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

Breno Gabriel de Souza Coelho

Transporte UDP e prática Wireshark

Juazeiro, BA
2022

Breno Gabriel de Souza Coelho

Transporte UDP e prática Wireshark

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

UDP (User Datagram Protocol) é um protocolo que atua na camada de transporte, projetado para ser o mais simples e direto possível, fornecendo apenas os mais essenciais recursos associados a essa camada, como a capacidade de realizar Multiplexação/Demultiplexação e um controle de erro. Sua simplicidade é compensada com velocidade de transmissão e menor tamanho, porém deixa problemas em aberto (como regulagem de uso de banda e confiabilidade dos retornos) que podem se tornar um empecilho para algumas aplicações

No entanto, é possível construir aplicações que forneçam esses serviços que faltam no protocolo. Um exemplo pode ser visto no GU, YUNHONG et al (2007), onde eles detalham o seu sistema UDT, um sistema de transferência de dados baseado em UDP e feito para ser usado em WAN's de alta velocidade. Tal sistema adiciona ao protocolo mecanismos para garantir sua confiabilidade e controle de congestionamento da rede. Esse segundo acaba sendo um elemento essencial, visto que em sua ausência o protocolo usaria uma excesso da alta quantidade de banda disponível, efetivamente desperdiçando capacidade de transmissão. Com a utilização de uma série de API's para tornar a implementação mais fácil, o grupo conseguiu construir uma tecnologia que é capaz de suprir ao UDP uma série de atividades adicionais, além de se adaptar à necessidade da época de maior velocidade de transmissão nas WAN's. Inclusive a biblioteca do UDT foi expandida, permitindo que uma série de algoritmos de congestionamento diferentes pudessem ser utilizados a gosto do desenvolvedor/sistema.

Em outro trabalho, um grupo de pesquisadores construíram outra tecnologia baseada em UDP, o REUDP, voltado para permitir um rápido fluxo de dados em clusters, resolvendo problemas presentes em seu funcionamento e a necessidade de alto uso de banda em algumas situações. De modo semelhante ao trabalho anterior, eles tornaram o UDP um protocolo confiável, e também gerenciam o uso de banda de maneira mais eficiente (dai o nome, R - reliable, confiável, e E - efficient, eficiente). O trabalho mencionado, em particular, apresenta um modelo teórico do funcionamento de seu sistema e das previsões de ganhos de velocidade e eficiência de transmissão.

A maioria dos traços antes mencionados que eram adicionados ao UDP são características já presentes no protocolo TCP, o que justifica que ele tenha se tornado um dos protocolos de transporte mais comuns, No entanto, WANG, JIGANG et al (1999), discute sobre como esse protocolo é problemático em conversações curtas, tão comuns em redes sociais e afins. Publicado no final do século passado, esse artigo mostra como UDP e TCP não são opostos essenciais, mas diferentes protocolos com vantagens e desvantagens. Isso é esclarecido ao analisarmos a proposta que o artigo fez de unir ambos os métodos de transmissão para poder garantir um uso eficaz das vantagens de cada um. Observando ganhos da ordem de 20-25% com conexões persistentes que usam HTTP 1.1 e de até 40-50% com não persistentes, os pesquisadores provaram a época um modelo misto curioso.

Uma situação diferente é discutida por ECKART, BEN et al (2008), no qual fala sobre as então incipientes redes acadêmicas, voltadas para transmissão de alta quantidade de dados científicos entre universidades e instituições. Os atributos particulares dessas redes, com alta banda disponível, alta latência, e “livres” de congestionamento, fazem das especificações do protocolo TCP mais incômodas do que positivas. Dessa forma o UDP, capaz de usar a banda em sua totalidade e mais adaptativo se tornou uma escolha natural, e dessa vez sem a necessidade de implementar nele as funções mais características das transmissões TCP. O problema retratado, no entanto, é sobre as limitações que os sistemas finais possuem para receber esses dados das redes de alta performance; e o foco desse artigo é detalhar um protocolo, chamado Performance Adaptive UDP, ou PA-UDP, voltado para maximizar de modo dinâmico e automático as transmissões de acordo com as necessidades e limitações dos dispositivos. Os resultados teóricos obtidos indicam que o protocolo é útil não só em transferências nessas redes, mas em sistemas análogos onde a confiabilidade e congestionamentos não são algo importante.

