EA080 - O — Protocolos de Internet - UDP/TCP

Professor: Christian Esteve Rothenberg

Leonardo Rodrigues Marques `RA: 178610`

1 Introdução

No sexto relatório técnico de redes de computadores, comparamos o funcionamento dos protocolos de redes mais usados, TCP e UDP. O UDP é um protocolo simples da camada de transporte usado na rede para serviços de broadcast e streaming. Apesar de não ser seguro, esse protocolo é extremamente leve e eficiente, já que não há perca de tempo na abertura e fechamento de conexões com o servidor e os datagramas são bem simples. Por outro lado, há o TCP, amplamente usado. A maioria das aplicações cibernéticas, como o SSH, FTP, HTTP se assentam nesse protocolo devido sua versatilidade e robustez. Esse protocolo implementa uma série de mecanismos para manter essas características como números de reconhecimento, timeouts, retransmissões e controle de fluxo. Apesar da integralidade do TCP, várias tecnologias estão sendo desenvolvidas a fim de levar as mesmas garantias para o UDP. O QUIC, desenvolvido pelo Google, está em fase experimental e já se demonstrou bastante promissor, aliando rapidez, simplicidade e segurança.

2 Metodologia

Nesse trabalho, usamos a ferramenta netcat e wireshark para analisar os protocolos usados no transporte de dados entre dois hosts. Primeiramente, usamos o UDP para transportar um arquivo 1200 Bytes e logo após, realizamos o mesmo teste, só que usando o TCP. Todos os pacotes dos dois protocolos foram capturados pelo Wireshark e posteriormente comparados para traçar o comportamento de cada um na transmissão dos dados.

Na segunda parte do trabalho, realizamos o mesmo teste de transmissão com um arquivo de 4000 Bytes para avaliar como cada protocolo atua na fragmentação dos dados. Para UDP, a análise é extremamente simplista. Em oposição, o TCP é um protocolo bem mais cauteloso e usa uma série de números de reconhecimento e sequência para garantir a entrega do pacote na ordem apropriada.

Finalmente, deixamos o UDP de lado e exploramos as funcionalidades do TCP. Avaliamos como a conexão entre o cliente e o servidor é gerada, como funciona os números de sequência e reconhecimento dos pacotes, quais os tipos de mensagens existentes nos segmentos e como é o comportamento desse protocolo diante a situações de congestionamento.

3 Resultado, Discussões e Conclusões

3.1 Questão 1

3.1.1

Utilizando-se do Wireshark para inspeção dos pacotes, observamos que apenas um pacote foi transmitido para cada datagrama UDP.

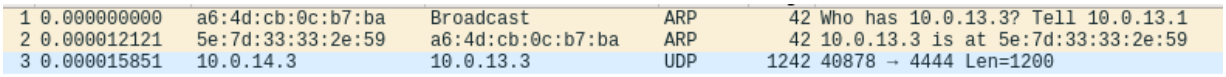


Figura 1: Captura de Pacotes UDP.

3.1.2

Nessa tarefa, o total de Bytes transferidos entre os hosts foi de 1242. Dentre 1242 Bytes, 1200 Bytes (96.62%) estão relacionados aos dados e 42 Bytes (3.38%) aos cabeçalhos dos protocolos. Na tabela abaixo, foi colocado o tamanho do cabeçalho de cada protocolo e uma porcentagem em relação ao total do pacote de 1242 Bytes.

Cabeçalho	Tamanho	Porcentagem (1242 Bytes)
Ethernet	14 Bytes	1.12%
IP	20 Bytes	1.61%
UDP	8 Bytes	0.64%

Tabela 1: Comparativo entre cabeçalhos.

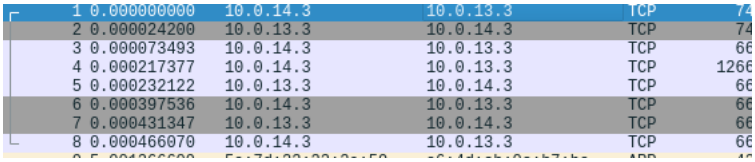
3.1.3

Após repetir o processo, apenas os campos **Destination Port: 4444**, **Lenght: 1208** e **Checksum: 0x33cf** não mudam.

3.2 Questão 2

3.2.1

Durante a transmissão, foram capturados **8** pacotes TCP. Nessa captura de pacote, foram obtidos 5 tipo de segmentos TCP. Abaixo da figura, há uma descrição de cada tipo.



