



UNIVERSIDADE
FEDERAL
DE PERNAMBUCO

**UNIVERSIDADE FEDERAL DE PERNAMBUCO – CAMPUS RECIFE
INFRAESTRUTURA DE COMUNICAÇÃO**

RAFAEL SILVA MOURA

RELATÓRIO DO PROJETO DE SOCKET

Recife - PE

Dezembro 2025.

APRESENTAÇÃO

Este projeto foi desenvolvido como trabalho final da disciplina **Infraestrutura de Comunicação**, no **Centro de Informática (CIN)** da **UFPE**. O objetivo principal foi implementar aplicações cliente-servidor utilizando os **protocolos TCP e UDP**, acompanhadas de um Servidor de Nomes para registro e consulta de serviços disponíveis.

Para exemplificar o funcionamento, foi criado um **serviço de calculadora remota**, permitindo que os clientes realizassem operações matemáticas simples. As aplicações registram automaticamente os serviços no **Servidor de Nomes** ao serem iniciadas, e os clientes podem verificar a disponibilidade do serviço, incluindo testes de serviços inexistentes para validar o retorno “NOT_FOUND”.

O projeto inclui a **medição do tempo de envio e recebimento das mensagens**, análise do desempenho dos protocolos e captura de pacotes via Wireshark, proporcionando compreensão prática da diferença entre TCP e UDP e do mecanismo de registro e descoberta de serviços em rede.

Sumário

.....	1
APRESENTAÇÃO.....	2
OBJETIVOS E REQUISITOS DO PROJETO.....	4
ARQUITETURA DO SISTEMA.....	5
IMPLEMENTAÇÃO.....	6
Servidor TCP e Cliente TCP:.....	6
Servidor UDP e Cliente UDP:.....	6
Servidor de Nomes:.....	6
Cliente de teste para serviço inexistente:.....	6
EXECUÇÃO E TESTES.....	7
FIGURA 1 – COMPORTAMENTOS DOS SEVIDORES.....	7
FIGURA 2 – COMPORTAMENTOS DOS CLIENTES.....	7
DESCRIPÇÃO TÉCNICA DOS ARQUIVOS E JUSTIFICATIVA DAS CAPTURAS.....	8
1. Descrição do Serviço Desenvolvido.....	8
1.1 Servidor de Nomes (<i>service_names.py</i>).....	8
1.2 Serviço de Calculadora TCP (<i>service_tcp.py</i>).....	8
1.3 Serviço de Calculadora UDP (<i>service_udp.py</i>).....	9
2. Análise das Capturas Wireshark.....	9
2.1 Registro do Serviço TCP no Servidor de Nomes.....	9
2.2 Registro do Serviço UDP no Servidor de Nomes.....	10
2.3 Consulta ao Servidor de Nomes.....	10
2.4 Comunicação UDP – Cálculo Remoto.....	11
2.5 Comunicação TCP – Cálculo Remoto.....	12
3. Comparação dos Tempos TCP x UDP.....	13
3.1 Desempenho do UDP.....	13
3.2 Desempenho do TCP.....	13
3.3 Conclusão da Comparação.....	13
ANÁLISE DE DESEMPENHO.....	14
CONSIDERAÇÕES FINAIS.....	15

OBJETIVOS E REQUISITOS DO PROJETO

O projeto tem como objetivos **implementar aplicações cliente-servidor** utilizando os protocolos **TCP e UDP**, permitindo a comunicação entre processos em rede de forma prática. Para exemplificar o funcionamento, foi criado um **serviço de calculadora remota**, no qual os clientes podem realizar operações matemáticas simples e receber os resultados de forma confiável no TCP ou de forma rápida no UDP.

Além disso, o projeto desenvolve um **Servidor de Nomes** que permite registrar e consultar serviços, garantindo que os clientes possam localizar os servidores disponíveis e validar serviços inexistentes, recebendo corretamente o retorno “NOT_FOUND”. Outro objetivo importante é **analisar o desempenho dos protocolos de transporte**, medindo o tempo de envio e recebimento das mensagens, e **capturar o tráfego de rede via Wireshark**, permitindo comparar as características de TCP e UDP na prática. Por fim, o projeto busca **proporcionar aprendizado aplicado** sobre arquitetura cliente-servidor, protocolos de transporte e descoberta de serviços em rede.

Para atingir esses objetivos, foram estabelecidos **requisitos funcionais e não funcionais**. Entre os **requisitos funcionais**, destacam-se a implementação de servidores e clientes TCP e UDP capazes de processar operações matemáticas simples, o registro de serviços no Servidor de Nomes, a possibilidade de consultar serviços existentes e validar serviços inexistentes com o retorno “NOT_FOUND”, além da medição do tempo total de comunicação entre cliente e servidor.