Agora mencionando um trabalho recente, HERRERO, ROLANDO (2020) fala sobre a utilização de protocolos de transporte no contexto de IoT. Em sistemas desse tipo há uma distribuição limitada de potência, o que induz a ocorrência de erros e limitações das taxas de transferência. O Constrained Aplication Protocol, (CoAP) trata de adicionar uma camada capaz de lidar com esses problemas característicos, e um dos traços marcantes de seu funcionamento é a utilização múltipla do protocolo TCP e UDP. O artigo foca em discutir sobre o QUIC (Quick UDP Internet Connection), protocolo alternativo ao TCP que é mais bem adaptado ao contexto IoT e capaz de funcionar em bom equilíbrio junto do CoAP. A obra compara sua operação com a do UDP e do TCP, apresentando que ela é mais eficaz, em geral, que o UDP, porém não dispensa necessariamente o TCP, de modo que um funcionamento híbrido desses protocolos pode ser a melhor solução para esse contexto.

2 METODOLOGIA

Perguntas:

1. Select the first UDP segment in your trace. What is the packet number4 of this segment in the trace file? What type of application-layer payload or protocol message is being carried in this UDP segment? Look at the details of this packet in Wireshark. How many fields there are in the UDP header? (You shouldn't look in the textbook! Answer these questions directly from what you observe in the packet trace.) What are the names of these fields?

2. By consulting the displayed information in Wireshark's packet content field for this packet (or by consulting the textbook), what is the length (in bytes) of each of the UDP header fields?

3. The value in the Length field is the length of what? (You can consult the text for this answer). Verify your claim with your captured UDP packet.

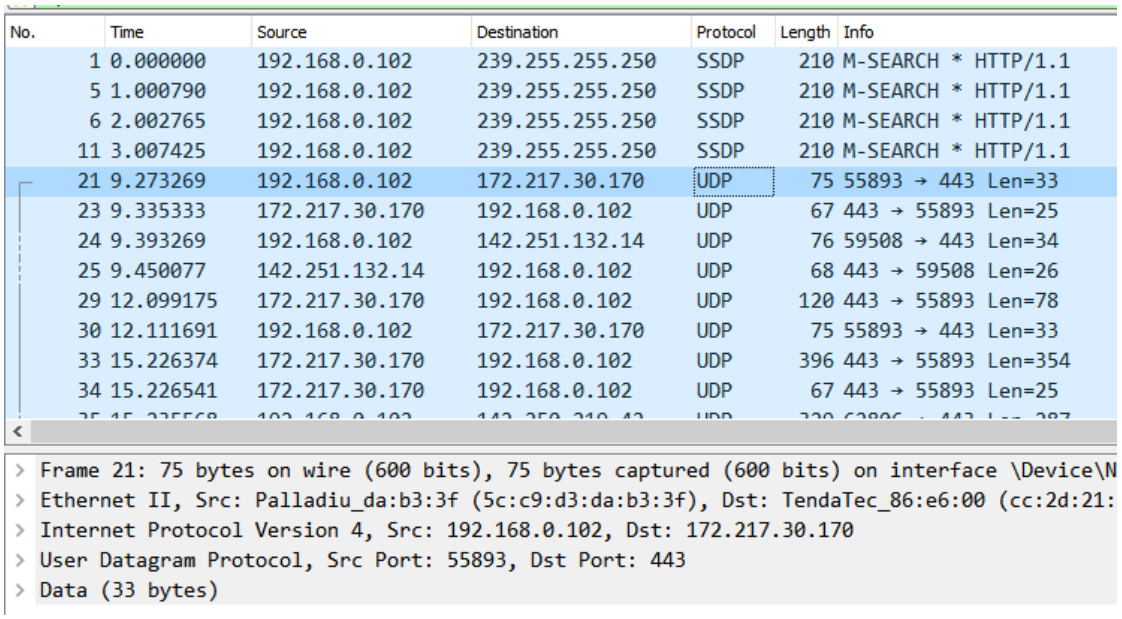
4. What is the maximum number of bytes that can be included in a UDP payload? (Hint: the answer to this question can be determined by your answer to 2. above)

