FUNDAÇÃO UNIVERSIDADE FEDERAL DO VALE DO SÃO FRANCISCO
CAMPUS JUAZEIRO
CURSO DE ENGENHARIA DA COMPUTAÇÃO

Breno Gabriel de Souza Coelho

Prática Wireshark e TCP

Juazeiro, BA
2022

Breno Gabriel de Souza Coelho

Prática Wireshark e TCP

Relatório requerido pelo professor Fábio Nelson de Sousa Pereira, como parte das exigências de avaliação da disciplina Redes de Computadores I, do curso de Engenharia da Computação, da Universidade do Vale do São Francisco.

Juazeiro, BA
2022

SUMÁRIO

1 INTRODUÇÃO

TCP (Transport Control Protocol - Protocolo de Controle do Transporte), como citado e discutido em LEUNG, KA-CHEONG et al (2007), é o protocolo de camada de transporte mais utilizado hoje em dia. Seus mecanismos de controle de fluxo, congestionamento e entrega confiável de pacotes constituem as principais razões pelas quais esse protocolo conseguiu tanto apelo. Assim como o UDP, sua estrutura base possui lacunas, que podem ser desenvolvidas de diferentes maneiras dependendo da implementação desejada, o que inclui, por exemplo, o que deve ser feito com pacotes fora de ordem, cálculo da temporização, outros mecanismos de controle de congestionamento e fluxo, etc.

O artigo citado anteriormente se debruça principalmente sobre os diferentes algoritmos que podem ser utilizados para lidar com a chegada de pacotes em ordem inadequada, assim como um debate sobre eficiência, prós e contras de cada caso. Propostas de algoritmo como essas são recorrentes, e geralmente tentam aperfeiçoar partes do funcionamento do TCP que podem deixar a desejar em alguns contextos.

No mesmo sentido, CHO, HYUN C. et al (2005) trata de uma rede neural capaz de gerar uma resposta mais realista ao problema de congestionamento do que o adotado até a época. Na sua visão, a natureza não linear e escolasticamente variável no tempo tornam os esquemas AQM (Active queue management) de regulamentação baseadas num nível de referência, inapropriadas. O modelo neural proposto regula o tamanho da fila de envio de acordo com um algoritmo de propagação reversa (tradução livre; do inglês, back propagation - BP), e segundo os testes feitos pelo grupo se provaram melhores no sentido de tempos de resposta, taxas de propagação e uso de banda em comparação com os esquemas “random early detection (RED)” e “proportional-integral (PI)-based AQM”.

Numa visão diferente, BAKRE, AJAY e BADRINATH, B. R. (1995) discutem como a não adequação do protocolo TCP para sistemas móveis dificulta consideravelmente a comunicação entre esses e sistemas fixos. Uma má utilização das redes de caminho seria a razão por trás desse fenômeno, afinal elas são mais voláteis do que seus contrapontos físicos. Com isso eles propõe o I-TCP, que nas suas palavras é “um protocolo de transporte indireto para hosts móveis”, que utiliza do recurso de Roteadores móveis de suporte (Mobility Support Routers - MSRs) para fornecer uma comunicação entre os níveis móvel e fixo das redes. Com essa modificação, os problemas relacionados a não confiabilidade dos sistemas sem fio são arrefecidos significativamente, com seus testes mostrando uma melhora significativa na comunicação entre um sistema móvel e fixo.

Nessa mesma direção, um problema alternativo é discutido por BALAKRISHNAN, HARI e PADMANABHAN, VENKATA N.(2001) , que tratam da assimetria natural que existe em sistemas sem fio, onde muitas vezes variáveis do sistema destino e remetente possuem diferenças significativas, como no caso da largura de banda disponível, capacidade do enlace ou outros, por exemplo. Esse

tipo de problema é potencialmente prejudicial para sistemas baseados em confirmação, como o TCP. O artigo mencionado discute em detalhes essa questão, identificando os problemas chave envolvidos e apresentando algumas formasde contornar esses transtornos.

