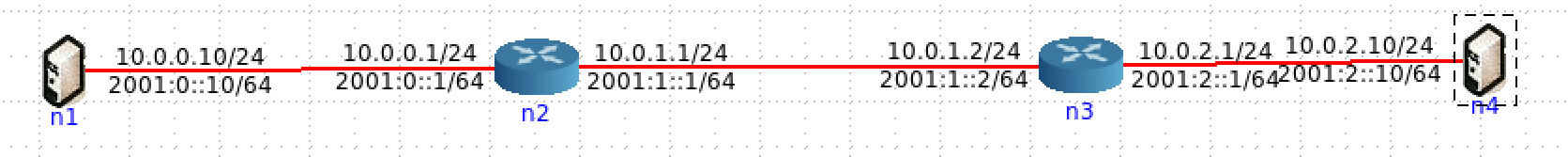
**TP4 – Protocolo IPv4**

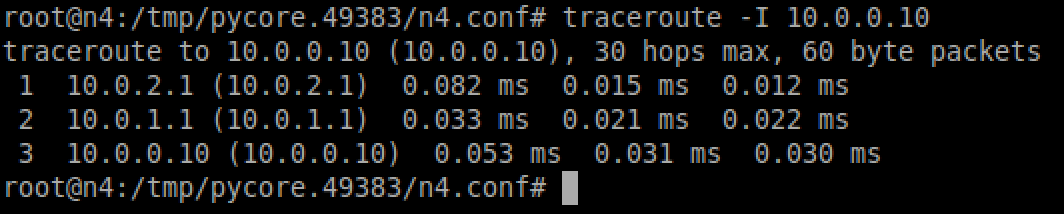
**Questões e Respostas**

**Parte I**

1. **Prepare uma topologia CORE para verificar o comportamento do *traceroute*. Ligue um *host* n1 a um router n2; o router n2 a um router n3 que, por sua vez, se liga a um *host* n4 (note que pode não existir conectividade IP imediata entre n1 e n4 até que o *routing* estabilize).**

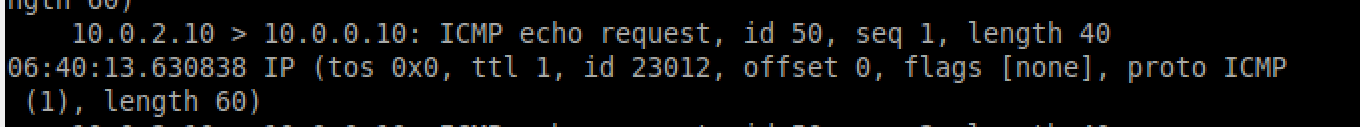


1. **Active o *wireshark* ou o *tcpdump* no *host* n4. Numa *shell* de n4, execute o comando *traceroute* -I para o endereço IP do *host* n1.**

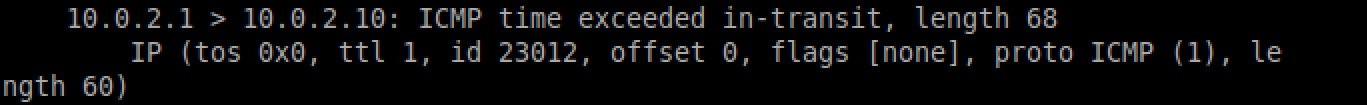


1. **Registe e analise o tráfego ICMP enviado por n4 e o tráfego ICMP recebido como resposta. Comente os resultados face ao comportamento esperado.**

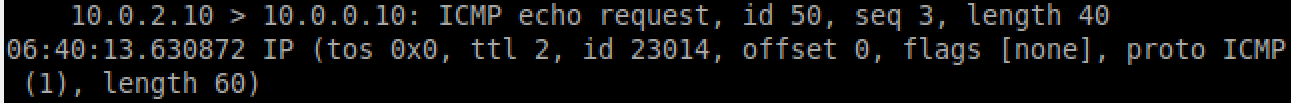
Primeiramente, n4 envia três datagramas (correspondentes a um *echo request*) com o campo TTL (*Time To Live*) igual a 1.