5. What is the largest possible source port number? (Hint: see the hint in 4.)

6. What is the protocol number for UDP? Give your answer in decimal notation. To answer this question, you'll need to look into the Protocol field of the IP datagram containing this UDP segment (see Figure 4.13 in the text, and the discussion of IP header fields).

7. Examine the pair of UDP packets in which your host sends the first UDP packet and the second UDP packet is a reply to this first UDP packet. (Hint: for a second packet to be sent in response to a first packet, the sender of the first packet should be the destination of the second packet). What is the packet number5 of the first of these two UDP segments in the trace file? What is the packet number6 of the second of these two UDP segments in the trace file? Describe the relationship between the port numbers in the two packets.

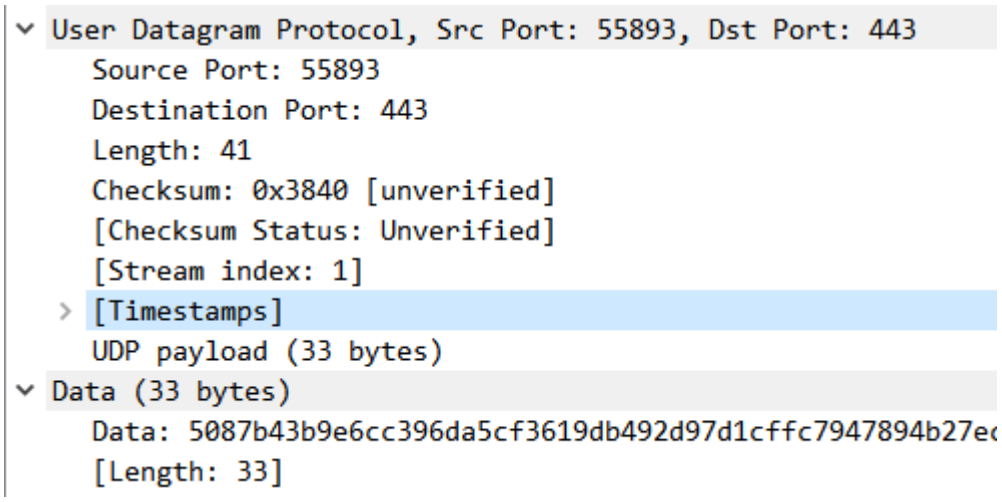
3 DISCUSSÃO



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.102	239.255.255.250	SSDP	210	M-SEARCH * HTTP/1.1
5	1.000790	192.168.0.102	239.255.255.250	SSDP	210	M-SEARCH * HTTP/1.1
6	2.002765	192.168.0.102	239.255.255.250	SSDP	210	M-SEARCH * HTTP/1.1
11	3.007425	192.168.0.102	239.255.255.250	SSDP	210	M-SEARCH * HTTP/1.1
21	9.273269	192.168.0.102	172.217.30.170	UDP	75	55893 → 443 Len=33
23	9.335333	172.217.30.170	192.168.0.102	UDP	67	443 → 55893 Len=25
24	9.393269	192.168.0.102	142.251.132.14	UDP	76	59508 → 443 Len=34
25	9.450077	142.251.132.14	192.168.0.102	UDP	68	443 → 59508 Len=26
29	12.099175	172.217.30.170	192.168.0.102	UDP	120	443 → 55893 Len=78
30	12.111691	192.168.0.102	172.217.30.170	UDP	75	55893 → 443 Len=33
33	15.226374	172.217.30.170	192.168.0.102	UDP	396	443 → 55893 Len=354
34	15.226541	172.217.30.170	192.168.0.102	UDP	67	443 → 55893 Len=25

