Uma imagem com texto, ClipArt

Descrição gerada automaticamente

Universidade do Minho

**Relatório do Trabalho Prático 2:**

**Protocolo IPv4 :: Datagramas IP e Fragmentação**

Ano letivo 2021/2022

Março 2022

Licenciatura em Engenharia Informática

Unidade Curricular de Redes de Computadores

Uma imagem com pessoa, parede, vestuário, interior

Descrição gerada automaticamenteUma imagem com pessoa, homem, parede, interior

Descrição gerada automaticamenteUma imagem com exterior, árvore, pessoa, pose

Descrição gerada automaticamente

Ana Gonçalves a93259 Luís Faria a93209 Jéssica Fernandes a93318

Índice

[Introdução 3](#_Toc99647590)

[Parte 1 4](#_Toc99647591)

[Questão 1 (Xubuntu) 4](#_Toc99647592)

[Questão 2 (Máquina Nativa: Ubuntu) 7](#_Toc99647593)

[Questão 3 (Máquina Nativa: Ubuntu com Tamanho de pacote = 4071) 11](#_Toc99647594)

[Parte 2 15](#_Toc99647595)

# Introdução

Neste relatório será abordado a resolução do trabalho prático da UC Redes de Computadores, cujo enunciado tinha como objetivo o aprofundamento do conhecimento sobre o protocolo IP, nomeadamente o formato de um pacote, a sua fragmentação, endereçamento e finalmente encaminhamento.

Deste modo, será apresentado neste documento, as resoluções da parte 1 e parte 2 , constituídas pelas perguntas e respostas justificadas, juntamente com uma conclusão do trabalho prático.

# Parte 1

## Questão 1 (Xubuntu)

***a) Active o wireshark ou o tcpdump no host Bela. Numa shell de Bela execute o comando traceroute -I para o endereço IP do Monstro***

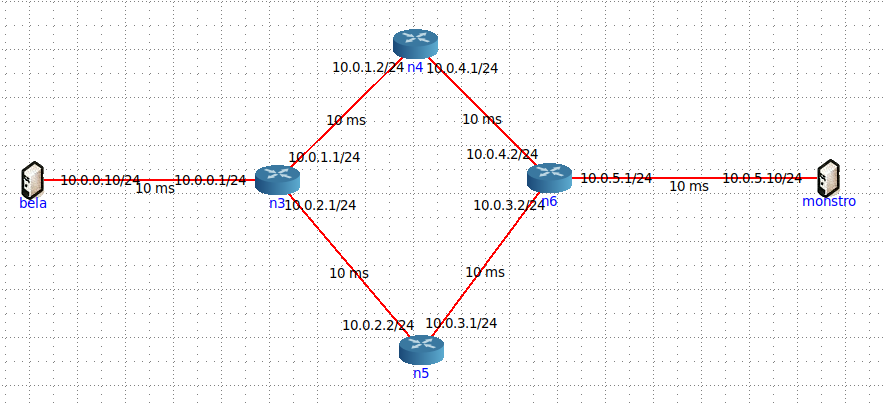


Figura : Topologia de Rede Core

Uma imagem com texto

Descrição gerada automaticamente

Figura : Resultado de Traceroute no host Bela

***b) Registe e analise o tráfego ICMP enviado pelo sistema Bela e o tráfego ICMP recebido como resposta. Comente os resultados face ao comportamento esperado.***

Uma imagem com mesa

Descrição gerada automaticamente

Figura : Tráfego capturado após Traceroute

Correspondem aos 3 pacotes enviados para o host Monstro com TTL de 1

Correspondem aos 3 pacotes enviados para o host Monstro com TTL de 2

Correspondem aos 3 pacotes enviados para o host Monstro com TTL de 3

E o ciclo continua até receber o primeiro pacote que confirma que chegou ao host Monstro.

Corresponde aos 3 pacotes enviados de volta ao host Bela que não conseguiram alcançar o host Monstro.

O comportamento esperado do Traceroute, consistia no envio de X pacotes com TTL a crescer linearmente, de modo a chegar ao destino pretendido. E com a figura acima apresentada podemos confirmar que é esse o comportamento que efetivamente acontece. Neste caso específico são enviados 3 pacotes com TTL igual, inicializado a 1.