Alguns protocolos diferentes foram propostos ao longo da história, muitos deles baseados em tecnologias já existentes. No texto sobre UDP, comentei sobre duas implementações alternativas dele que visavam ser de propósito geral. Na pesquisa para TCP, me deparei com o SCTP, (Stream Control Transmission Protocol), proposto por STEWART, RANDALL e METZ, CHRISTOPHER. (2001), é voltado para operações de telefonia via web (mais especificamente, redes Voice-Overt (VoIP)), conexões nas quais é preciso haver um grande fluxo de informações constantes e o mais confiável possível pela rede. Proposto pela Internet Engeneering Task Force, foi desenvolvido com o propósito de se juntar ao TCP e UDP como um dos protocolos de transporte mais comuns, e possi um serviço orientado à conexão, confiável, de ponto a ponto e voltado para serviços via IP pela web.

2 METODOLOGIA

1. What is the IP address and TCP port number used by the client computer (source) that is transferring the alice.txt file to gaia.cs.umass.edu? To answer this question, it's probably easiest to select an HTTP message and explore the details of the TCP packet used to carry this HTTP message, using the "details of the selected packet header window" (refer to Figure 2 in the "Getting Started with Wireshark" Lab if you're uncertain about the Wireshark windows).

2. What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?

3. What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? (Note: this is the "raw" sequence number carried in the TCP segment itself; it is NOT the packet # in the "No." column in the Wireshark window. Remember there is no such thing as a "packet number" in TCP or UDP; as you know, there are sequence numbers in TCP and that's what we're after here. Also note that this is not the relative sequence number with respect to the starting sequence number of this TCP session.). What is it in this TCP segment that identifies the segment as a SYN segment? Will the TCP receiver in this session be able to use Selective Acknowledgments (allowing TCP to function a bit more like a "selective repeat" receiver, see section 3.4.5 in the text)?

4. What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN? What is it in the segment that identifies the segment as a SYNACK segment? What is the value of the Acknowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value?

5. What is the sequence number of the TCP segment containing the header of the HTTP POST command? Note that in order to find the POST message header, you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with the ASCII text "POST" within its DATA field4 ,5. How many bytes of data are contained in the payload (data) field of this TCP segment? Did all of the data in the transferred file alice.txt fit into this single segment?

6. Consider the TCP segment containing the HTTP "POST" as the first segment in the data transfer part of the TCP connection. At what time was the first segment (the one containing the HTTP POST) in the data-transfer part of the TCP connection sent? At what time was the ACK for this first data-containing segment received? What is the RTT for this first data-containing segment? What is the RTT value the second data-carrying TCP segment and its ACK? What is the EstimatedRTT value (see Section 3.5.3, in the text) after the ACK for the second data-carrying segment is received? Assume that in making this calculation after the received of the ACK for the second segment, that the initial value of EstimatedRTT is

equal to the measured RTT for the first segment, and then is computed using the EstimatedRTT equation on page 242, and a value of = 0.125.

7. What is the length (header plus payload) of each of the first four data-carrying TCP segments?6

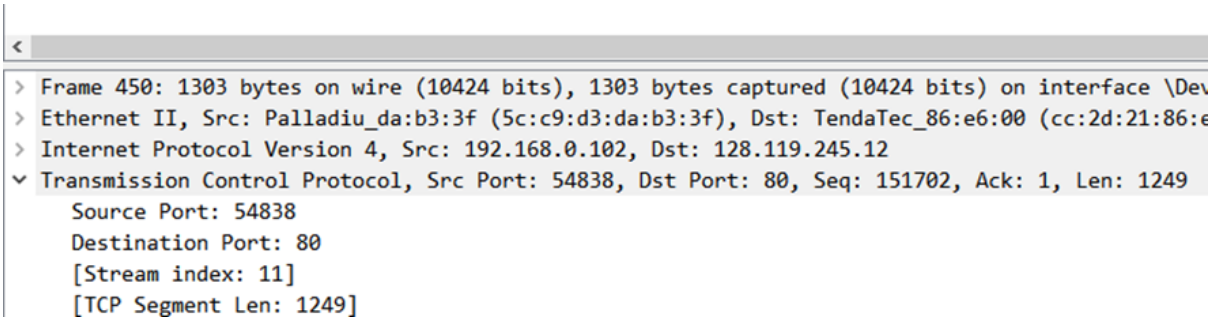
8. What is the minimum amount of available buffer space advertised to the client by gaia.cs.umass.edu among these first four data-carrying TCP segments7? Does the lack of receiver buffer space ever throttle the sender for these first four datacarrying segments?

9. Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?

10. How much data does the receiver typically acknowledge in an ACK among the first ten data-carrying segments sent from the client to gaia.cs.umass.edu? Can you identify cases where the receiver is ACKing every other received segment (see Table 3.2 in the text) among these first ten data-carrying segments?

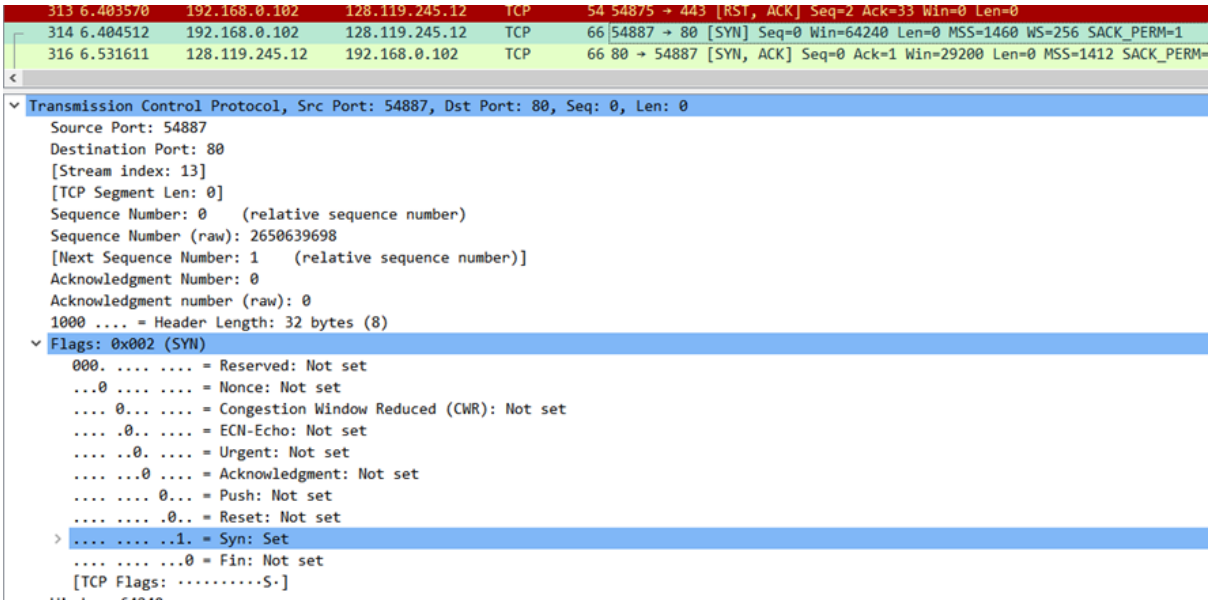
11. What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

2 DISCUSSÃO



1. Analisando o pacote HTTP POST que foi enviado, vemos que o TCP responsável por o transmitir foi enviado através da porta 54838 a partir do meu dispositivo, cujo endereço IP pode ser visto acima na linha que trata do protocolo IP, sendo 192.168.0.102 seu valor.