Já os **requisitos não funcionais** incluem a execução simultânea das aplicações TCP e UDP, garantindo independência entre os processos, a utilização de valores pré-determinados para evitar interferência na medição de desempenho, a exibição de mensagens coloridas no terminal para facilitar a visualização de status e resultados, e a captura de pacotes via Wireshark para análise detalhada do tráfego de rede, permitindo uma avaliação prática das diferenças entre TCP e UDP.

ARQUITETURA DO SISTEMA

O sistema desenvolvido segue a **arquitetura cliente-servidor**, na qual os clientes enviam solicitações e os servidores processam as operações, retornando os resultados. Existem **dois tipos de servidores**, TCP e UDP, que operam de **forma independente**, mas oferecem o **mesmo serviço de calculadora remota**. Todos os servidores se registram automaticamente no Servidor de Nomes ao serem inicializados, permitindo que os clientes descubram os serviços disponíveis e obtenham o endereço IP e a porta correspondentes.

O Servidor de Nomes responde com “NOT_FOUND” caso o serviço solicitado **não esteja registrado**. A comunicação TCP garante confiabilidade e entrega ordenada das mensagens, enquanto a comunicação UDP proporciona maior rapidez, ainda que sem garantias de entrega. Essa arquitetura permite **executar testes simultâneos** dos dois protocolos, medir o tempo de envio e recebimento das mensagens e analisar o tráfego de rede por meio de capturas no **Wireshark**, proporcionando compreensão prática das diferenças entre TCP e UDP e do mecanismo de registro e descoberta de serviços.

IMPLEMENTAÇÃO

Na seção a seguir, são detalhadas as implementações dos diferentes componentes do sistema, incluindo os servidores e clientes TCP e UDP, o Servidor de Nomes e o cliente de teste para serviço inexistente. **Serão descritas as funcionalidades de cada elemento, o fluxo de comunicação entre clientes e servidores, bem como os mecanismos de registro e consulta de serviços.** Além disso, são destacadas as configurações utilizadas para medição de tempo de execução e a análise do tráfego de rede por meio de capturas no Wireshark, permitindo compreender o comportamento prático do sistema.

Servidor TCP e Cliente TCP:

- I. O servidor TCP aguarda conexões de clientes e, para cada conexão, envia um menu de operações matemáticas.
- II. Os clientes enviam a operação escolhida e os números, e recebem o resultado calculado.
- III. Ao encerrar, o servidor atualiza o Servidor de Nomes, removendo o serviço registrado.

Servidor UDP e Cliente UDP:

- I. O servidor UDP opera sem conexão, recebendo datagramas contendo a operação e os valores, processando e retornando o resultado diretamente ao cliente.
- II. Os clientes enviam os pacotes de forma automática e recebem a resposta, permitindo medir o tempo de comunicação.
- III. Também é realizado o registro e remoção do serviço no Servidor de Nomes.

Servidor de Nomes:

- I. Permite que servidores registrem serviços com IP, porta e nome do serviço.
- II. Atende consultas de clientes, retornando os dados do serviço ou "NOT_FOUND" caso não exista.
- III. Mantém um controle de serviços ativos, permitindo consultas confiáveis pelos clientes.

Cliente de teste para serviço inexistente:

Envia solicitações de um serviço que não foi registrado, verificando se o Servidor de Nomes retorna corretamente "NOT_FOUND".

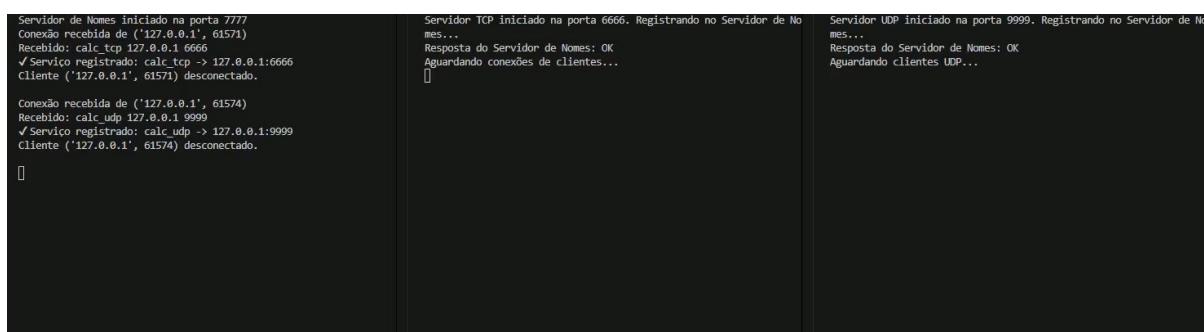
Além disso, todas as aplicações foram configuradas para **medir o tempo total de execução** e enviar mensagens coloridas no terminal, facilitando a visualização do status e dos resultados. A captura de pacotes via **Wireshark** permitiu analisar o tráfego e comparar o desempenho entre TCP e UDP de forma prática.

