

# Relatório Trabalho Final

## Alunos:

Arthur Henrique Neves Dias

Caio Henrique Alvarenga Gonçalves

Felipe Augusto M. Constantino

Henrique Mendonça Castelar Campos

João Paolinelli e Silva

Pedro Corrêa Rigotto

- **Video youtube:**

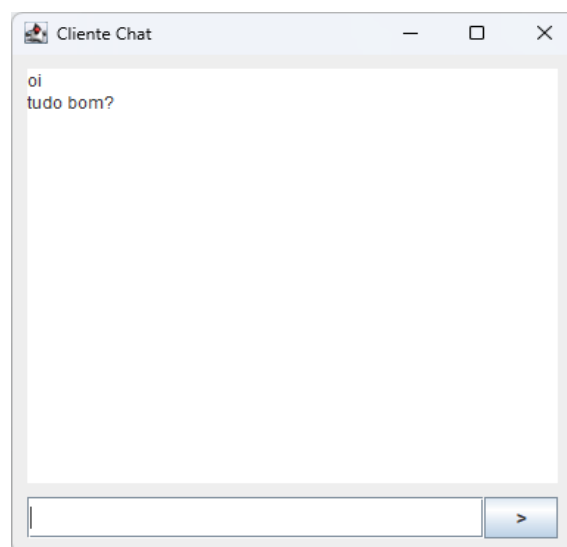
<https://www.youtube.com/watch?v=fN6CRID14SI>

- **Objetivo**

Neste trabalho prático, buscamos criar uma aplicação de rede usando Java. A aplicação deve utilizar TCP e UDP, conter multithreading e interface gráfica usando Swing.

- **Relatório do Desenvolvimento:**

Foi desenvolvida uma aplicação em Java com interface gráfica em Swing. O que escolhemos foi um aplicativo de chat em texto e voz, utilizando TCP para o envio de texto e UDP para a comunicação via áudio. Foram feitas várias threads, uma para cada funcionalidade, sendo elas a thread do cliente de texto, uma para a funcionalidade de áudio, que se separa em mais duas, para envio e recebimento de áudio, além de uma thread para o servidor de áudio.



*Exemplo da interface do cliente*

O código é separado entre cliente e servidor. O servidor hospeda o chat em texto, e envia o histórico todo da conversa em cada requisição de leitura que recebe. O aplicativo do cliente também engloba um servidor de áudio. Dois clientes são conectados entre si, cada um configurado com o IP do outro, e assim é realizada a conversa via áudio. Apenas um servidor de texto é disponibilizado para qualquer número de clientes. Os clientes requisitam periodicamente atualização do log da conversa, e enviam mensagens a serem adicionadas ao log digitando a mensagem na caixa e apertando o botão de envio. Deste modo, até clientes que se juntarem à conversa após mensagens serem enviadas terão acesso ao histórico. O servidor de texto não possui interface gráfica.

- **Desafios**

Os nossos principais desafios foram o uso de áudio, a configuração dos roteadores e o trabalho em equipe. Tivemos complicações ao tentar conciliar servidor e cliente no mesmo aplicativo, porém esses problemas foram resolvidos utilizando multithreading. Também houve confusão ao conciliar dois tipos de IPs diferentes (do servidor de texto e do outro cliente de voz), o que causou erros como o cliente conversar consigo mesmo, porém esses erros foram corrigidos.

- **Código fonte:**

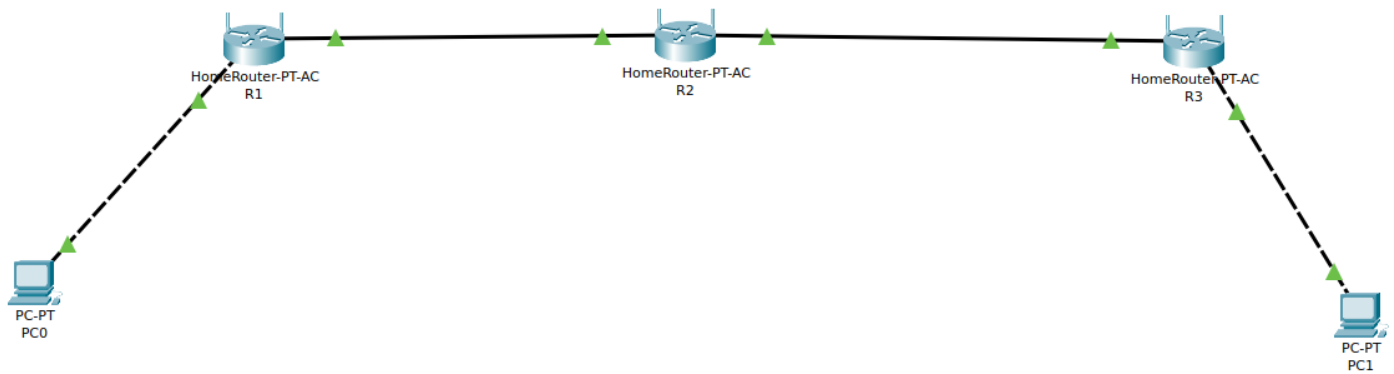
***Repositório configuração servidor e cliente & funcionalidade áudio e mensagem:***