1	0.000000000	10.0.14.3	10.0.13.3	TCP	74
2	0.000024200	10.0.13.3	10.0.14.3	TCP	74
3	0.000073493	10.0.14.3	10.0.13.3	TCP	66
4	0.000217377	10.0.14.3	10.0.13.3	TCP	1266
5	0.000232122	10.0.13.3	10.0.14.3	TCP	66
6	0.000397536	10.0.14.3	10.0.13.3	TCP	66
7	0.000431347	10.0.13.3	10.0.14.3	TCP	66
8	0.000466070	10.0.14.3	10.0.13.3	TCP	66

Figura 2: Captura de Pacotes TCP.

1. **SYN:** A abertura ativa é realizada por meio do envio de um SYN pelo cliente ao servidor. O cliente define o número de sequência de segmento como um valor aleatório A.

2. **SYN+ACK:** Em resposta, o servidor responde com um SYN-ACK. O número de reconhecimento (acknowledgment) é definido como sendo um a mais que o número de sequência recebido, i.e. A+1, e o número de sequência que o servidor escolhe para o pacote é outro número aleatório B.

3. **ACK:** Finalmente, o cliente envia um ACK de volta ao servidor. O número de sequência é definido ao valor de reconhecimento recebido, i.e. A+1, e o número de reconhecimento é definido como um a mais que o número de sequência recebido, i.e B+1.

4. **PSH+ACK:** Nesse caso, a transmissão de dados é iniciada e armazenada em um buffer no aplicativo receptor.

5. **FIN+ACK:** Nesse pacote, é enviado o último pacote do remetente ao destinatário sinalizando o fechamento da conexão.

3.2.2

Os pacotes TCP foram detalhados na tabela abaixo:

Pacote	Cabeçalhos	Tamanho (Bytes)
SYN (74 Bytes)	Ethernet	14
	IP	20
	TCP	40
SYN, ACK (74 Bytes)	Ethernet	14
	IP	20
	TCP	40
FIN, ACK (74 Bytes)	Ethernet	14
	IP	20
	TCP	32
PSH, ACK (1266 Bytes)	Ethernet	14
	IP	20
	TCP	32
	DATA	1200
ACK (66 Bytes)	Ethernet	14
	IP	20
	TCP	32
FIN (66 Bytes)	Ethernet	14
	IP	20
	TCP	32

Tabela 2: Caption

É possível observar que os cabeçalhos somados geram um quantidade de 420 Bytes, oque representa **26%** do total de dados transmitidos. Esse valor é extremamente alto se comparado aos **3.38%** do protocolo UDP.

3.3 Questão 3

UDP: Ao invés de enviar um pacote de dados como anteriormente, o protocolo UDP basicamente enviou ordenadamente os dados em dois pacotes. O primeiro datagrama contém 2048 Bytes de dados, enquanto o segundo datagrama contém 1952 Bytes de dados, totalizando 4000 Bytes do arquivo. Não houve verificação de recepção.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.14.3	10.0.13.3	UDP	2090	53913 → 4444 Len=2048
2	0.000392004	10.0.14.3	10.0.13.3	UDP	1994	53913 → 4444 Len=1952

Figura 3: Captura de Pacotes usando protocolo UDP.