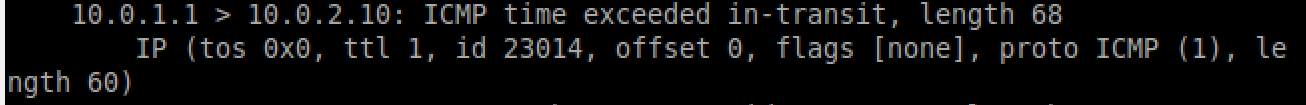


Como o número de saltos é inferior ao mínimo necessário para chegar de n4 a n1, é recebida uma mensagem de controlo ICMP (*Internet Control Message Protocol*) informando da falha no envio (i.e. *time exceeded in-transit*), procedente de n3. É de notar, no entanto, que apesar de a mensagem ICMP fazer referência a um tempo excedido, na verdade, o que aconteceu foi que o datagrama utilizou todos os saltos possíveis (definidos pelo TTL), sendo que a falha no envio não se deve a nenhuma razão temporal.

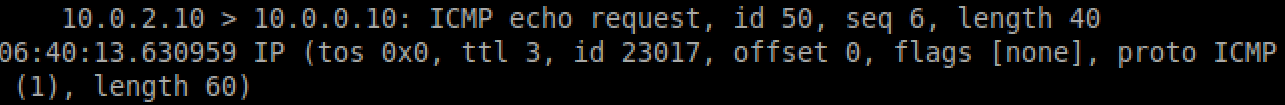


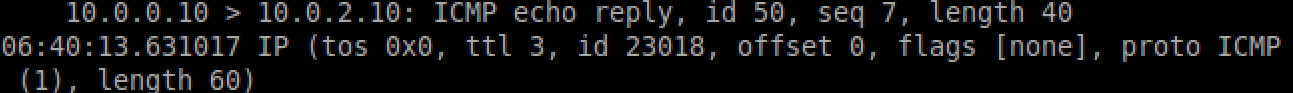
De seguida, são enviados mais três datagramas com o campo TTL igual a 2, ocorrendo o mesmo (sendo que a mensagem ICMP, *time exceeded in-transit*, é enviada por n2 e não por n3).



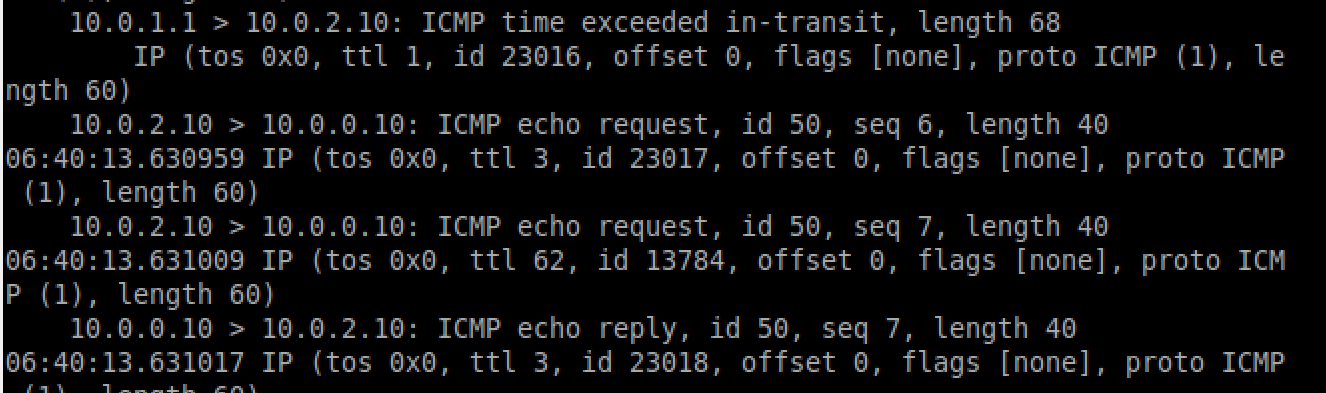


Por último, são enviados outros três datagramas mas agora com o TTL igual a 3 pelo que, como já se atingiu número mínimo de saltos para chegar de n4 a n1, é recebida a resposta ao *echo* *request* (i.e. um *echo* *reply*) procedente de n1.





1. **Qual deve ser o valor inicial mínimo do campo TTL para alcançar o destino n1? Verifique na prática que a sua resposta está correta.**



O valor inicial mínimo do campo TTL para alcançar o destino n1 deve ser 3 (de n4 para n3 (1), de n3 para n2 (2) e de n2 para n1 (3)\).