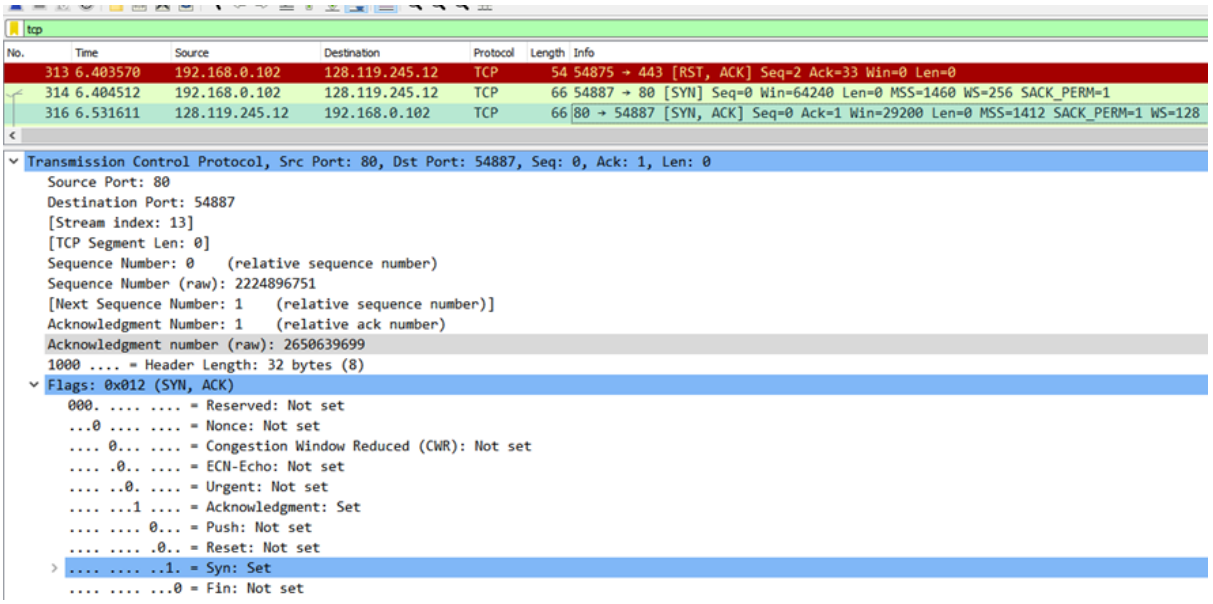
2. O endereço IP destino é 128.119.245.12 (do site gaia.cs.umass.edu). Como é uma transferência baseada em HTTP, a porta de recebimento no destinatário é a 80. E como já mencionado, os pacotes foram enviados pela porta 54838.