EXECUÇÃO E TESTES

Para testar o funcionamento do sistema, **os servidores TCP e UDP foram iniciados em terminais separados**, juntamente com o Servidor de Nomes. Em seguida, os clientes foram executados de forma automática, enviando operações matemáticas pré-determinadas para cada servidor. Um dos clientes consultou o serviço TCP, outro o serviço UDP, e um terceiro cliente enviou uma requisição para um serviço inexistente, verificando o retorno “NOT_FOUND”. Durante os testes, os servidores processaram corretamente as requisições, retornando os resultados esperados aos clientes, enquanto o Servidor de Nomes respondeu conforme o registro de cada serviço. **Todos os pacotes trocados foram capturados via Wireshark**, permitindo analisar detalhadamente o tráfego de rede e validar o funcionamento do sistema.

A análise temporal das operações mostrou que, apesar do UDP ser **teoricamente mais rápido** por não exigir handshake nem confirmação de entrega, o **TCP apresentou tempo total menor** (0,000345 segundos) em comparação ao UDP (0,000542 segundos). Isso ocorreu porque, em operações muito pequenas, como a soma de dois números, o TCP se beneficia do gerenciamento de **buffers do sistema operacional**, que permite agrupar pacotes e enviar múltiplos dados de uma só vez, enquanto no UDP cada pacote é tratado de **forma independente, aumentando o overhead**. Em redes locais, onde a latência de transmissão é mínima, esse efeito se torna mais evidente, **favorecendo o TCP**. Apesar desse resultado, o UDP continua sendo vantajoso em aplicações que **enviam grandes volumes de dados** sem necessidade de confiabilidade, enquanto o TCP garante entrega confiável e ordenada, sendo ideal para aplicações que necessitam consistência nos dados.

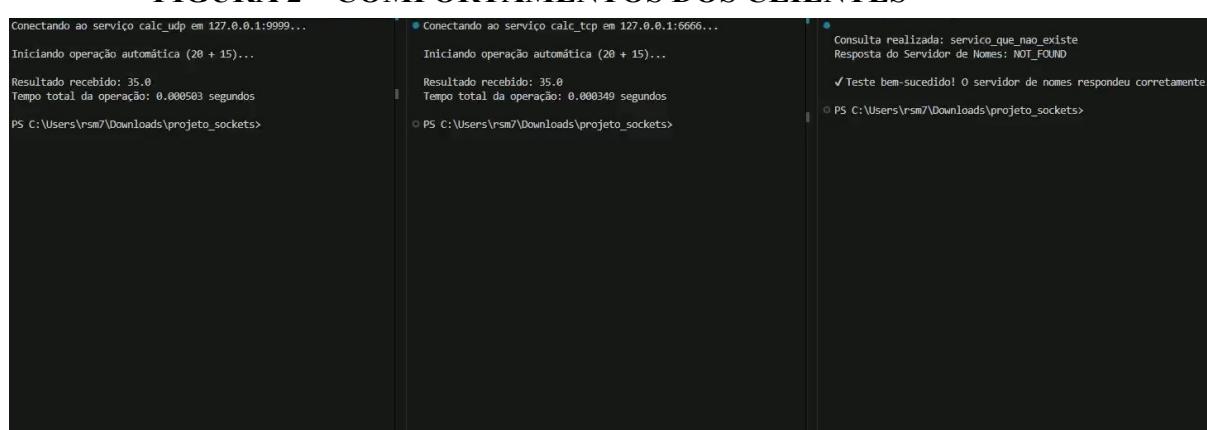
FIGURA 1 – COMPORTAMENTOS DOS SEVIDORES



The figure displays three terminal windows illustrating the behavior of different servers:

- Servidor de Nomes iniciado na porta 7777**: Shows a connection from a client at port 61571. The response "OK" indicates the service was registered successfully.
- Servidor TCP iniciado na porta 6666. Registrando no Servidor de Nomes...**: Shows a connection from a client at port 61574. The response "OK" indicates the service was registered successfully.
- Servidor UDP iniciado na porta 9999. Registrando no Servidor de Nomes...**: Shows a connection from a client at port 61574. The response "OK" indicates the service was registered successfully.