TCP: Por outro lado, o TCP trata a questão de fragmentação de forma mais cautelosa quando comparado ao UDP. Para manter um conexão mais confiável, TCP mantém um controle do envio de mensagens para o host de destino. Esse controle é feito através da troca de mensagens ACK. Ao enviar dados de um host para o outro, o TCP envia uma mensagem de sinalização identificado com o byte a ser transmitido + 1. A transmissão é feita, e o host de destino retorna com a mensagem de confirmação ACK identificado com o último byte transmitido + 1. Na transmissão de 1200 Bytes, esse processo ocorreu apenas uma vez. Já com 4000 Bytes, houve fragmentação do pacote provavelmente devido ao MTU. Nesse caso, uma mensagem ACK com identificador de sequência 1 é enviado, o pacote é transmitido e uma congirmação ACK com identificador 2049 é recebido. Após a recepção, outro envio de dados é feito com identificador de sequência 2049 e uma mensagem de confirmação é recebida pelo host origem com identificador 4001. Isso conlcui que a transmissão dos dados foi feita com sucesso.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.14.3	10.0.13.3	TCP	74	38358 → 4444 [SYN] Seq=0 Win=29200 Len=0
2	0.000027085	10.0.13.3	10.0.14.3	TCP	74	4444 → 38358 [SYN, ACK] Seq=0 Ack=1 Win=2
3	0.000070780	10.0.14.3	10.0.13.3	TCP	66	38358 → 4444 [ACK] Seq=1 Ack=1 Win=29696
4	0.000146096	10.0.14.3	10.0.13.3	TCP	2114	38358 → 4444 [PSH, ACK] Seq=1 Ack=1 Win=2
5	0.000155495	10.0.13.3	10.0.14.3	TCP	66	4444 → 38358 [ACK] Seq=1 Ack=2049 Win=332
6	0.000698461	10.0.14.3	10.0.13.3	TCP	2018	38358 → 4444 [PSH, ACK] Seq=2049 Ack=1 Wi
7	0.000710983	10.0.13.3	10.0.14.3	TCP	66	4444 → 38358 [ACK] Seq=1 Ack=4001 Win=373
8	0.002464551	10.0.14.3	10.0.13.3	TCP	66	38358 → 4444 [FIN, ACK] Seq=4001 Ack=1 Wi
9	0.002542250	10.0.13.3	10.0.14.3	TCP	66	4444 → 38358 [FIN, ACK] Seq=1 Ack=4002 Wi
10	0.002611339	10.0.14.3	10.0.13.3	TCP	66	38358 → 4444 [ACK] Seq=4002 Ack=2 Win=296
11	5.015221479	06:17:08:56:54:0a	1a:46:a2:dd:06:28	ARP	42	Who has 10.0.13.1? Tell 10.0.13.3
12	5.015214879	1a:46:a2:dd:06:28	06:17:08:56:54:0a	ARP	42	Who has 10.0.13.3? Tell 10.0.13.1
13	5.015295314	06:17:08:56:54:0a	1a:46:a2:dd:06:28	ARP	42	10.0.13.3 is at 06:17:08:56:54:0a
14	5.015300706	1a:46:a2:dd:06:28	06:17:08:56:54:0a	ARP	42	10.0.13.1 is at 1a:46:a2:dd:06:28

Figura 4: Captura de Pacotes usando potocolo TCP.

3.4 Questão 4

3.4.1

No processo de handshake, foram transmitidos três pacotes **(3 4 5)**. Cada pacote contém um tipo de segmento diferente. No primeiro (3), a flag marcada é **SYN**. Isso significa que essa é uma mensagem de saudação para abrir uma nova conexão com o host de destino. No segundo (4), a flags marcadas são **SYN** e **Acknowledgment**. Essa mensagem é a confirmação para o cliente de abertura de conexão pelo servidor e reconhecimento de número de sequência. Por fim, no terceiro pacote (5), a flag marcada é **ACK**. Essa é uma mensagem de confirmação, onde há um número de reconhecimento que foi recebido do servidor.

1	0.000000000	f6:3c:4e:a8:92:7b	Broadcast	ARP	42 Who has 10.0.13.3? Tell 10.0.13.1
2	0.000013320	3a:07:84:bb:08:15	f6:3c:4e:a8:92:7b	ARP	42 10.0.13.3 is at 3a:07:84:bb:08:15
3	0.000016923	10.0.14.3	10.0.13.3	TCP	74 60554 → 4444 [SYN] Seq=0 Win=29200 L
4	0.000039855	10.0.13.3	10.0.14.3	TCP	74 4444 → 60554 [SYN, ACK] Seq=0 Ack=1
5	0.000082411	10.0.14.3	10.0.13.3	TCP	66 60554 → 4444 [ACK] Seq=1 Ack=1 Win=2
6	5.008525684	3a:07:84:bb:08:15	f6:3c:4e:a8:92:7b	ARP	42 Who has 10.0.13.1? Tell 10.0.13.3
7	5.008598920	f6:3c:4e:a8:92:7b	3a:07:84:bb:08:15	ARP	42 10.0.13.1 is at f6:3c:4e:a8:92:7b

Figura 5: Handshake do Protocolo TCP.

3.4.2

Os números de sequência trocados são **1295dcd1**, **eb839e37** e **1295dc1e**.

Figura 6: Números de Sequência.

3.4.3

O número de sequência do primeiro byte do dado da aplicação é **1295dc1e**.

Figura 7: Número de Sequência do Primeiro Byte.

3.4.4

Início: O tamanho máximo da janela definido pelo cliente é 29200 Bytes. Já para o servidor, esse valor é de 28960 Bytes.

Figura 8: Valores dos tamanhos da janela do Servidor e Cliente.