Com a figura a seguir também confirmamos o caso de paragem:

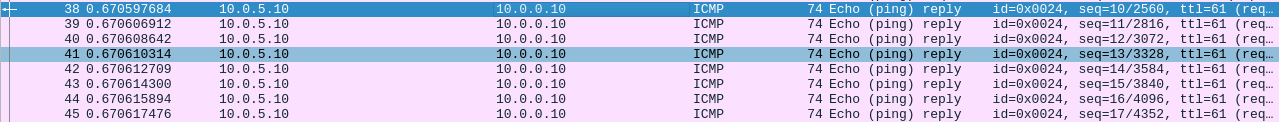


Figura : Tráfego capturado de Confirmação de Receção pelo Host Destino

Esta figura contêm os pacotes transmitidos imediatamente a seguir à figura 2. Nesta figura, verificamos que os 3 primeiros pacotes confirmam que o pacote anteriormente enviado chegou ao host. E por isso, a partir desse momento já não existe a necessidade enviar pacotes com maior TTL pois já se encontrou o valor de TTL necessário para alcançar o host.

***c) Qual deve ser o valor inicial mínimo do campo TTL para alcançar o servidor Monstro ? Verifique na prática que a sua resposta está correta***

Como verificamos na questão anterior, o host Bela recebeu 3 sequencias de 3 pacotes que não alcançaram o destino. Sendo que cada sequência está associada a um pacote enviado com TTL de 1,2 e 3 correspondentemente, podemos confirmar que o valor mínimo de TTL para alcançar o servidor Monstro é de 4.

Uma imagem com mesa

Descrição gerada automaticamente

Figura 5: Pacote selecionado como o primeiro pacote enviado pelo host destino

Nesta figura, conseguimos identificar que o pacote selecionado foi enviado com sucesso do host Monstro, em resposta ao pacote com TTL de 4.

***d) Calcule o valor médio do tempo de ida-e-volta (RTT - Round-Trip Time) obtido no acesso ao servidor. Para melhorar a média, poderá alterar o número pacotes de prova com a opção -q.***

Uma imagem com texto, quadro de resultados

Descrição gerada automaticamente

Figura 6: Comando traceroute com maior número de repetições, no host Bela para o host Monstro

Tratando-se a última linha dos vários RTT calculados, a média é igualada a 80.8338 ms.

***e) O valor médio do atraso num sentido (One-Way Delay) poderia ser calculado com precisão dividindo o RTT por dois? O que torna difícil o cálculo desta métrica?***

Não, o facto de que o caminho percorrido na transmissão Bela->Mostro poder ser diferente do caminho percorrido na transmissão Monstro->Bela implica que a divisão por 2 do RTT não vai ser uma versão fiável do One-Way Delay. Isto porque, se o caminho de ida for muito mais demorado, ao contrário do caminho de volta, o One-Way Delay num sentido vai ser muito mais elevado ao real, e o contrário acontece para o sentido oposto.

## Questão 2 (Máquina Nativa: Ubuntu)

Uma imagem com texto

Descrição gerada automaticamente***a) Qual é o endereço IP da interface ativa do seu computador?***

Figura 7: Pacote de Standard Query feita pela máquina nativa

Como podemos verificar o IP da interface do computador pessoal é 172.26.3.68

***b) Qual é o valor do campo protocolo? O que permite identificar?***

Na seguinte figura verificamos que o valor do campo do protocolo é ICMP, e é identificado pelo valor entre parentesis, que no caso de ICMP é 1.

Figura 8: Primeiro Pacote enviado pela máquina nativa

***c) Quantos bytes tem o cabeçalho IPv4? Quantos bytes tem o campo de dados (payload) do datagrama? Como se calcula o tamanho do payload?***

Para poder verificar estas informações é necessário selecionar o pacote ICMP correto, e recolher o tamanho do pacote IP 60 bytes. Sendo o tamanho do overhead de 20 bytes, fica confirmado que o tamanho do campo de dados é de:



Figura 9: Primeiro Pacote enviado pela máquina nativa

***d) O datagrama IP foi fragmentado? Justifique***

Não, podemos confirmar através da leitura do wireshark que não existe nenhuma notificação de fragmentação de pacote: através da flag a indicar mais fragmentos nem de offsets diferentes de 0.