FIGURA 2 – COMPORTAMENTOS DOS CLIENTES



The figure displays three terminal windows illustrating client behaviors:

- Conectando ao serviço calc_udp em 127.0.0.1:9999...**: Shows a connection attempt to the UDP server. The result is a sum of 35.0, with a total time of 0.000503 seconds.
- Conectando ao serviço calc_tcp em 127.0.0.1:6666...**: Shows a connection attempt to the TCP server. The result is a sum of 35.0, with a total time of 0.000349 seconds.
- Consulta realizada: serviço_não_existe**: Shows a query to the Name Server for a non-existent service. The response is "NOT_FOUND". A note indicates the test was successful because the name server responded correctly.

DESCRIÇÃO TÉCNICA DOS ARQUIVOS E JUSTIFICATIVA DAS CAPTURAS

Nesta sessão será apresentada a **Descrição do serviço de comunicação desenvolvido**, seguido da análise detalhada das capturas realizadas utilizando o Wireshark (arquivo **capturas_wireshark.pcapng**). O objetivo é correlacionar o comportamento observado durante a execução dos serviços com os protocolos de transporte empregados (TCP e UDP) e, por fim, realizar uma comparação estruturada entre os tempos de resposta de cada protocolo.

Obs: Nas descrições das capturas no Wireshark, não mencionarei explicitamente os pacotes de confirmação ACK. Entretanto, considere que para cada pacote enviado houve o respectivo recebimento e confirmação.

1. Descrição do Serviço Desenvolvido

O sistema é composto por **três partes principais**: um Servidor de Nomes, um Serviço de Calculadora TCP e um Serviço de Calculadora UDP. Cada componente possui uma **função específica** dentro da arquitetura.

1.1 Servidor de Nomes (*service_names.py*)

O Servidor de Nomes funciona como um **diretório centralizado**. Sua função é receber o registro dos serviços que desejam ser localizados na rede, armazenando seu nome, endereço e porta. Além disso, **responde a consultas de clientes** que precisam descobrir onde determinado serviço está ativo.

Esse mecanismo permite que clientes e servidores funcionem de forma **desacoplada**, sem a necessidade de configurar portas manualmente para cada comunicação.

1.2 Serviço de Calculadora TCP (*service_tcp.py*)

A versão TCP da calculadora estabelece uma **conexão confiável** com o cliente. Após o estabelecimento da conexão, o servidor envia um menu de operações matemáticas, recebe a escolha do cliente e solicita os números. Todo o fluxo ocorre de maneira **sequencial e controlada**, com garantia de entrega.

Esse serviço registra-se previamente no Servidor de Nomes, informando o nome do serviço, IP e porta de funcionamento.

1.3 Serviço de Calculadora UDP (*service_udp.py*)

A versão UDP da calculadora realiza operações semelhantes, porém sem estabelecer conexão. Cada troca entre cliente e servidor ocorre por meio de datagramas independentes. Apesar de teoricamente ser mais rápido, esse método não garante entrega nem ordenação, mas é suficiente para operações simples.

Assim como a versão TCP, o serviço envia seu registro ao Servidor de Nomes no momento em que inicia.

2. Análise das Capturas Wireshark



Nesta seção, além da explicação dos eventos, são indicados os frames exatos do arquivo ***capturas_wireshark.pcapng*** onde cada evento pode ser encontrado. A seguir estão descritos os principais eventos observados no arquivo ***capturas_wireshark.pcapng***, acompanhados das indicações dos pacotes correspondentes para inserção das imagens.

2.1 Registro do Serviço TCP no Servidor de Nomes

Nos primeiros pacotes da captura, observa-se o estabelecimento da conexão TCP com o Servidor de Nomes e, em seguida, o **envio da mensagem contendo o nome do serviço** e sua porta de operação.