Na prática, é isso que acontece: verifica-se que para os datagramas enviados por n4 com o TTL igual a 3, é enviado (por n1) a resposta ao *echo request* (i.e. um *echo reply*).

1. **Qual o valor médio do tempo de ida-e‐volta (*Round-Trip Time*) obtido?**

O valor médio do tempo de ida-e-volta (RTT – *Round-Trip Time*) é de .

1. **Pretende‐se agora usar o *traceroute* na sua máquina nativa, e gerar de datagramas IP de diferentes tamanhos.**
2. **Qual é o endereço IP da interface ativa do seu computador?**

C:\Users\win8\AppData\Local\Microsoft\Windows\INetCacheContent.Word\2-a-IP.PNG

O endereço IP da interface ativa do nosso computador é 192.168.100.200.

1. **Qual é o valor do campo protocolo? O que identifica?**

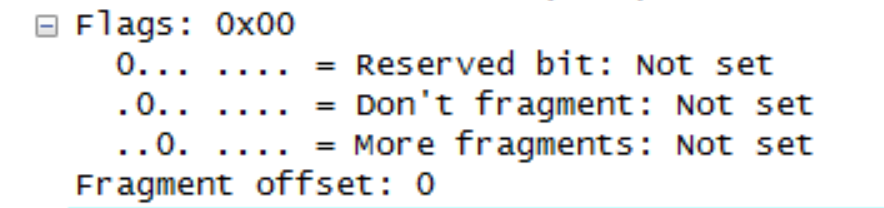
C:\Users\win8\AppData\Local\Microsoft\Windows\INetCacheContent.Word\2-b-PROTOCOL.PNG

O valor do campo protocolo é ICMP (1). Identifica o *Internet Control Message Protocol*.

1. **Quantos *bytes* tem o cabeçalho IP(v4)? Quantos bytes tem o campo de dados (*payload*) do datagrama? Como se calcula o tamanho do *payload*?**

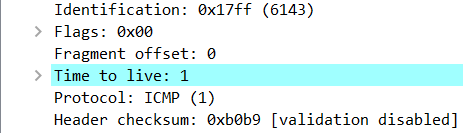
O cabeçalho IPv4 tem 20 bytes. O campo de dados (*payload*) do datagrama tem 28 bytes. O *payload* é dado pelo valor do cabeçalho somado com os primeiros 8 bytes do campo de dados do datagrama.

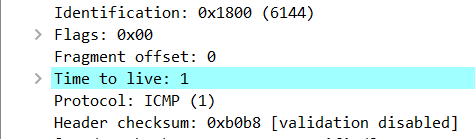
1. **O datagrama IP foi fragmentado? Justifique.**

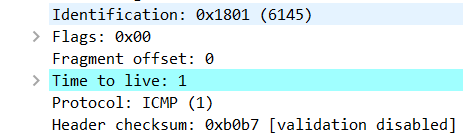


O datagrama não foi fragmentado visto que o campo *More fragments* está a 0 e o *Fragment offset* também está a 0. Ou seja, como o payload nesta trama começa na posição 0 (do datagrama original) – segundo o *Fragment* *offset* – e como não se esperam mais fragmentos desse datagrama original – segundo o *More fragments* – pode-se concluir que esta trama transporta o datagrama original na totalidade.

1. **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 sua máquina. Para a sequência de mensagens ICMP enviadas pelo seu computador, indique que campos do cabeçalho IP variam de pacote para pacote.**