Figura 10: Primeiro pacote enviado pela máquina nativa

***e) Ordene os pacotes capturados de acordo com o endereço IP fonte (e.g., selecionando o cabeçalho da coluna Source), e analise a sequência de tráfego ICMP gerado a partir do endereço IP atribuído à interface da sua máquina. Para a sequência de mensagens ICMP enviadas pelo seu computador, indique que campos do cabeçalho IP variam de pacote para pacote.***

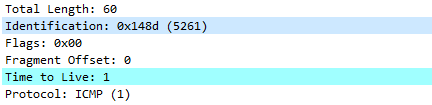


Figura 11: Primeiro pacote enviado

Uma imagem com texto

Descrição gerada automaticamente

Figura 12: Segundo pacote enviado

Uma imagem com texto

Descrição gerada automaticamente

Figura 13: Terceiro pacote enviado

Uma imagem com texto

Descrição gerada automaticamente

Figura 14: Quarto pacote enviado

As figuras anteriores apresentam os campos de IP do primeiro, segundo, terceiro e quarto pacote, respetivamente. Como são enviados 3 pacotes por 1 TTL, dentro desses grupos de 3 pacotes (Figura 11, 12 e 13 correspondem ao primeiro grupo enviado) , varia apenas o identificados, e de grupo em grupo, o TTL varia adicionalmente.

***f) Observa algum padrão nos valores do campo de Identificação do datagrama IP e TTL?***

Sim, verificamos que o identificador aumenta linearmente a cada pacote, e o TTL aumenta linearmente de 3 em 3 pacotes, iniciando o ciclo em TTL igual a 1.

***g) Ordene o tráfego capturado por endereço destino e encontre a série de respostas ICMP TTL excedido enviadas ao seu computador. Qual é o valor do campo TTL? Esse valor permanece constante para todas as mensagens de resposta ICMP TTL excedido enviados ao seu host? Porquê?***

Uma imagem com texto

Descrição gerada automaticamenteNos vários pacotes ICMP com Time-to-live excedido, foi verificado que os TTL recebidos são todos elevados e maioritariamente constantes (254 / 255). Isto deve-se ao facto de que os routers e o host não terem informação da quantidade de saltos que os pacotes vão ter de percorrer para chegar ao destino. E por isso, é necessário um valor alto de TTL para confirmar a sua chegada ao destino correto.

Figura 15: Primeiro pacote recebido por TTL excedido

## Questão 3 (Máquina Nativa: Ubuntu com Tamanho de pacote = 4071)

***a) Localize a primeira mensagem ICMP. Porque é que houve necessidade de fragmentar o pacote inicial?***

A primeira mensagem ICMP enviada foi a seleciona na seguinte figura:

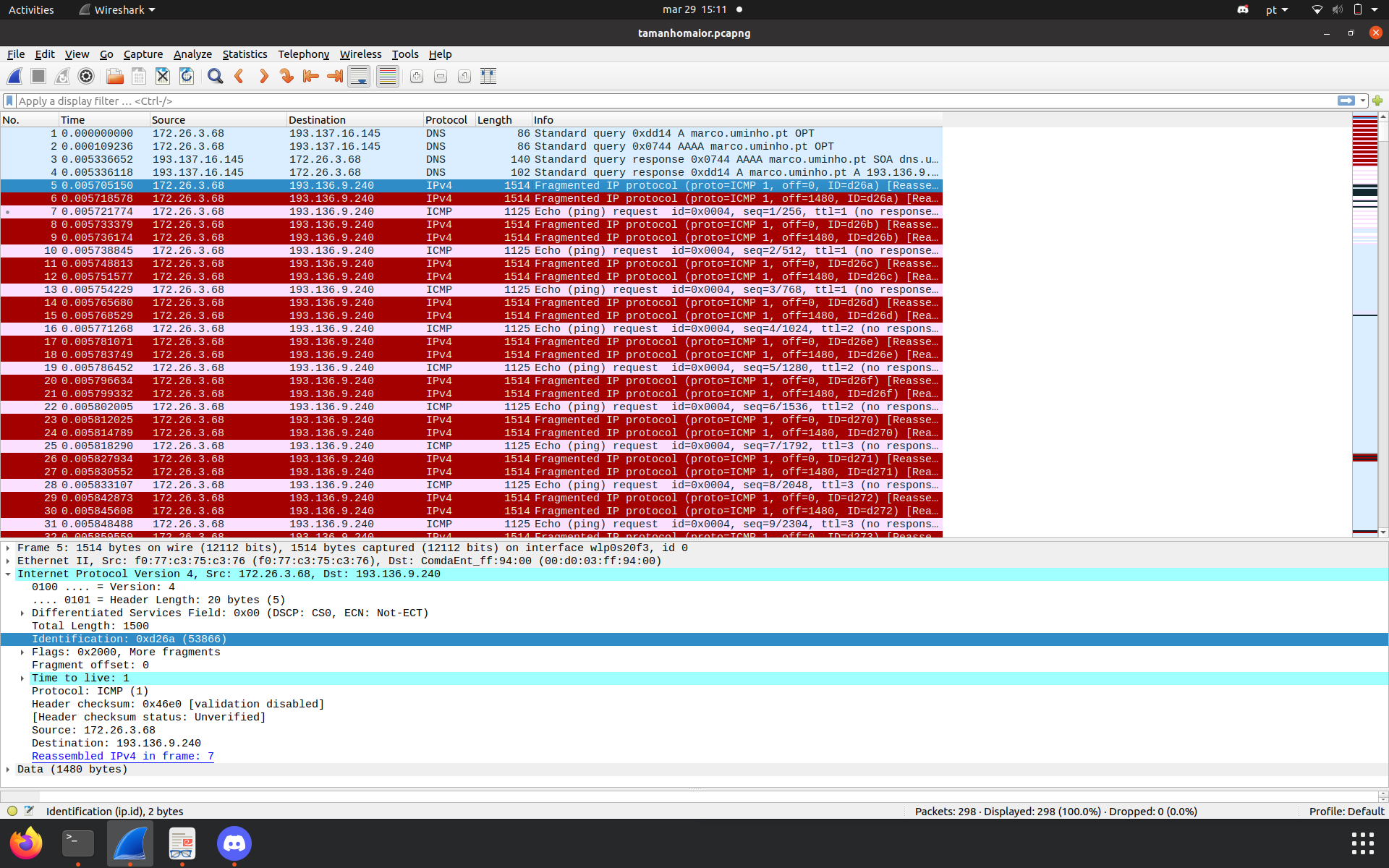


Figura 16: Primeiro fragmento enviado

Esta fragmentação ocorreu por uma única razão: o tamanho do pacote que foi forçado. O facto de o traceroute enviar um pacote de tamanho tão grande, obrigou a uma fragmentação deste, mais especificamente, em 3 pacote.

***b) Imprima o primeiro fragmento do datagrama IP segmentado. Que informação no cabeçalho indica que o datagrama foi fragmentado? Que informação no cabeçalho IP indica que se trata do primeiro fragmento? Qual é o tamanho deste datagrama IP?***

Selecionando o primeiro fragmento do enviado, verificamos que podemos confirmar que se trata de um pacote fragmento graças à flag levantada que anuncia a existência de mais pacotes fragmentos por receber.

Identificamos que se trata do primeiro fragmento pois o seu offset é igual a 0, o que implica que na reconstrução do pacote original, ele vai ser inserido no início do pacote construído.