Imagen:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	127.0.0.1	127.0.0.1	TCP	56	58381 → 7777 [SYN] Seq=0
2	0.000049	127.0.0.1	127.0.0.1	TCP	56	7777 → 58381 [SYN, ACK]
3	0.000106	127.0.0.1	127.0.0.1	TCP	44	58381 → 7777 [ACK] Seq=1

(3-way handshake do serviço TCP para com o servidor de nomes)

> Frame 4: 67 bytes on wire (536 bits), 67 bytes captured (536 bits) > Null/Loopback > Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1 > Transmission Control Protocol, Src Port: 58381, Dst Port: 7777, Seq=1, Ack=1, Len=44 Data (23 bytes) Data: 63616c635f746370203132372e302e302e31203636363636 [Length: 23]	00 00 00 45 00 00 3f 82 57 40 00 80 06 00 00 00 00 01 7f 00 00 01 e4 0d 1e 61 58 6f a0 56 da 2d 4f 50 18 00 ff a1 45 00 00 63 61 6c 63 74 63 70 20 31 32 37 2e 30 2e 30 2e 31 20 36 36 36E..? .W@.....aXo-V ...OP.... E..calc .tcp 127.0.0.1 6 666
---	---	---

(frame 4: registro do serviço calc_tcp para o endereço 127.0.0.01 com a porta 6666)

2.2 Registro do Serviço UDP no Servidor de Nomes

Após o registro TCP, observa-se nova conexão TCP sendo aberta, desta vez referindo-se ao registro do serviço UDP. A mensagem enviada segue o **mesmo formato**, diferenciando-se apenas pelo nome e porta atribuída ao serviço.

Imagen:

12	5.247241	127.0.0.1	127.0.0.1	TCP	56 57700 → 7777 [SYN] Seq=0
13	5.247290	127.0.0.1	127.0.0.1	TCP	56 7777 → 57700 [SYN, ACK] Seq=1 Ack=1 Win=65495 Len=14
14	5.247351	127.0.0.1	127.0.0.1	TCP	44 57700 → 7777 [ACK] Seq=1 Ack=1 Win=65495 Len=14

(3-way handshake do servido UDP que usa o TCP para registrar somente o seu serviço no servidor de nomes, uma vez que é necessário que não ocorra perda de informações)

> Frame 15: 67 bytes on wire (536 bits), 67 bytes captured (536 bits) on interface \Device\NPF_{...}	0000 02 00 00 00 45 00 00 3f 82 62 40 00 80 06 00 00 . . . E .. ? - b@ ...
> Null/Loopback	0010 7f 00 00 01 7f 00 00 01 e1 64 1e 61 72 68 df 09 d arh ..
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1	0020 fc 61 7f 5c 50 18 00 ff e0 a6 00 00 63 61 6c 63 . . . a \P . . . calc
> Transmission Control Protocol, Src Port: 57700, Dst Port: 7777, Seq: 1, Ack: 1, Len: 23	0030 5f 75 64 70 20 31 32 37 2e 30 2e 30 2e 31 20 39 . . . _udp 127 .0.0.1 9
▼ Data (23 bytes)	0040 39 39 39 . . . 999

(frame 15: contém o envio do serviço UDP para o servidor de nomes)

2.3 Consulta ao Servidor de Nomes

Em um momento posterior da captura, o cliente envia uma requisição ao Servidor de Nomes solicitando o endereço do serviço UDP. A **resposta contém o IP e porta** previamente registrados.

Imagen:

23 11.319664	127.0.0.1	127.0.0.1	TCP	56 57702 → 7777 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 WS=256 SACK_PERM
24 11.319749	127.0.0.1	127.0.0.1	TCP	56 7777 → 57702 [SYN, ACK] Seq=0 Ack=1 Win=65495 Len=0 MSS=65495 WS=256 SACK_PERM
25 11.319811	127.0.0.1	127.0.0.1	TCP	44 57702 → 7777 [ACK] Seq=1 Ack=1 Win=65280 Len=0
26 11.319894	127.0.0.1	127.0.0.1	TCP	52 57702 → 7777 [PSH, ACK] Seq=1 Ack=1 Win=65280 Len=8
27 11.319915	127.0.0.1	127.0.0.1	TCP	44 7777 → 57702 [ACK] Seq=1 Ack=9 Win=65280 Len=0
28 11.320210	127.0.0.1	127.0.0.1	TCP	58 7777 → 57702 [PSH, ACK] Seq=9 Ack=9 Win=65280 Len=14
29 11.320235	127.0.0.1	127.0.0.1	TCP	44 57702 → 7777 [ACK] Seq=9 Ack=15 Win=65280 Len=0
30 11.320256	127.0.0.1	127.0.0.1	TCP	44 7777 → 57702 [FIN, ACK] Seq=15 Ack=9 Win=65280 Len=0
31 11.320281	127.0.0.1	127.0.0.1	TCP	44 57702 → 7777 [FIN, ACK] Seq=9 Ack=15 Win=65280 Len=0
32 11.320308	127.0.0.1	127.0.0.1	TCP	44 7777 → 57702 [ACK] Seq=16 Ack=10 Win=65280 Len=0