Após transmissão de dados: O tamanho máximo da janela definido pelo cliente é 58 Bytes. Já para o servidor, esse valor é de 57 Bytes.

Figura 9: Valores dos tamanhos da janela do Servidor e Cliente.

3.4.5

O valor de MSS definido pelo cliente e servidor é **1460** Bytes.

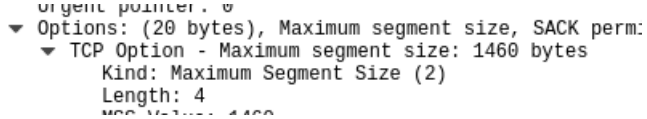


Figura 10: Valor do MSS na captura do pacote.

3.4.6

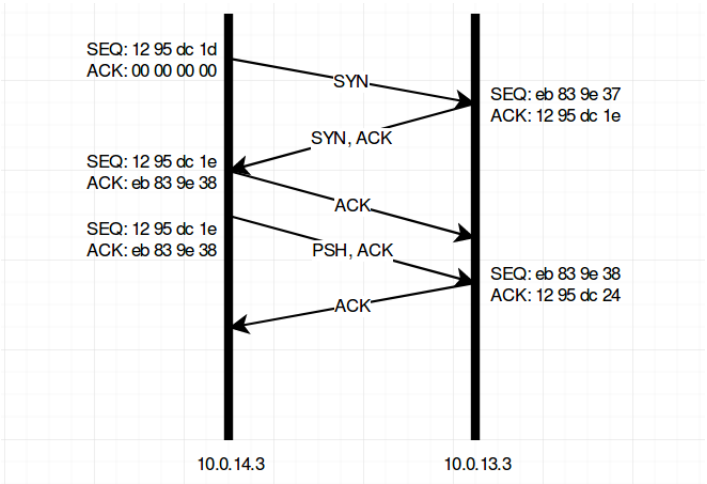


Figura 11: Troca de mensagens entre hosts h1 e h4.

3.4.7

Os pacotes envolvidos são **(8, 9 e 10)**. Cada pacote contém um tipo de segmento TCP diferente. No primeiro(8), a flags marcadas são **FIN** e **Acknowledgment**. Isso significa que essa é uma mensagem de fechamento da conexão entre o cliente e o servidor. No segundo pacote(9), as flags marcadas são as mesmas do pacote 8. Essa também é uma mensagem de fechamento de conexão. Por fim, o pacote 3, as flags marcadas são **Acknowledgment** . Essa é uma mensagem de confirmação, onde há um número de reconhecimento que foi recebido pelo servidor .

3.5 Questão 5

Nessa questão, avaliamos o comportamento do TCP quanto ao congestionamento de dados. Para isso, criamos um ambiente de simulação em que o host H3 envia 100 megaBytes de dados até o host h2 através do protocolo UDP(tráfego de fundo) simultaneamente a transmissão de segementos TCP do host h1 para o host h4. Esse processo causa um congestionamento no fluxo de dados.

No **gráfico 13 e 14**, há duas linhas: uma de fluxo de segmentos TCP e uma da estimativa da janela de recebimento do cliente. O gráfico 14 é apenas uma ampliação do gráfico 1. No **gráfico 15**, é mostrado a latência em função do tempo. E por último, o **gráfico 16** representa o fluxo de dados em função do tempo.

Como estamos em uma condição de congestionamento e o TCP trata essa condição a fim de amenizar perdas, é possível observar as estratégicas utilizadas desse protocolo. Nos tempos **13** e **24** segundos do gráfico 15, observamos que há um aumento na latência. Diante dessa circunstância, o fluxo de dados é reduzido nesses tempos. Essa redução é visível na queda acentuada da linha de fluxo do gráfico 16. Ao mesmo tempo, há um alongamento no tempo de "permanência"do **ACK** transmitido nesses tempos (gráfico 13).