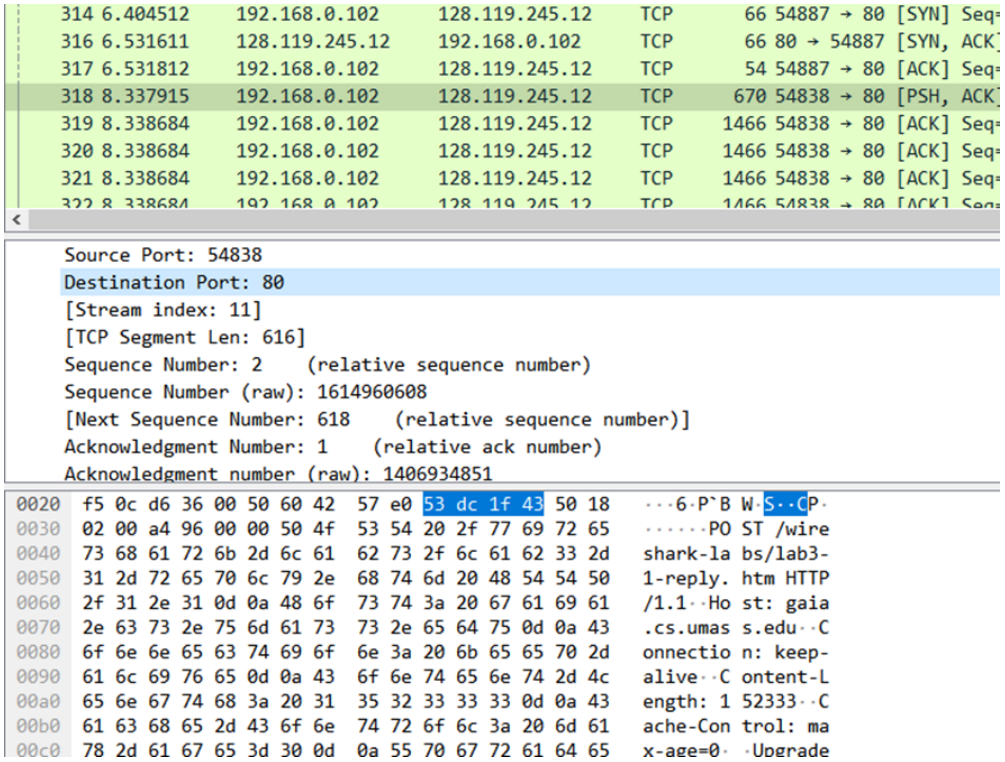
3. A imagem acima é um print do pacote TCP SYN enviado pelo meu computador. Olhando para o cabeçalho dele, vemos que o número de sequência bruto inicial é 2.650.639.698, apesar de grande, está dentro do limite esperado, que seria de até 2^32 – 1 (4 bilhões aproximadamente).

No campo “Flags” destacado, vamos que a flag SYN possui valor 1, e isso mostra que esse é um pacote SYN (adicionalmente, o próprio Wireshark dá essa dica no “nome” do pacote).

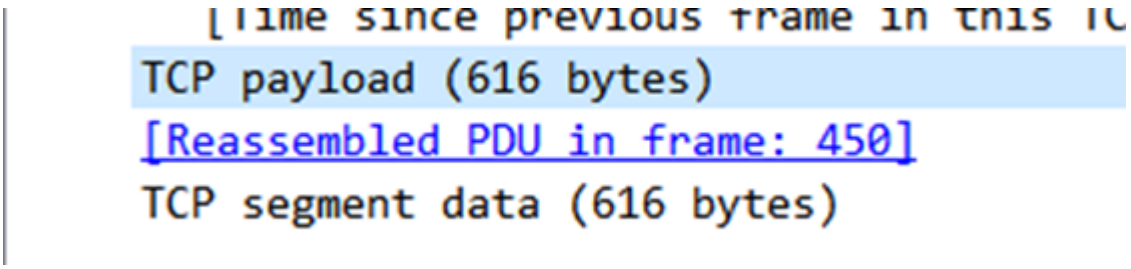
Na 5th versão, que é a que eu tenho, não existe o tópico 3.4.5.



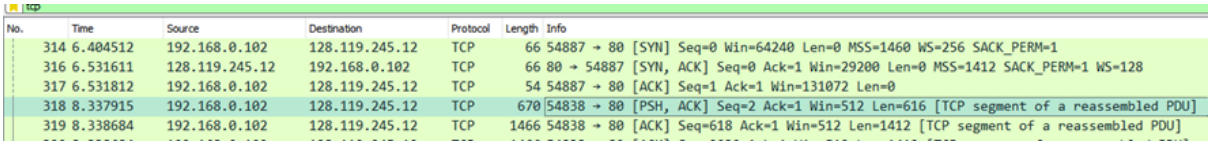
4. Analogamente, vemos que o número de sequência vale 2.224.896.751. Ambos os campos de flag Acknowledgment (reconhecimento) e SYN estão com valor 1, o que nos faz saber que esse é um pacote SYN-ACK. O campo de reconhecimento (Acknowledgment) possui valor igual ao do número de sequência do pacote SYN enviado, 2.650.639.698, como esperado. Como discutido no livro texto, esse valor foi definido a gosto do servidor gaia.cs.umass.edu, possivelmente de modo aleatório.