(3-way handshake do cliente UDP que usa o TCP para consultar somente o serviço “calc_udp” no servidor de nomes, uma vez que é necessário que não ocorra perda de informações)

(frame 26: consulta. Frame 28: resposta do servidor de nomes com o ip e porta do servidor_udp)

> Frame 28: 58 bytes on wire (464 bits), 58 bytes captured (464 bits) on interface \Device\NPF_{...}	0000 02 00 00 00 45 00 00 36 82 6f 40 00 80 06 00 00 . . . E .. 6 o@ ...
> Null/Loopback	0010 7f 00 00 01 7f 00 00 01 e1 64 1e 61 66 16 0d 55 98 a f ..
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1	0020 29 80 ec d7 50 18 00 ff c2 b0 00 00 31 32 37 2e) . . . P . . . 127.
> Transmission Control Protocol, Src Port: 7777, Dst Port: 57702, Seq: 1, Ack: 9, Len: 14	0030 30 2e 30 2e 31 20 39 39 39 39 . . . 0.0.1 99 99
▼ Data (14 bytes)	Data: 312372e302e312039393939 [Length: 14]

(frame 30 e 31: fecha a conexão TCP)

2.4 Comunicação UDP – Cálculo Remoto

A comunicação via UDP apresenta uma série de datagramas curtos, incluindo:

- envio da mensagem inicial para iniciar o diálogo,
- envio do menu pelo servidor,
- envio da escolha da operação,
- envio dos números,
- retorno do resultado.

O tempo entre as mensagens intercambiadas é extremamente baixo, característica natural do protocolo UDP por não exigir handshake.

Imagens da captura:

[Pacote inicial UDP (“start”) — localizar o **frame 33**]

```
> Frame 33: 37 bytes on wire (296 bits), 37 bytes captured (296 bits) on interface \Device\NPF_
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> User Datagram Protocol, Src Port: 51359, Dst Port: 9999
└ Data (5 bytes)
    Data: 7374617274
    [Length: 5]
```

0000 02 00 00 00 45 00 00 21 82 74 00 00 80 11 00 00	...E..! t.....
0010 7f 00 00 01 7f 00 00 01 c8 9f 00 27 0f 00 0d c9 3b;
0020 73 74 61 72 74	start

[Pacote contendo o menu UDP — localizar o **frame 34** (payload maior)]

```
> Frame 34: 233 bytes on wire (1864 bits), 233 bytes captured (1864 bits) on interface \Device\NPF_
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> User Datagram Protocol, Src Port: 9999, Dst Port: 51359
└ Data (201 bytes)
    Data [...]: 0a1b5b39366d3d3d2043616c63756c61646f72612052656d6f7461203d3d3d1b5b306d0a457363
    [Length: 201]
```

0000 02 00 00 00 45 00 00 e5 82 75 00 00 80 11 00 00	...E... u.....
0010 7f 00 00 01 7f 00 00 01 27 0f c8 9f 00 d1 21 f5
0020 01 b1 5b 39 36 6d 3d 3d 3d 20 43 61 6c 63 75 6c	..[93m== = Calcul
0030 61 64 6f 72 61 20 52 65 6d 6f 74 61 20 3d 3d 3d	adore Re mota ===
0040 1b 5b 30 6d 0a 45 73 63 6f 6c 68 61 20 61 20 6f	.[0m-Esc olha a o
0050 70 65 72 61 c3 a7 c3 a3 6f 28 6d 61 74 65 6d c3	pera.... o matem-
0060 a1 74 69 63 61 3a 0a 1b 5b 39 33 6d 31 2e 1b 5b	tica... [93m1.[
0070 30 6d 20 53 6f 6d 61 20 28 2b 29 0a 1b 5b 39 33	0m Soma (+)...[93
0080 6d 32 2e 1b 5b 30 6d 20 53 75 62 74 72 61 c3 a7	m2..[0m Subtra..
0090 c3 a3 6f 28 28 2d 29 0a 1b 5b 39 33 6d 33 2e 1b	..o (-)...[93m3..
00a0 5b 30 6d 20 4d 75 6c 74 69 70 6c 69 63 61 c3 a7	[0m Mult iplica..
00b0 c3 a3 6f 28 28 2a 29 0a 1b 5b 39 33 6d 34 2e 1b	..o (*)...[93m4..
00c0 5b 30 6d 20 44 69 76 69 73 c3 a3 6f 20 28 2f 29	[0m Divi s.o (/)
00d0 0a 1b 5b 39 33 6d 35 2e 1b 5b 30 6d 20 53 61 69	..[93m5. [0m Sai
00e0 72 20 28 45 78 69 74 29 0a	r (Exit) .