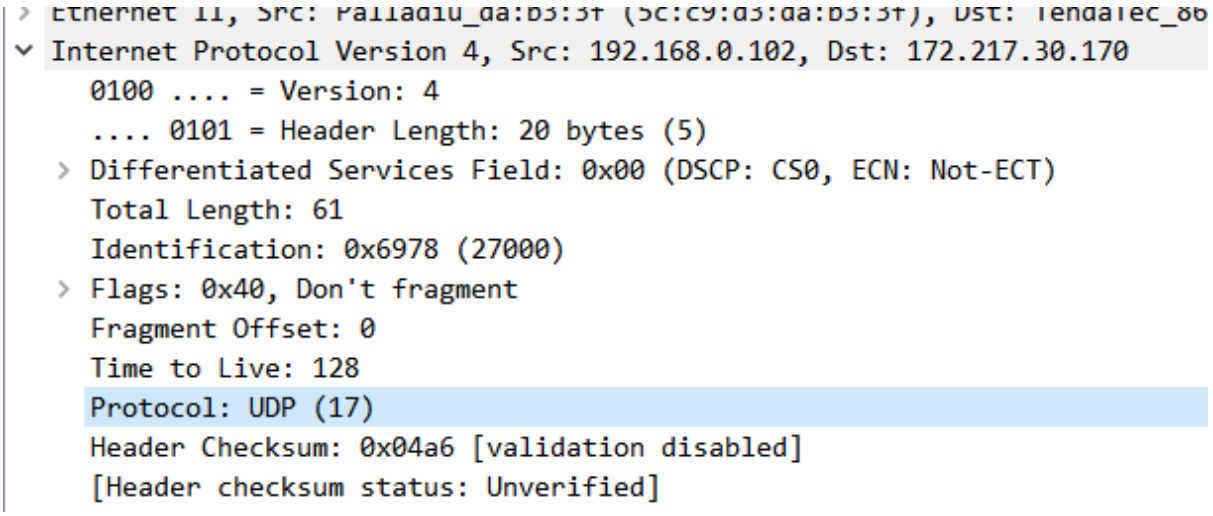
```
> Frame 21: 75 bytes on wire (600 bits), 75 bytes captured (600 bits) on interface \Device\N
> Ethernet II, Src: Palladiu_da:b3:3f (5c:c9:d3:da:b3:3f), Dst: TendaTec_86:e6:00 (cc:2d:21:
> Internet Protocol Version 4, Src: 192.168.0.102, Dst: 172.217.30.170
> User Datagram Protocol, Src Port: 55893, Dst Port: 443
> Data (33 bytes)
```

1. Ele foi o pacote de número 21. Aparentemente ele está enviando dados diretamente, olhando pela imagem pode-se ver que alguma aplicação no meu computador enviou o pacote (o meu IP é o 192.168. etc), e pelo número da porta usado (55893) podemos confirmar esse fato. Possivelmente algum software no meu sistema, meu browser por exemplo, deve ter enviado uma mensagem com algum objetivo. O dado enviado é uma sequência de caracteres: 5087b43b9e6cc396da…
   Existem 4 campos no cabeçalho da mensagem, como esperado. Interessante notar que a soma de segurança não foi confirmada, então se o envio tiver tido algum erro, não tem como saber. (imagem abaixo)
   Os campos são a Porta de Origem, a Porta de Destino, o Tamanho do arquivo e a “Checksum”, Soma de Checagem.