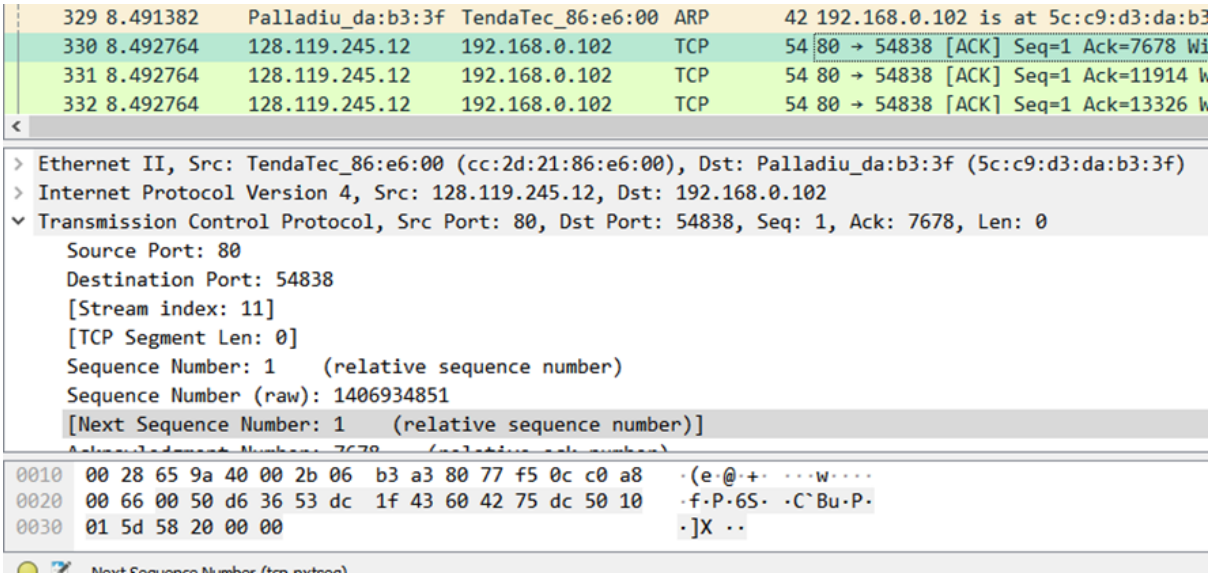
5. Esse é o segmento acima. O número de sequência vale 1.614.960.608.



Esse segmento possui 616 bytes. Isso é bem menos que o tamanho do arquivo transferido. Como esperado, ele foi quebrado em partes menores e então enviado.



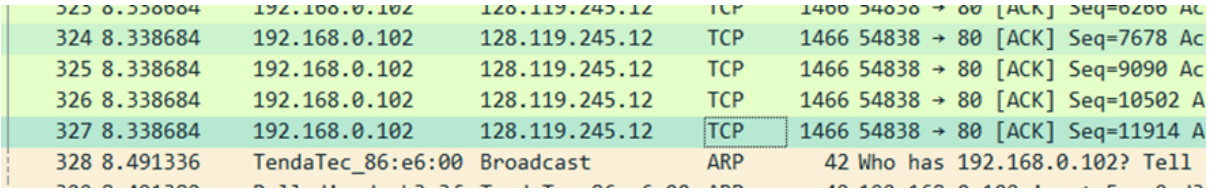
6. No tempo 8.337915 (imagino que essa seja a duração em segundos após o início da captura, já que abri e finalizei ela bem rápido).



Não foi recebido um ACK para o pacote que continha a primeira parte da mensagem, com número de sequência relativo 2, mas sim um para o de número de sequência relativo 7678. Como o TCP funciona com reconhecimentos acumulativos, esse pacote ACK reconheceu o recebimento de todos os pacotes do 2 até o 7678.