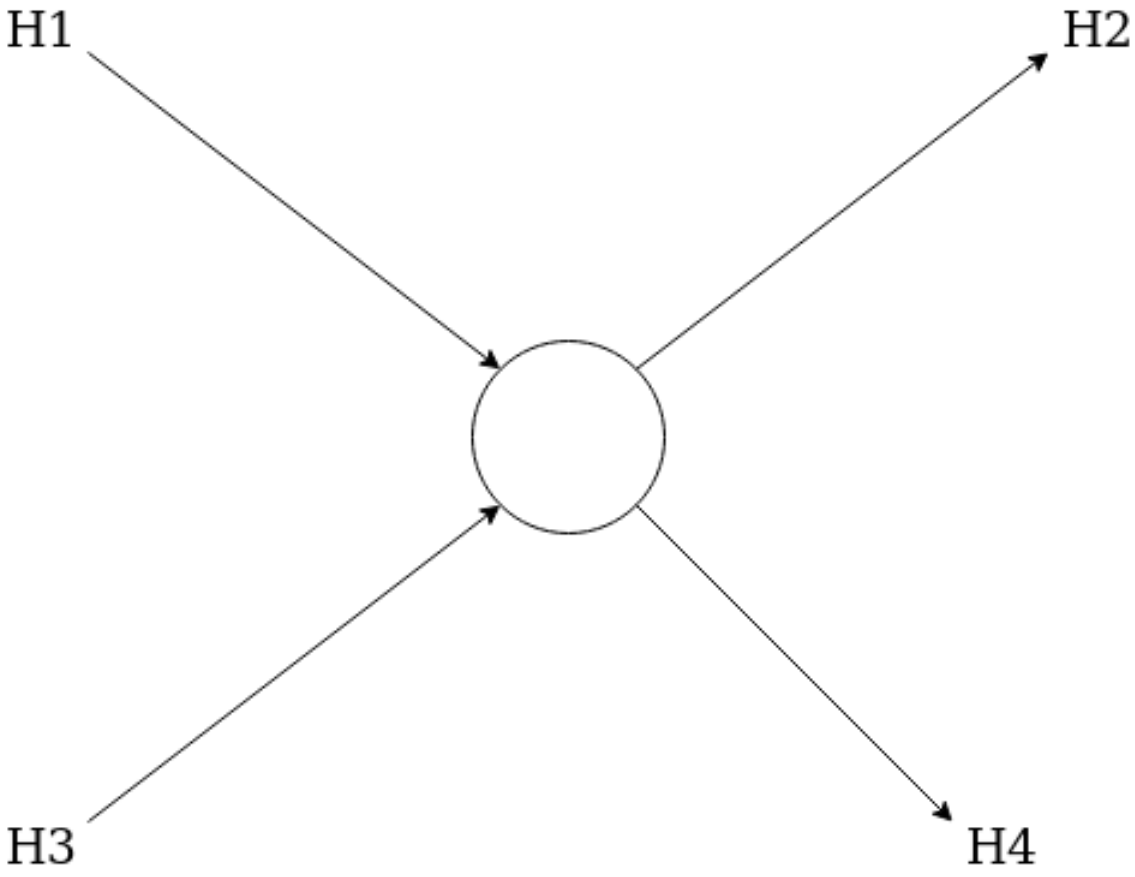


Figura 12: Fluxos do segmentos TCP de H1 a H2 e datagramass de H3 a H2.