[https://github.com/Henriquemcc/Trabalho\\_Pratico\\_-\\_Redes\\_de\\_Computadores\\_I\\_-\\_2023-2/tree/main/Aplicacao](https://github.com/Henriquemcc/Trabalho_Pratico_-_Redes_de_Computadores_I_-_2023-2/tree/main/Aplicacao)

***Configuração dos roteadores:***

[https://github.com/Henriquemcc/Trabalho\\_Pratico\\_-\\_Redes\\_de\\_Computadores\\_I\\_-\\_2023-2/tree/main/Roteadores](https://github.com/Henriquemcc/Trabalho_Pratico_-_Redes_de_Computadores_I_-_2023-2/tree/main/Roteadores)

- **Projeto no Cisco Packet Tracer:**



**Descrição dos elementos:**

**PC0:** Este é um computador de usuário final conectado ao roteador R1.

**Roteadores (R1, R2, R3):** Estes são os roteadores da série Home Router-PT-AC. Eles são usados para conectar diferentes segmentos de rede. R1 está conectado ao PC0, R3 está conectado ao PC1, e R2 está no meio, atuando como um dispositivo intermediário para o tráfego entre os PCs e possivelmente para outras redes não mostradas no diagrama.

**PC1:** Este é outro computador de usuário final conectado ao roteador R3.

As linhas contínuas representam conexões diretas (cabeadas, assumindo pelo ícone de conector Ethernet), e as linhas pontilhadas representam conexões sem fio.

Os pequenos triângulos verdes indicam a presença de atividade de rede ou a funcionalidade correta dos dispositivos em um determinado momento.

Esse tipo de diagrama é útil para entender como os dispositivos estão interconectados, para planejar ou testar configurações de rede antes de implementá-las em equipamentos reais.

- Avaliação no Wireshark:

PC1.pcapng										PC2.pcapng									
No.	Time	Source	Destination	Protocol	Length	Info				No.	Time	Source	Destination	Protocol	Length	Info			
5067	1009.5063775..	172.16.0.2	192.168.0.3	TCP	54	6790 → 28721 [ACK] Seq=1 Ack=2				6486	1034.955881	192.168.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024			
5068	1009.5070088..	192.168.0.3	172.16.0.2	TCP	60	28721 → 6790 [FIN, PSH, ACK] Seq=8 Ack=2				6487	1034.985635	192.168.0.3	192.168.0.2	UDP	1066	9879 → 9876 Len=1024			
5069	1009.5072058..	172.16.0.2	192.168.0.3	TCP	54	6790 → 28721 [FIN, ACK] Seq=1 Ack=2				6488	1034.985993	192.168.0.3	192.168.0.2	UDP	1066	9879 → 9876 Len=1024			
5070	1009.5077163..	192.168.0.3	172.16.0.2	TCP	60	28721 → 6790 [ACK] Seq=8 Ack=2				6489	1035.055245	192.168.0.3	192.168.0.2	TCP	66	28674 → 6790 [SYN] Seq=0 Win=6			
5071	1009.5078115..	192.168.0.3	172.16.0.2	UDP	1066	9879 → 9876 Len=1024				6490	1035.055836	192.168.0.3	192.168.0.2	TCP	66	6790 → 28674 [SYN, ACK] Seq=8			
5072	1009.5078430..	192.168.0.3	172.16.0.2	UDP	1066	9879 → 9876 Len=1024				6491	1035.055999	192.168.0.3	192.168.0.2	TCP	54	28674 → 6790 [ACK] Seq=1 Ack=1			
5073	1009.5745411..	172.16.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024				6492	1035.057378	192.168.0.3	192.168.0.2	TCP	55	28674 → 6790 [PSH, ACK] Seq=1 Ack=1			
5074	1009.5745734..	172.16.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024				6493	1035.057936	192.168.0.2	192.168.0.3	TCP	60	6790 → 28674 [ACK] Seq=1 Ack=2			
5075	1009.6680653..	172.16.0.2	192.16.0.2	UDP	1066	9879 → 9876 Len=1024				6494	1035.057989	192.168.0.3	192.168.0.2	TCP	58	28674 → 6790 [PSH, ACK] Seq=2 Ack=2			
5076	1009.6687443..	192.168.0.3	172.16.0.2	UDP	1066	9879 → 9876 Len=1024				6495	1035.058372	192.168.0.2	192.168.0.3	TCP	60	6790 → 28674 [ACK] Seq=1 Ack=6			
5077	1009.6997336..	172.16.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024				6496	1035.058372	192.168.0.2	192.168.0.3	TCP	60	6790 → 28674 [PSH, ACK] Seq=1 Ack=6			
5078	1009.6997040..	172.16.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024				6497	1035.058372	192.168.0.2	192.168.0.3	TCP	60	6790 → 28674 [FIN, PSH, ACK] Seq=1 Ack=6			
5079	1009.7936813..	192.168.0.3	172.16.0.2	UDP	1066	9879 → 9876 Len=1024				6498	1035.058509	192.168.0.3	192.168.0.2	TCP	54	28674 → 6790 [ACK] Seq=6 Ack=4			
5080	1009.7948980..	192.168.0.3	172.16.0.2	UDP	1066	9879 → 9876 Len=1024				6499	1035.058769	192.168.0.3	192.168.0.2	TCP	54	28674 → 6790 [ACK] Seq=6 Ack=5			
5081	1009.8249074..	172.16.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024				6500	1035.060095	192.168.0.3	192.168.0.2	TCP	54	28674 → 6790 [FIN, ACK] Seq=6 Ack=7			
5082	1009.8249465..	172.16.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024				6501	1035.060485	192.168.0.2	192.168.0.3	TCP	60	6790 → 28674 [ACK] Seq=5 Ack=7			
5083	1009.9063165..	192.168.0.3	172.16.0.2	TCP	66	28722 → 6790 [SYN] Seq=0 Win=6				6502	1035.081073	192.168.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024			
5084	1009.9063866..	172.16.0.2	192.168.0.3	TCP	66	6790 → 28722 [SYN, ACK] Seq=0 Ack=1				6503	1035.081073	192.168.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024			
5085	1009.9069024..	192.168.0.3	172.16.0.2	TCP	60	28722 → 6790 [ACK] Seq=1 Ack=1				6504	1035.112060	192.168.0.3	192.168.0.2	UDP	1066	9879 → 9876 Len=1024			
5086	1009.9091362..	192.168.0.3	172.16.0.2	TCP	60	28722 → 6790 [PSH, ACK] Seq=1 Ack=1				6505	1035.112275	192.168.0.3	192.168.0.2	UDP	1066	9879 → 9876 Len=1024			
5087	1009.9091724..	172.16.0.2	192.168.0.3	TCP	54	6790 → 28722 [ACK] Seq=1 Ack=2				6506	1035.206440	192.168.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024			
5088	1009.9141733..	192.168.0.3	172.16.0.2	TCP	60	28722 → 6790 [PSH, ACK] Seq=2 Ack=2				6507	1035.206440	192.168.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024			
5089	1009.9142034..	172.16.0.2	192.168.0.3	TCP	54	6790 → 28722 [ACK] Seq=1 Ack=6				6508	1035.237403	192.168.0.3	192.168.0.2	UDP	1066	9879 → 9876 Len=1024			
5090	1009.9143370..	172.16.0.2	192.168.0.3	TCP	55	6790 → 28722 [PSH, ACK] Seq=1 Ack=6				6509	1035.237562	192.168.0.3	192.168.0.2	UDP	1066	9879 → 9876 Len=1024			