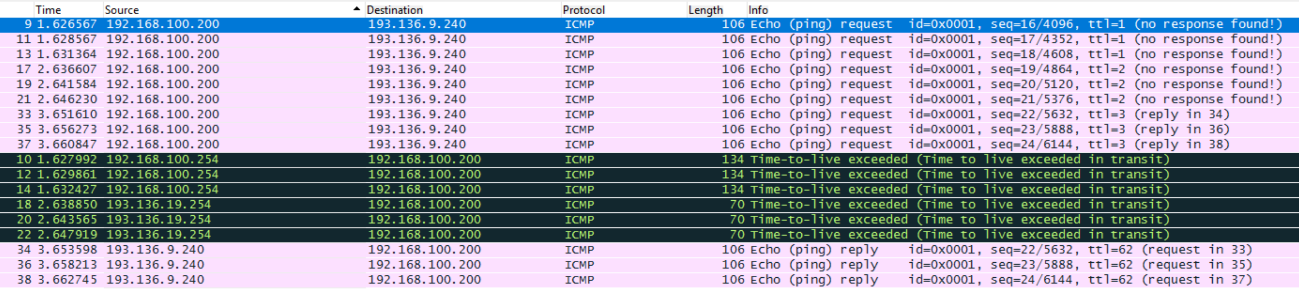






Segundo o *Wireshark*, os campos que variam são: a identificação (*Identification*), o TTL e o *header checksum*.

1. **Observa algum padrão nos valores do campo de Identificação do datagrama IP e TTL?**



São enviadas três tramas com TTL igual a 1, três com TTL igual a 2 e outras três com TTL igual a 3.

Só as últimas três tramas chegam ao destino (33, 35 e 37). Todas as outras deram origem a respostas ICMP do tipo “*Time to live exceeded”* (saltos insuficientes para atingir o destino).

A campo da identificação (*Identification*) é sempre incrementado em uma unidade para as tramas enviadas para o mesmo destino.

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

Para os *Echo Requests* relativos às tramas com TTL igual a 1, o valor do campo TTL das mensagens de resposta ICMP TTL *exceeded* é igual a 64.

Para os *Echo Requests* relativos às tramas com TTL igual a 2, o valor do campo TTL das mensagens de resposta ICMP TTL *exceeded* é igual a 254.

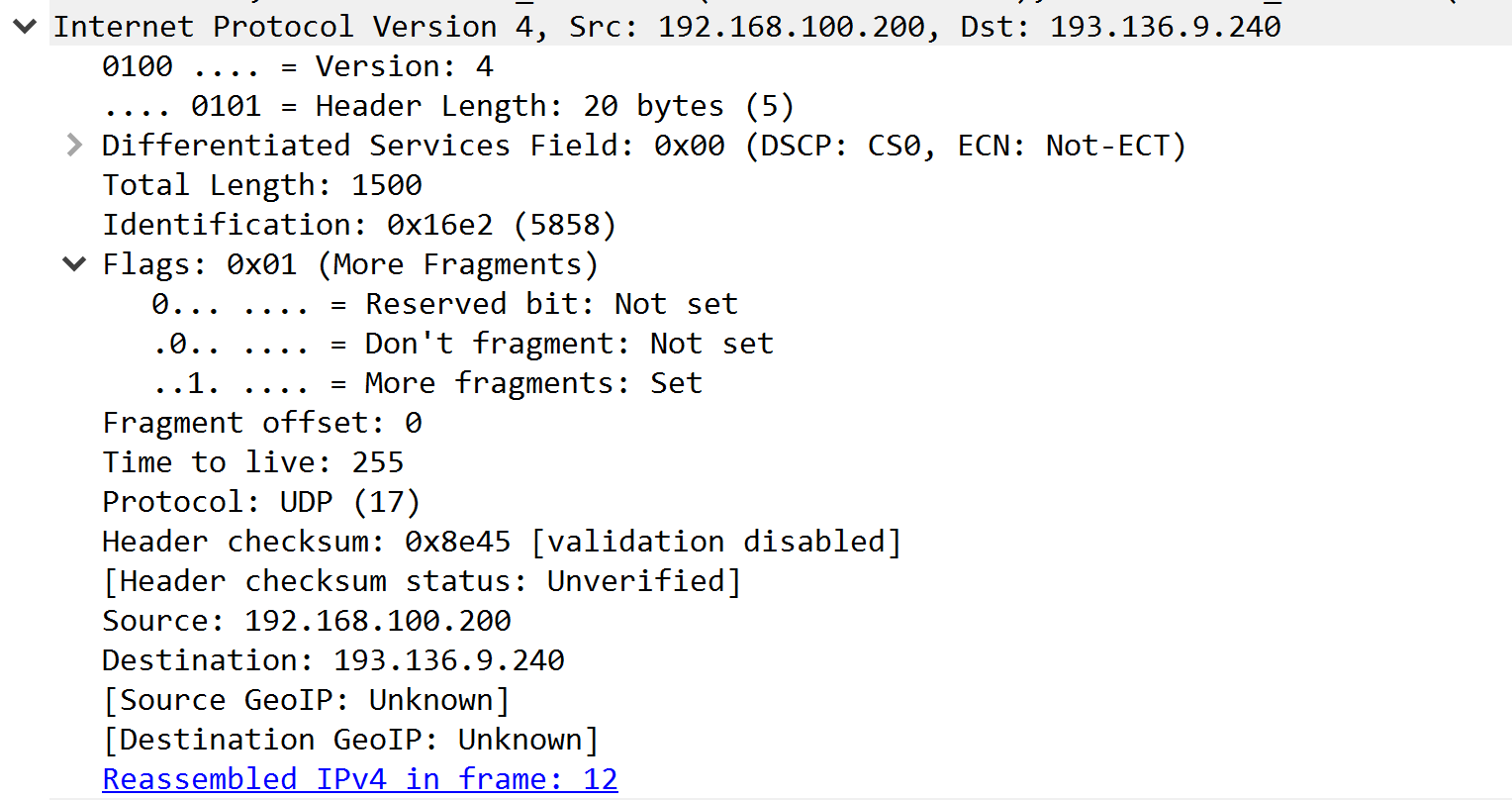
Porque é que varia? Router não sabe quando lhe chega TTL = 0 (quantos saltos houve)