[Pacote contendo o resultado — localizar o **frame 40**]

```
> Frame 40: 36 bytes on wire (288 bits), 36 bytes captured (288 bits) on interface \Device\NPF_
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> User Datagram Protocol, Src Port: 9999, Dst Port: 51359
  Data (4 bytes)
    Data: 33352e30
    [Length: 4]
```

Frame 40: 36 bytes on wire (288 bits), 36 bytes captured (288 bits) on interface \Device\NPF_
Null/Loopback
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
User Datagram Protocol, Src Port: 9999, Dst Port: 51359
 Data (4 bytes)
 Data: 33352e30
 [Length: 4]

2.5 Comunicação TCP – Cálculo Remoto

A comunicação TCP inclui:

- o handshake de três fases (SYN, SYN-ACK, ACK),
- envio do menu pelo servidor,
- envio dos números,
- retorno do resultado.

O tempo entre o início da conexão e o envio do menu é visivelmente maior que na versão UDP, em tese, devido ao custo do handshake, porém ao ver na prática a **conexão TCP foi mais rápida**.

Imagens:

[Pacote SYN para o serviço TCP — localizar o **frame 54**]

```
> Frame 54: 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface \Device\NPF_
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 57705, Dst Port: 6666, Seq: 0, Len: 0
```

Frame 54: 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface \Device\NPF_
Null/Loopback
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 57705, Dst Port: 6666, Seq: 0, Len: 0

[Pacote contendo o menu TCP — localizar o **frame 57**]

```
> Frame 57: 245 bytes on wire (1960 bits), 245 bytes captured (1960 bits) on interface \Device\NPF_
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 6666, Dst Port: 57705, Seq: 1, Ack: 1, Len: 201
  Data (201 bytes)
    Data [...] 0a1b5b39366d3d3d2043616c63756c61646f72612052656d6f7461203d3d1b5b306d0a457363
    [Length: 201]
```

Frame 57: 245 bytes on wire (1960 bits), 245 bytes captured (1960 bits) on interface \Device\NPF_
Null/Loopback
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 6666, Dst Port: 57705, Seq: 1, Ack: 1, Len: 201
 Data (201 bytes)
 Data [...] 0a1b5b39366d3d3d2043616c63756c61646f72612052656d6f7461203d3d1b5b306d0a457363
 [Length: 201]

0000 02 00 00 00 45 00 00 t1 82 8c 40 00 80 00 00 00E...@.....
0010 7f 00 00 01 7f 00 00 01 1a 0a e1 69 2b 61 a4 edI.....E
0020 9b a1 98 46 50 18 00 ff c1 b1 00 00 0a 1b 5b 39FP....-[9
0030 36 6d 3d 3d 20 43 61 6c 63 75 6c 61 64 6f 72 6m== Ca lculador
0040 61 20 52 65 6d 6f 74 61 20 3d 3d 3d 1b 5b 30 6d a Remota ===-[0m
0050 0a 45 73 63 6f 6c 68 61 20 61 20 6f 70 65 72 61 Escolha a opera
0060 c3 a7 c3 a3 f2 20 6d 61 74 65 6d c3 a1 74 69 63 ...-o ma tem..tic
0070 61 3a 0a 1b 5b 39 33 6d 31 2e 1b 5b 30 6d 20 53 a-[93m 1.-[0m S
0080 6f 6d 61 20 28 2b 29 0a 1b 5b 39 33 6d 32 2e 1b oma (+).-[93m2.
0090 5b 30 6d 20 53 75 62 74 72 61 c3 a7 c3 a3 f2 0a [0m Subt ra...o
00a0 28 2d 29 0a 1b 5b 39 33 6d 33 2e 1b 5b 30 6d 20 (-)-[93 m3.-[0m
00b0 4d 75 6c 74 69 70 6c 69 63 61 c3 a7 c3 a3 f2 0b Multiplic ca...o
00c0 28 2a 29 0a 1b 5b 39 33 6d 34 2e 1b 5b 30 6d 20 (*)-[93 m4.-[0m
00d0 44 69 76 69 73 c3 a3 f2 20 28 2f 29 0a 1b 5b 39 Divis.o (/)-[9
00e0 33 6d 35 2e 1b 5b 30 6d 20 53 61 69 72 20 28 45 3m5.-[0m Sair (E
00f0 78 69 74 29 0a xit)-