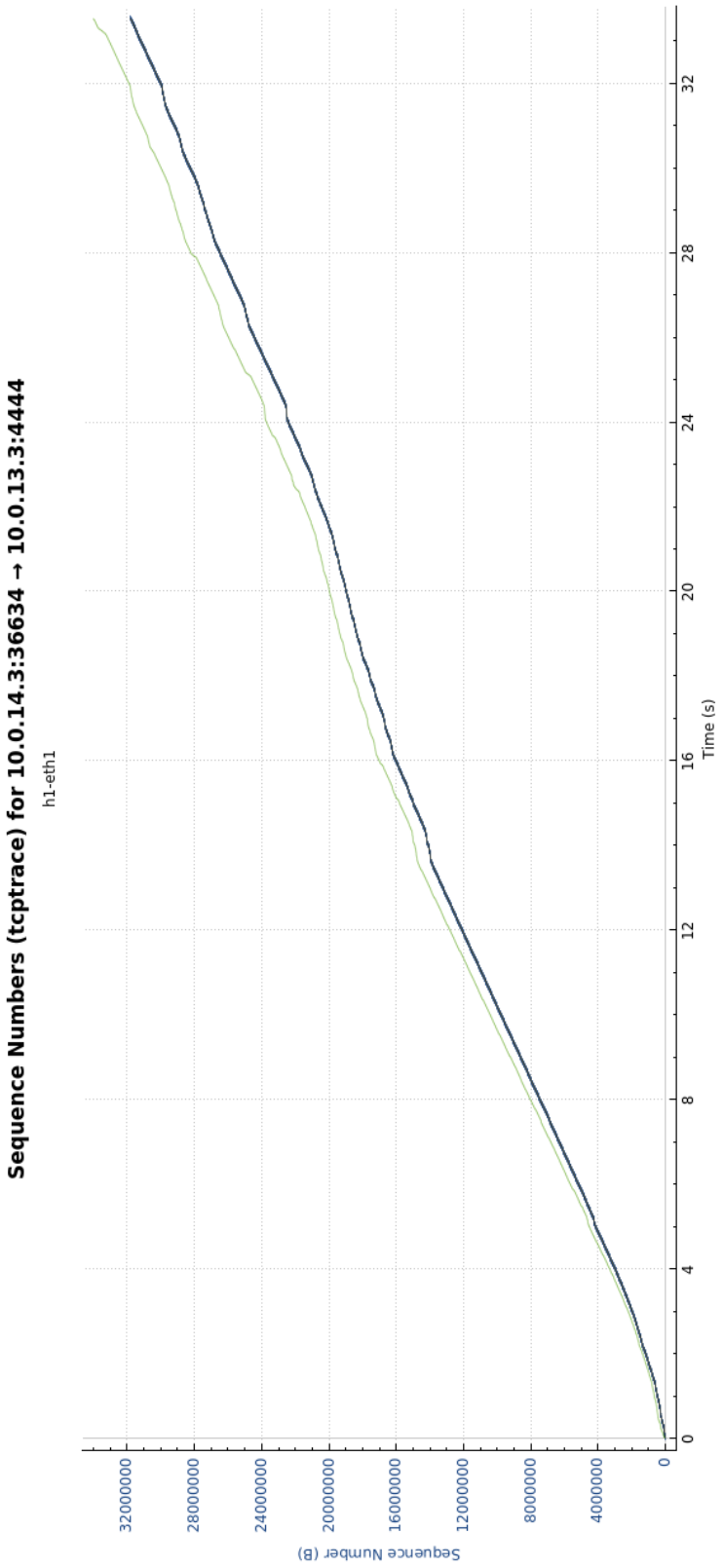


Figura 13: Gráfico Tempo Sequência (tcptrace).