1. **Pretende-se agora analisar a fragmentação de pacotes IP. Reponha a ordem do tráfego capturado usando a coluna do tempo de captura. Observe o tráfego depois do tamanho de pacote ter sido definido para 40XX bytes.**
2. **Localize a primeira mensagem ICMP. Porque é que houve necessidade de fragmentar o pacote inicial?**



Houve necessidade de fragmentar o pacote inicial porque o protocolo IP só suporta pacotes até 1500 bytes. Como o tamanho foi definido para 4021 bytes, houve necessidade de fragmentar o pacote inicial em três mais pequenos.

1. **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?**

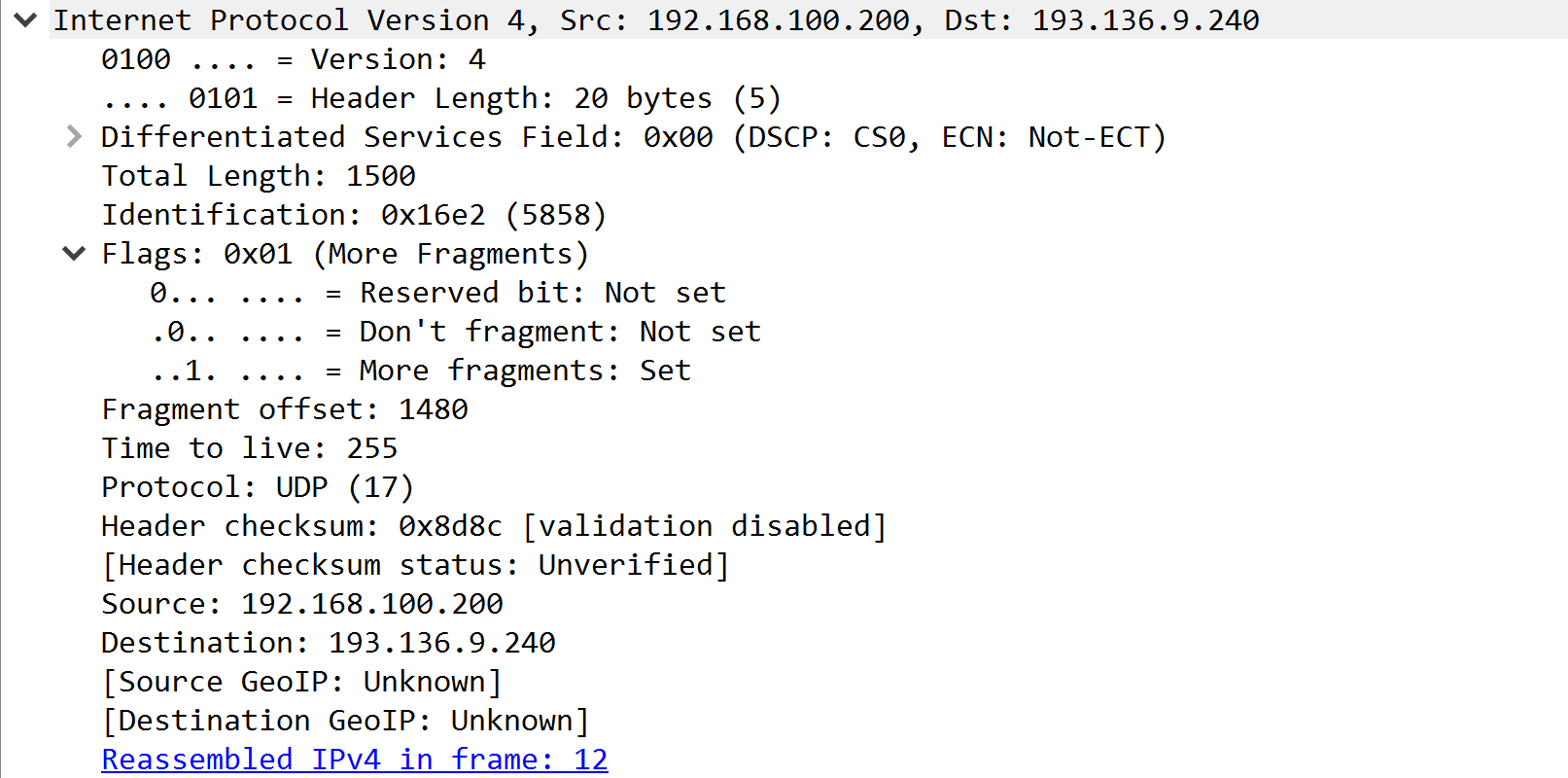


A informação do cabeçalho que indica que o datagrama foi fragmentada é a *flag* *More fragments*. Verifica-se que se trata do primeiro fragmento devido ao campo do *fragment offset* estar a zero.



O tamanho do datagrama IP é de 1514 bytes.

1. **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?**

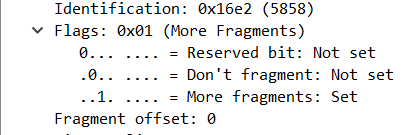


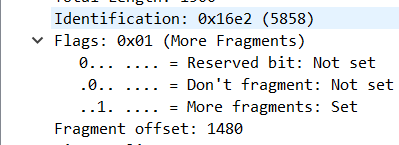
Podemos verificar que não se trata do primeiro fragmento devido ao campo do *fragment offset* que está a 1480. Sim, ainda existem mais fragmentos. Podemos confirmar tal informação a partir do valor da *flag* relativa à existência de mais fragmentos (“*More fragments”*) que está a 1.

É importante notar que podemos verificar que se tratam de fragmentos correspondentes ao mesmo datagrama original a partir do campo de identificação (*Identification*).

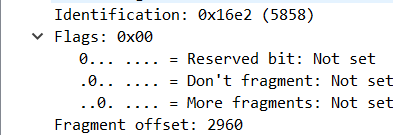