5091	1009.9153072..	172.16.0.2	192.168.0.3	TCP	62	6790 → 28722 [FIN, PSH, ACK] Seq=1 Ack=6				6510	1035.296073	fe80::d6d3:86c7::f602::1:ffa9:c191	ff02::1:ffa9:c191	ICMPv6	80	Neighbor Solicitation for fe80::d6d3:86c7::f602::1:ffa9:c191			
5092	1009.9155043..	192.168.0.3	172.16.0.2	TCP	60	28722 → 6790 [ACK] Seq=6 Ack=16				6511	1035.331752	192.168.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024			
5093	1009.9155913..	192.168.0.3	172.16.0.2	TCP	60	28722 → 6790 [ACK] Seq=6 Ack=11				6512	1035.331752	192.168.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024			
5094	1009.9181857..	192.168.0.3	172.16.0.2	TCP	60	28722 → 6790 [FIN, ACK] Seq=6 Ack=11				6513	1035.363342	192.168.0.3	192.168.0.2	UDP	1066	9879 → 9876 Len=1024			
5095	1009.9182103..	172.16.0.2	192.168.0.3	TCP	54	6790 → 28722 [ACK] Seq=11 Ack=2				6514	1035.363629	192.168.0.3	192.168.0.2	UDP	1066	9879 → 9876 Len=1024			
5096	1009.9199981..	192.168.0.3	172.16.0.2	UDP	1066	9879 → 9876 Len=1024				6515	1035.456962	192.168.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024			
5097	1009.9202044..	192.168.0.3	172.16.0.2	UDP	1066	9879 → 9876 Len=1024				6516	1035.456962	192.168.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024			
5098	1009.9501500..	172.16.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024				6517	1035.488295	192.168.0.3	192.168.0.2	UDP	1066	9879 → 9876 Len=1024			
5099	1009.9502515..	172.16.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024				6518	1035.488529	192.168.0.3	192.168.0.2	UDP	1066	9879 → 9876 Len=1024			
5100	1010.0451563..	192.168.0.3	172.16.0.2	UDP	1066	9879 → 9876 Len=1024				6519	1035.490478	D-LinkIn_4d:8c:79	Broadcast	ARP	60	who has 192.168.255.169? Tell me			
5101	1010.0452865..	192.168.0.3	172.16.0.2	UDP	1066	9879 → 9876 Len=1024				6520	1035.490478	D-LinkIn_4d:8c:79	Broadcast	ARP	60	who has 192.168.255.168? Tell me			
5102	1010.0768900..	172.16.0.2	192.168.0.3	UDP	1066	9879 → 9876 Len=1024				6521	1035.490675	D-LinkIn_4d:8c:79	Broadcast	ARP	60	who has 192.168.255.167? Tell me			
<div>&gt; Frame 5077: 1066 bytes on wire (8528 bits), 1066 bytes captured</div> <div>Ethernet II, Src: 92:f9:7d:85:4d:d6 (92:f9:7d:85:4d:d6), Dst: Tp</div> <div>Internet Protocol Version 4, Src: 172.16.0.2, Dst: 192.168.0.3</div> <div>User Datagram Protocol, Src Port: 9879, Dst Port: 9876</div> <div>Source Port: 9879</div> <div>Destination Port: 9876</div> <div>Length: 1032</div> <div>Checksum: 0x70d7 [unverified]</div> <div>[Checksum Status: Unverified]</div> <div>[Stream index: 6]</div> <div>[Timestamps]</div> <div>&gt; Data (1024 bytes)</div>										<div>&gt; Internet Protocol Version 4, Src: 192.168.0.3, Dst: 192.168.0.3</div> <div>Transmission Control Protocol, Src Port: 28674, Dst Port: 6790</div> <div>Source Port: 28674</div> <div>Destination Port: 6790</div> <div>[Stream index: 43]</div> <div>[Conversation completeness: Complete, WITH_DATA (31)]</div> <div>[TCP Segment Len: 4]</div> <div>Sequence Number: 2 (relative sequence number)</div> <div>Sequence Number (raw): 486930433</div> <div>[Next Sequence Number: 6 (relative sequence number)]</div> <div>Acknowledgment Number: 1 (relative ack number)</div> <div>Acknowledgment number (raw): 2165170479</div> <div>0101 ..... = Header Length: 20 bytes (5)</div> <div>..... = Data offset (new src)</div>									
PC1.pcapng										PC2.pcapng									
Packets: 10064 / Displayed: 10064 (100.0%)										Packets: 14788 / Displayed: 14788 (100.0%)									

## Logs da execução em ambos os computadores

Foi feita a gravação das mensagens no Wireshark enquanto era feita a execução dos códigos para a gravação do vídeo. Como podemos verificar na imagem acima, existem mensagens TCP e UDP sendo transmitidas entre os computadores. No PC2, foi executado o código do servidor de texto. Uma sequência de mensagens TCP são trocadas a cada segundo, para atualizar o log do chat. As mensagens UDP identificam a transmissão de áudio, e podemos ver que ela acontece em ambas as direções.