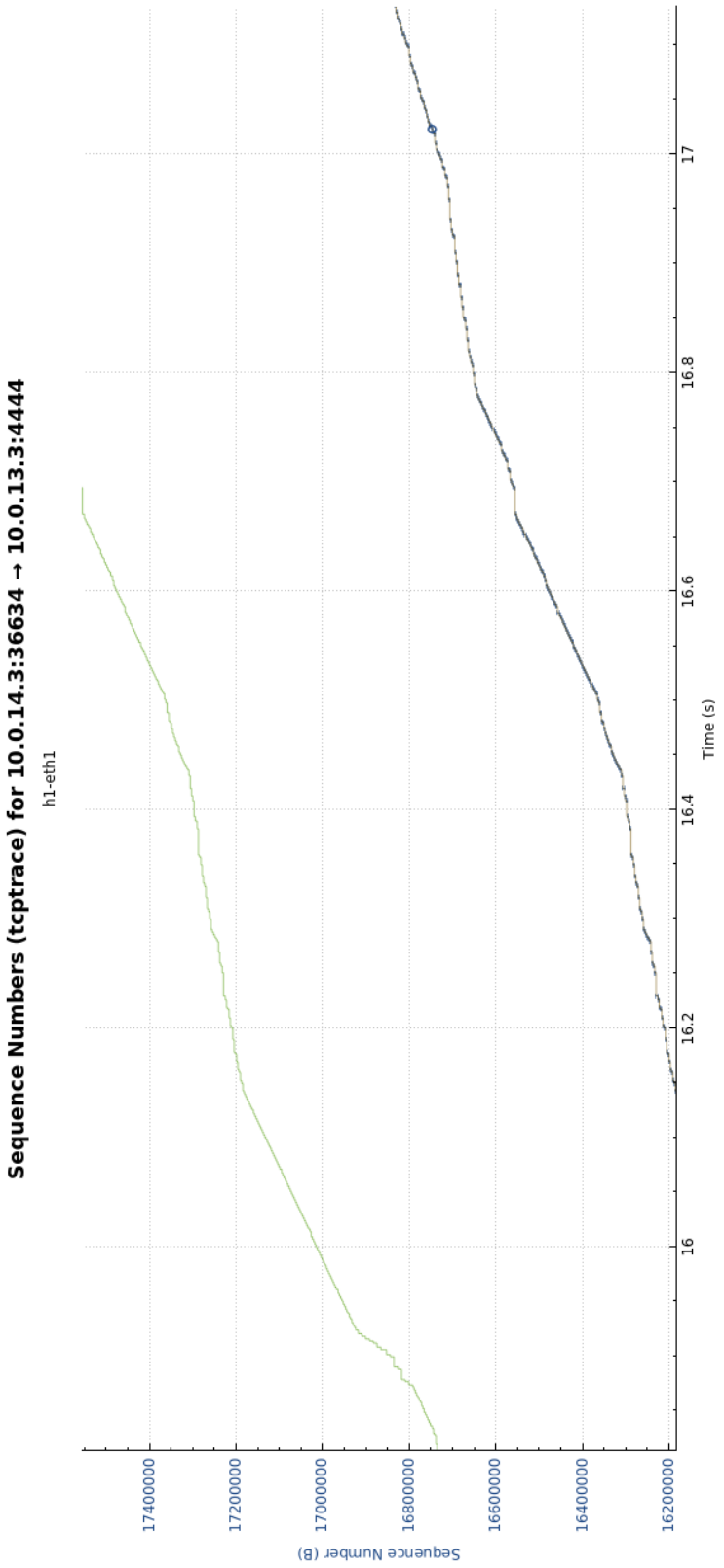


Figura 14: Gráfico Tempo Sequência (tcptrace).

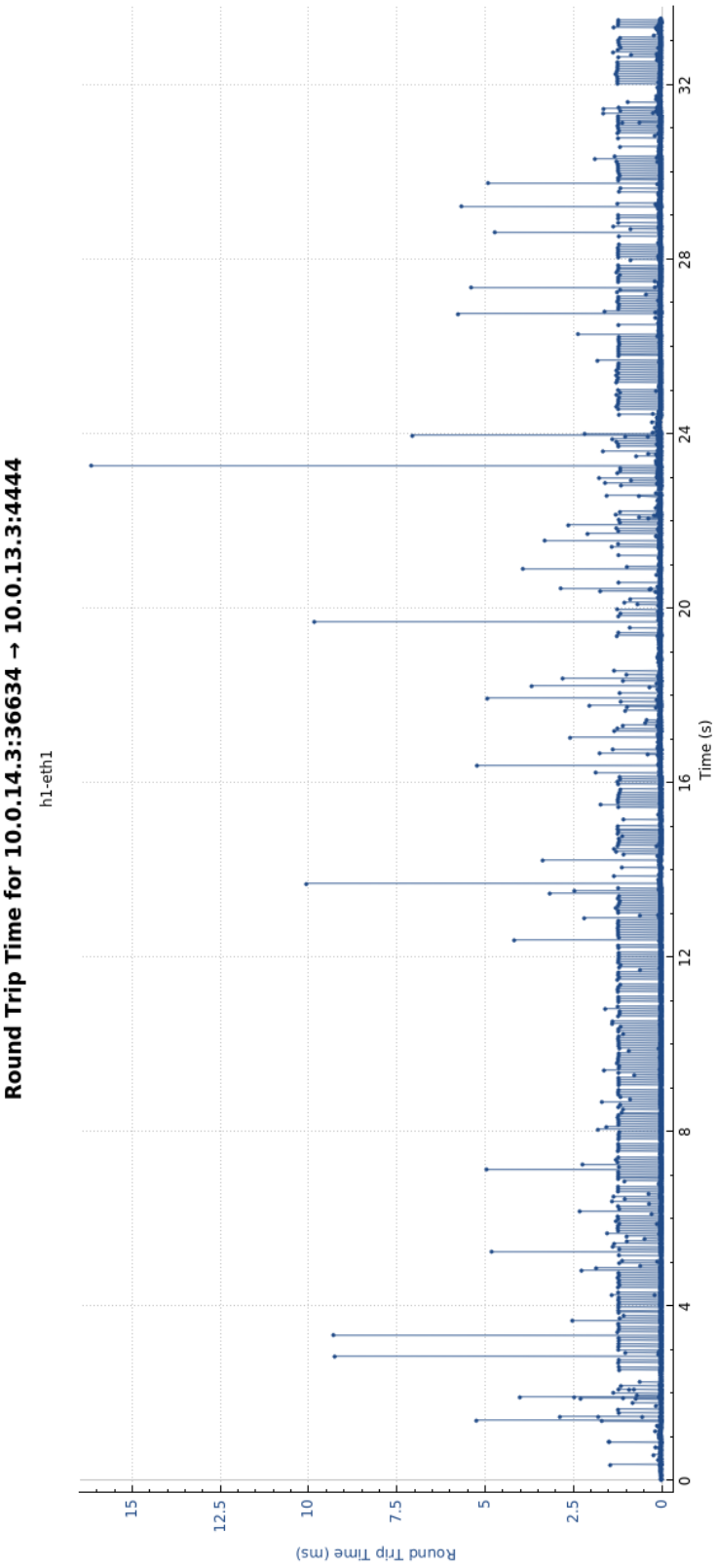


Figura 15: Gráfico Round Trip Time (RTT).