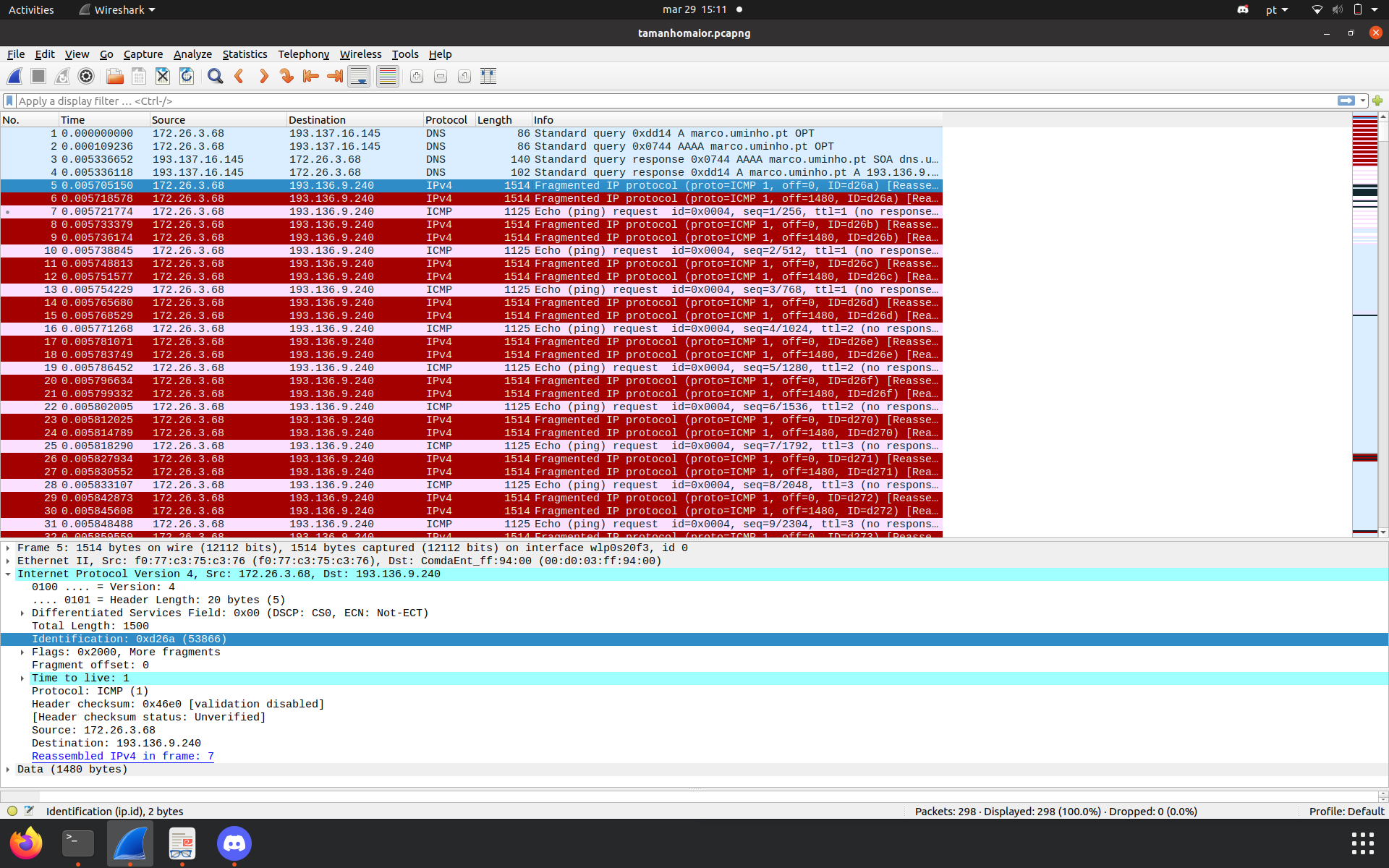
E finalmente, como podemos verificar pela figura, o tamanho do primeiro datagrama fragmentado é de 1500 bytes.

Figura 17: Primeiro fragmento enviado

***c) Imprima o segundo fragmento do datagrama IP original. Que informação do cabeçalho IP indica que não se trata do 1º fragmento? Há mais fragmentos? O que nos permite afirmar isso?***

Na seguinte figura, verificamos que não se trata do primeiro fragmento visto que offset já não se encontra igual a 0. No entanto, ainda existem mais fragmentos por enviar pois a flag “more fragments” ainda se encontra levantada.

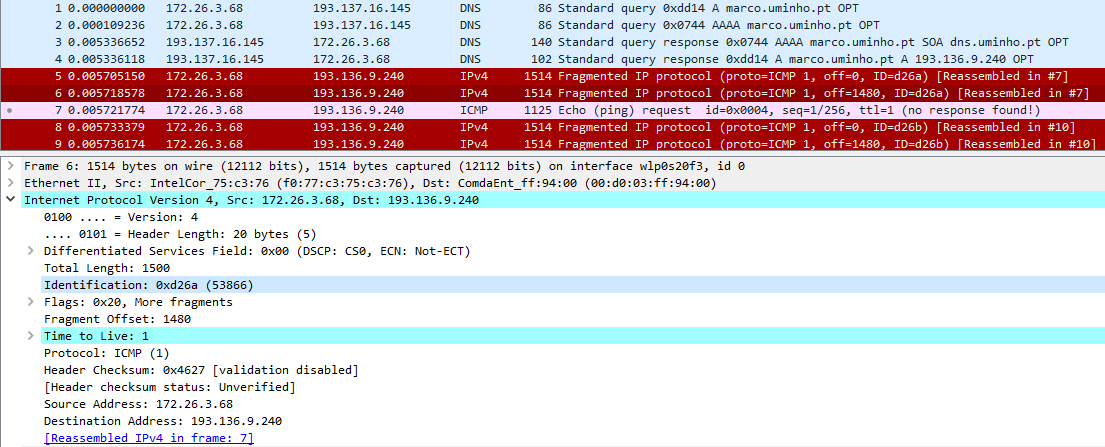


Figura 18: Segundo fragmento enviado

***d) Quantos fragmentos foram criados a partir do datagrama original?***

Como podemos verificar, foram criados 3 fragmentos apartir de um datagrama de tamanho 4071 bytes.