[Pacote contendo o resultado — localizar o **frame 69**]

```
> Frame 69: 48 bytes on wire (384 bits), 48 bytes captured (384 bits) on interface \Device\NPF_
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 6666, Dst Port: 57705, Seq: 271, Ack: 6, Len: 4
  Data (4 bytes)
    Data: 33352e30
      [Length: 4]

0000  02 00 00 00 45 00 00 2c 82 98 40 00 80 06 00 00  ....E.., .@.....
0010  7f 00 00 01 7f 00 00 01 1a 0a e1 69 2b 61 a5 fb  .....i+a...
0020  9b a1 98 4b 50 18 00 ff 4e a4 00 00 33 35 2e 30  ..KP...N..35.0
```

3. Comparação dos Tempos TCP × UDP

Com base nas medições observadas na captura, é possível identificar diferenças claras no tempo de resposta entre os dois protocolos:

3.1 Desempenho do UDP

Os datagramas UDP apresentam tempos de resposta extremamente baixos. A ausência do handshake permite que o servidor responda imediatamente após o recebimento do pacote inicial. Em todas as interações observadas, o tempo entre pedido e resposta permaneceu na ordem de microssegundos. Todavia o **TCP foi mais rápido** do que o UDP nesse cenário.

3.2 Desempenho do TCP

A comunicação TCP apresenta tempos mais altos, principalmente devido à fase de estabelecimento da conexão. Mesmo após o handshake, o TCP continua exigindo confirmações de recebimento, o que aumenta ligeiramente o tempo total de cada operação.

3.3 Conclusão da Comparação

De forma geral, em teoria:

- UDP: mais rápido, menor latência, ideal para interações rápidas.
- TCP: mais confiável, porém ligeiramente mais lento.

Essa diferença foi confirmada na análise temporal da captura.

ANÁLISE DE DESEMPENHO

Durante os testes realizados, observou-se que a operação via **TCP apresentou tempo total menor** (0,000345 segundos) em comparação ao UDP (0,000542 segundos). Embora o **UDP seja teoricamente mais rápido** por não exigir handshake nem confirmação de entrega, no experimento realizado o TCP mostrou melhor desempenho. Isso ocorre porque, em **operações muito pequenas**, como a soma de dois números, o TCP se beneficia do gerenciamento de buffers do sistema operacional, que permite agrupar pacotes e **enviar múltiplos dados de uma só vez**.

No UDP, cada pacote é tratado de **forma independente**, exigindo várias chamadas de envio e recebimento, o que **aumenta o overhead proporcionalmente**. Além disso, em redes locais, onde a latência de transmissão é mínima, o tempo gasto gerenciando pacotes e chamadas de sistema passa a dominar a operação, **favorecendo o TCP**. Apesar desse resultado, o UDP continua sendo vantajoso em aplicações que enviam grandes volumes de dados sem necessidade de confiabilidade, enquanto o TCP garante entrega confiável e ordenada, sendo ideal para aplicações que necessitam consistência nos dados.

The screenshot shows two terminal windows side-by-side, both running on Windows (PS prompt). The left window is titled "Conectando ao serviço calc_udp em 127.0.0.1:9999..." and the right window is titled "Conectando ao serviço calc_tcp em 127.0.0.1:6666...". Both windows show the same sequence of commands and output, demonstrating the execution of a UDP and TCP client respectively.

```
Conectando ao serviço calc_udp em 127.0.0.1:9999...
Iniciando operação automática (20 + 15)...
Resultado recebido: 35.0
Tempo total da operação: 0.000503 segundos
PS C:\Users\rsm7\Downloads\projeto_sockets>

● Conectando ao serviço calc_tcp em 127.0.0.1:6666...
Iniciando operação automática (20 + 15)...
Resultado recebido: 35.0
Tempo total da operação: 0.000349 segundos
○ PS C:\Users\rsm7\Downloads\projeto_sockets>
```

CONSIDERAÇÕES FINAIS

O projeto permitiu a implementação prática de sistemas **cliente-servidor** utilizando os protocolos TCP e UDP, com registro e descoberta de serviços por meio de um **Servidor de Nomes**. Foi possível compreender a diferença entre os protocolos de transporte, medir o desempenho de cada um e validar o comportamento em situações de serviços inexistentes.

Além disso, a captura de pacotes via Wireshark reforçou a compreensão do tráfego de rede e da arquitetura do sistema. O trabalho proporcionou experiência prática em comunicação entre processos, programação de sockets e análise de desempenho de redes, consolidando os conceitos estudados na disciplina de **Infraestrutura de Comunicação**.