1. **Quantos fragmentos foram criados a partir do datagrama original? Como se deteta o último fragmento correspondente ao datagrama original?**

Foram criados três fragmentos a partir do datagrama original. Em primeiro lugar, deteta-se um fragmento correspondente ao datagrama original através do valor hexa do identificador (*Identification* – que se mantém constante para todos os fragmentos do datagrama original).





Assim, para detetar o último fragmento correspondente ao datagrama original, primeiro confirma-se o campo *Identification* e, de seguida, verifica-se que se trata mesmo do último fragmento através da *flag* relativa à existência de mais fragmentos (*More fragments*) que se deve encontrar a 0.



1. **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.**

Como se pode verificar nas imagens anteriores (que são relativas ao mesmo datagrama original), os únicos campos que mudam são: *More fragments* e *Fragment offset*. Sabemos que fazem todos parte do mesmo datagrama original porque o campo *Identification* é igual para os três.

Através do *Fragment offset* sabemos em que posição do datagrama original a informação desta trama começa. Com a *flag* *More fragments* conseguimos saber se existem mais fragmentos desse datagrama para chegar. Saberemos que estamos no último fragmento quando a *Identification* for igual às restantes (imagem anterior) e *More fragments* for igual a 0. Com estas informações conseguimos reconstruir o datagrama original a partir dos vários fragmentos.

**Parte II**