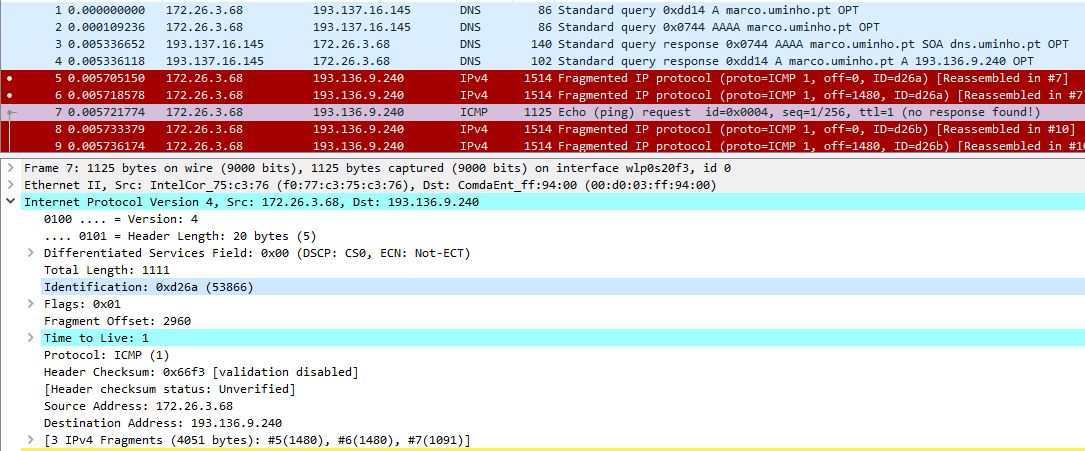


Figura 19: Terceiro e último fragmento do primeiro datagrama original

***f) Indique, resumindo, os campos que mudam no cabeçalho IP entre os diferentes fragmentos, e explique a forma como essa informação permite reconstruir o datagrama original.***

Considerando um pacote original fragmentado, cada um dos fragmentos vai ter o mesmo identificadores e um offset diferente. Para construir o datagrama original, será necessário reunir todos os pacotes recebidos com o mesmo identificador até a flag de “more fragments” já não estiver levantada, e reuni-los de acordo com os seus offsets, começando pelo offset igual a 0.

Uma imagem com texto

Descrição gerada automaticamente

Figura 20: Primeiro fragmento

Uma imagem com texto

Descrição gerada automaticamente

Figura 21: Segundo Fragmento

Uma imagem com texto

Descrição gerada automaticamente

Figura 22: Terceiro e último fragmento

***e) Verifique o processo de fragmentação através de um processo de cálculo.***

Sendo o tamanho original do datagrama de 4071, vamos verificar como ocorreu a fragmentação capturada pelo wireshark.

Se não ocorrer fragmentação, seria enviado um pacote com tamanho de 4071 bytes, em que 20 deles corresponderiam ao overhead e 4051 de conteúdo. No entanto, com a fragmentação, isso não aconteceu.

O que efetivamente aconteceu foi que um dos pacotes foi preenchido com o máximo de conteúdo possível, 1480 bytes, juntamente com 20 bytes, correspondendo a um tamanho total de 1500 bytes, como se pode verificar na figura 20 e 21. Esta situação ocorre 2 vezes, pois o tamanho do pacote obriga a recorrer a 3 pacotes fragmentados. No entanto, no terceiro fragmento, a quantidade de dados restante vai ser igual a 4051 – 1480 – 1480 = 1091. Sendo esta a quantidade de dados, e 20 bytes de overhead, vai resultar num pacote de tamanho de 111, confirmado pela figura 22.

***h) Escreva uma expressão lógica que permita detetar o último fragmento correspondente ao datagrama original***

Através de pseudocódigo conseguimos fazer esta identificação:

While (flag != “more fragments” || wantedID == currentID || offset != 0)

Fragment = newFragment

LastFragment = Fragment

# Parte 2