Esse ACK chegou aos 8.492764, na temporização do Wireshark. O pacote 7678 foi enviado no tempo 8.338684, então o RTT para esse pacote valeu: 0,15408 segundos, ou 154 milissegundos aproximadamente.

O segundo pacote TCP que foi confirmado foi o de número de sequência relativo 11914, enviado aos 8.338684 e seu ACK sendo recebido aos 8.492764. Veja que os números são iguais, pois foram enviados e recebidos quase ao mesmo tempo, veja:



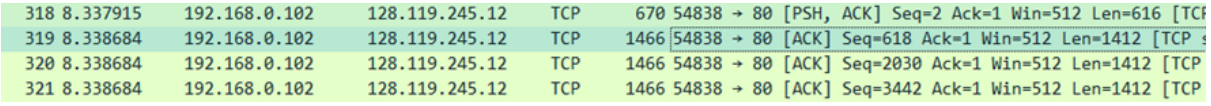
(os ACKs para ambos podem ser vistos na outra imagem logo acima)

Dessa forma o RTT para esse pacote terá o mesmo valor, 154 ms aproximadamente.

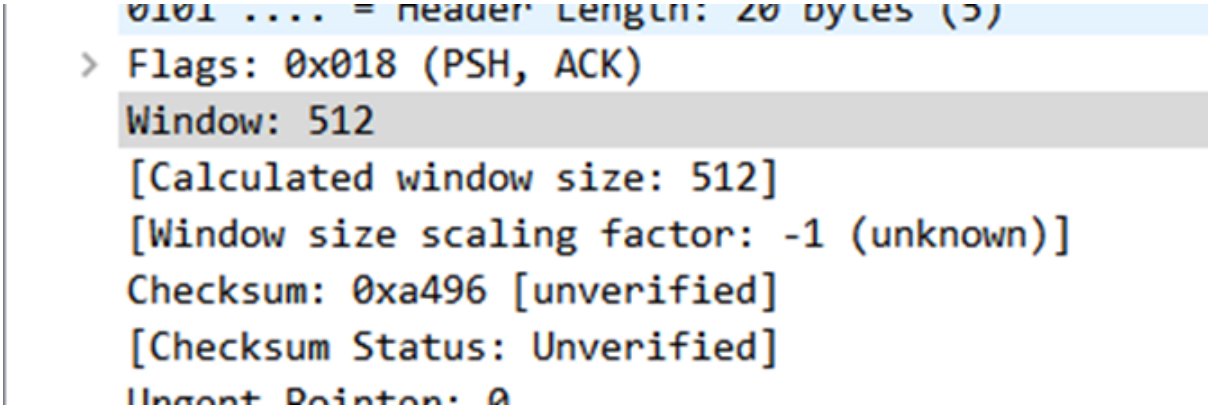
Por fim, tomando o EstimatedRTT como 0.154 inicialmente, e alfa = 0.125, seu valor subjacente, usando a equação vista, será:

(new) EstimatedRTT = 0,875 * 0,154 + 0,154 * 0,125 = 0,13475 + 0,01925 = 0,154

É intuitivamente razoável, como os valores foram iguais a “média” não mudou.



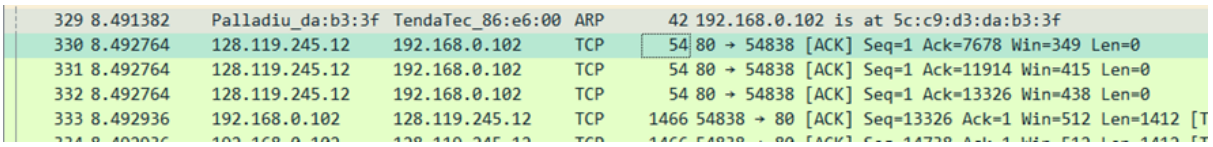
7. O primeiro foi mais leve, 616Bytes. Mas todos os demais possuem igual tamanho, 1412Bytes. Esse deve ser meu valor de MSS.