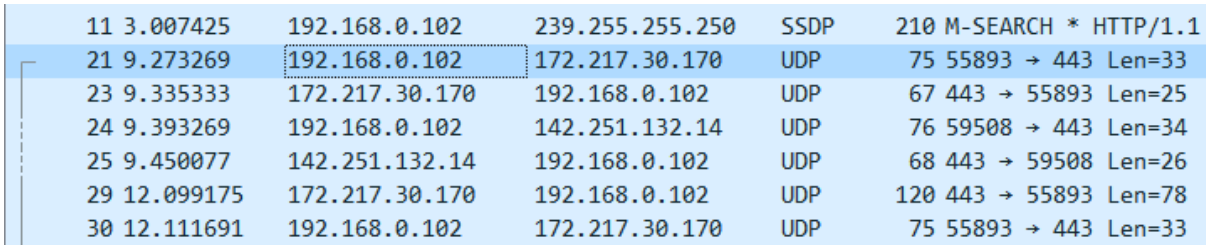


```
v User Datagram Protocol, Src Port: 55893, Dst Port: 443
    Source Port: 55893
    Destination Port: 443
    Length: 41
    Checksum: 0x3840 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 1]
  > [Timestamps]
    UDP payload (33 bytes)
v Data (33 bytes)
    Data: 5087b43b9e6cc396da5cf3619db492d97d1cffc7947894b27ec
    [Length: 33]
```

2. Pelo mencionado no livro, cada campo header ocupa 2 bytes. Mas dá para fazer algumas contas. Olhando em Data vemos que o “length” (tamanho da mensagem) é 33. Por outro lado, o campo cabeçalho diz 41. A diferença deve ser o tamanho do cabeçalho, ou seja, 8 bytes. Supondo que cada campo tem o mesmo tamanho, então de fato temos 2 bytes para cada cabeçalho.
3. É o tamanho do segmento inteiro, *incluindo* o cabeçalho, como a resposta da 2º pergunta explicou.
4. Como existe no máximo 2 bytes para o tamanho, o tamanho por sua vez é medido em bytes também, então o limite de carga vale 65.535 (total de combinações em 2by, menos um, para contar o zero) - 8 (cabeçalho), o que dá 65.527 bytes no máximo.
5. Seguindo a mesma lógica, deve valer 65.535 (porque existe a porta 0)



6. Olhando pela imagem, é o 17.



7. Aparentemente são os pacotes 21 e 23. Analisando as portas (que inclusive podem ser vistas na foto) vemos que o primeiro segmento partiu da porta 55893 e foi enviado para a porta 443. Sua resposta, como é de se esperar, saiu dessa porta de chegada (443) e foi enviada para a porta de envio (55893). Essencialmente, uma é o oposto da outra.

4 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.

5 REFERÊNCIA BIBLIOGRÁFICA

GU, YUNHONG; GROSSMAN, ROBERT L. **UDT: UDP-based data transfer for high-speed wide area networks**. **Computer Networks,** v. 51, n. 7, p. 1777-1799, (2007)

WANG, JIGANG et al. **Reliable and efficient data transfer protocol based on UDP in cluster system.** In: First International Multi-Symposiums on Computer and Computational Sciences (IMSCCS'06). p. 518-524. (2006)

CIDON, ISRAEL et al. **Hybrid TCP-UDP transport for Web traffic**. In: 1999 IEEE International Performance, Computing and Communications Conference (Cat. No. 99CH36305). p. 177-184. (1999)

ECKART, BEN; HE, XUBIN; WU, QISHI. **Performance adaptive UDP for high-speed bulk data transfer over dedicated links.** In: 2008 IEEE International Symposium on Parallel and Distributed Processing. p. 1-10. (2008)

HERRERO, ROLANDO. **Analysis of the constrained application protocol over quick UDP internet connection transport**. Internet of Things, v. 12, p. 100328, (2020)