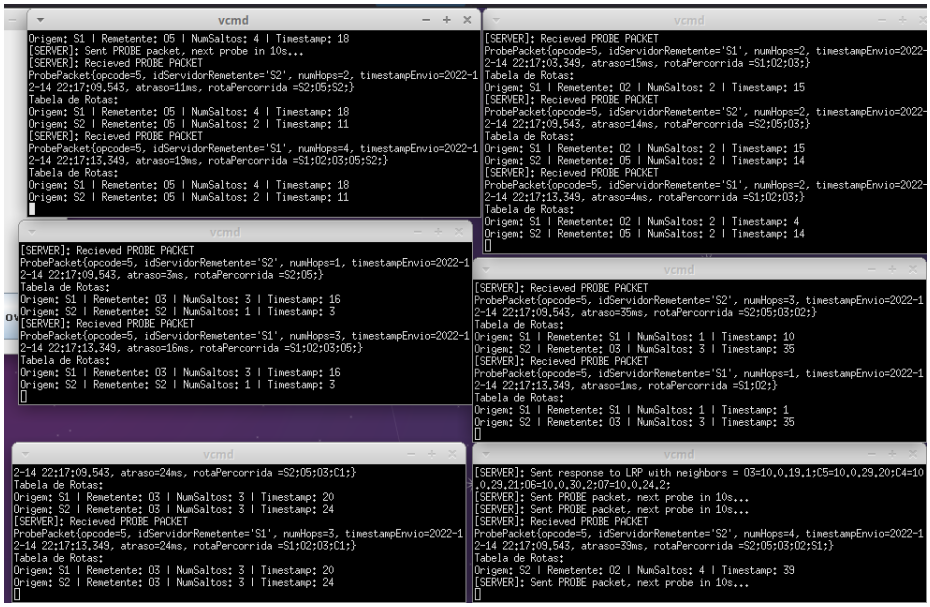

Fig. 4. Flooding e Construção de Rotas

7 Conclusões

Fazendo uma análise final do projeto desenvolvido tendo em conta os objetivos propostos, demonstramos bastante agrado para com o mesmo. Sentimos que foi um trabalho desafiante e complexo, que pediu muita discussão e atenção aos pormenores uma vez que as alternativas de soluções são múltiplas e incubia-nos analisar os prós e contras de cada uma, o que, para além de desenvolver competências técnicas permitiu aumentar as capacidades de trabalho de grupo e de espírito crítico.

Sentimos que embora o nosso projeto cumpra todos os objetivos propostos, à medida que fomos desenvolvendo o serviço fomos nos apercebendo de melhores estratégias que poderiam ser implementadas no que toca ao suporte e funcionamento do serviço, mas que por limitações de tempo foram descartadas.