8. Analisando os quatro primeiros, e grande parte dos TCPs enviados, vi que o valor do tamanho da janela se manteve sempre igual a 512. O fator de escalamento do tamanho da janela é dado como -1, desconhecido. Por alguma razão, parece que ele não foi informado.

9. Aparentemente não. Eu filtrei todos os pacotes no Trace para visualizar somente aqueles que foram enviados pelo IP destino, ou seja, os ACKs de confirmação. Em seguida vasculhei esses ACKs, analisando se haviam dois

reconhecimentos para um mesmo pacote (sinalização de erro no recebimento). Como não havia, todos devem ter sido recebidos com sucesso. Impressionante?



329	8.491382	Palladiu_da:b3:3f	TendaTec_86:e6:00	ARP	42	192.168.0.102 is at 5c:c9:d3:da:b3:3f
330	8.492764	128.119.245.12	192.168.0.102	TCP	54	80 → 54838 [ACK] Seq=1 Ack=7678 Win=349 Len=0
331	8.492764	128.119.245.12	192.168.0.102	TCP	54	80 → 54838 [ACK] Seq=1 Ack=11914 Win=415 Len=0
332	8.492764	128.119.245.12	192.168.0.102	TCP	54	80 → 54838 [ACK] Seq=1 Ack=13326 Win=438 Len=0
333	8.492936	192.168.0.102	128.119.245.12	TCP	1466	54838 → 80 [ACK] Seq=13326 Ack=1 Win=512 Len=1412 [T

10. Os primeiros três reconhecimentos recebidos (que englobam pouco mais de 10 pacotes) foram os acima. Respectivamente, eles afirmaram o recebimento de 7678 Bytes, 4236 Bytes e 1412 Bytes. Isso dá uma média direta de 4442 Bytes reconhecidos para cada. Analisando esses valores, vemos que o último ACK é o único que reconhece somente um segmento de tamanho MSS. Todos os outros reconhecem vários de uma vez, o primeiro cerca de 7 e o segundo, 3.

11. O tempo dos primeiros 10 pacotes de 1412Bytes são iguais, de maneira que pode-se concluir que foram todos enviados juntos em um curtíssimo intervalo de tempo. A pausa provavelmente ocorreu para evitar saturar o buffer destino, ou por algum mecanismo de controle de congestionamento.

De toda forma o tempo até receber a confirmação de recebimento de algum dos pacotes foi o RTT antes calculado (que, como visto, foi igual pra todos), de 0,154 s aproximadamente.

Isso significa que um total de 10 * 1412 = 14120 bytes foram enviados num intervalo de 0,154, o que equivale a uma taxa de envio de 91688,31 Bytes/s, ou 91 KB/s, para essa operação.

3 CONCLUSÃO

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

4 REFERÊNCIA BIBLIOGRÁFICA

LEUNG, KA-CHEONG; LI, VICTOR OK; YANG, DAIQIN. **An overview of packet reordering in transmission control protocol (TCP): problems, solutions, and challenges.** IEEE transactions on parallel and distributed systems, v. 18, n. 4, p. 522-535 (2007)

CHO, HYUN C.; FADALI, M. SAMI; LEE, HYUNJEONG. **Neural network control for TCP network congestion.** In: Proceedings of the 2005, American Control Conference. p. 3480-3485 (2005)

BAKRE, AJAY; BADRINATH, B. R. **I-TCP: Indirect TCP for mobile hosts.** In: Proceedings of 15th International Conference on Distributed Computing Systems. p. 136-143. (1995)

BALAKRISHNAN, HARI; PADMANABHAN, VENKATA N. **How network asymmetry affects TCP.** IEEE Communications Magazine, v. 39, n. 4, p. 60-67, (2001)

STEWART, RANDALL; METZ, CHRISTOPHER. **SCTP: new transport protocol for TCP/IP.** IEEE Internet Computing, v. 5, n. 6, p. 64-69 (2001)