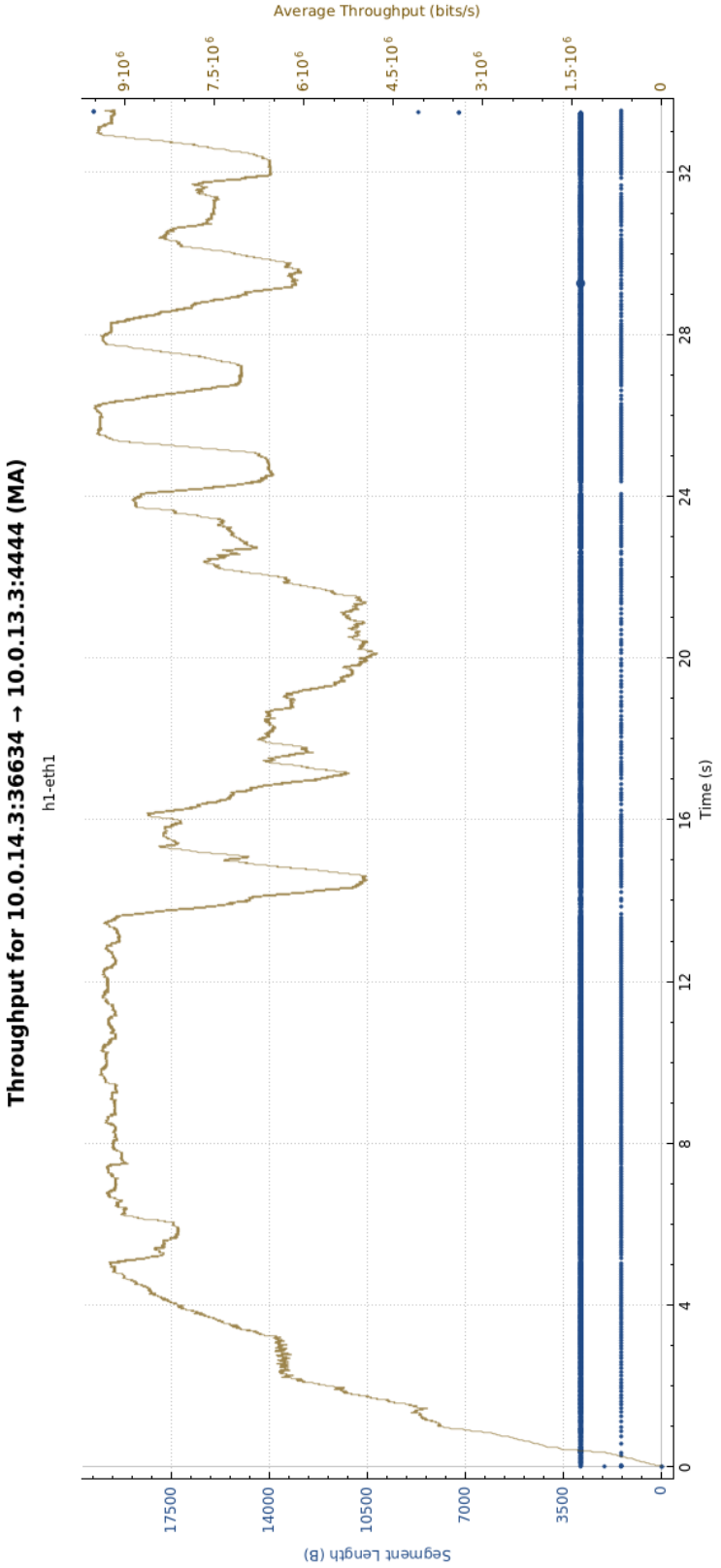


Figura 16: Gráfico Vazão de Dados.

4 Referências

1. https://en.wikipedia.org/wiki/Transmission_Control_Protocol

2. https://bit.ly/2JIJhPy

3. https://bit.ly/2JPRWQ8

4. https://en.wikipedia.org/wiki/Retransmission_(data_networks)