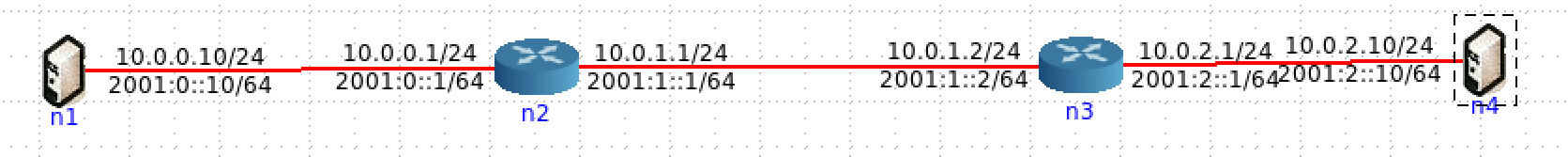
**TP4 – Protocolo IPv4**

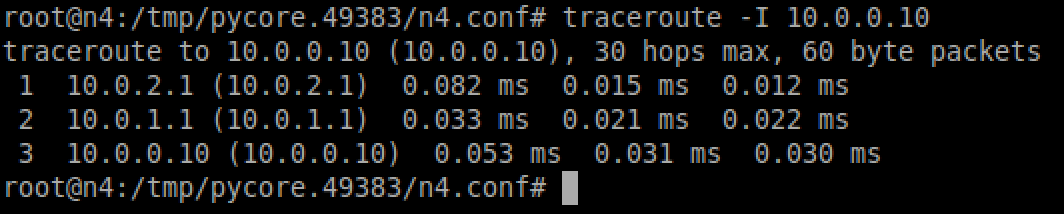
**Questões e Respostas**

**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).**

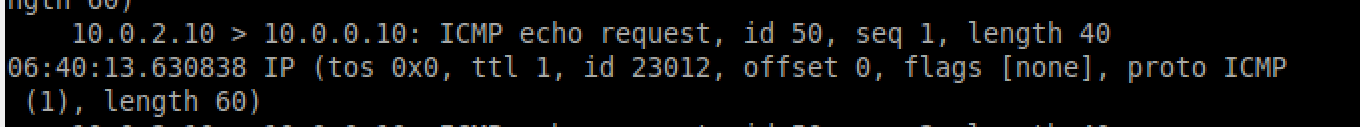


1. **Active o *wireshark* ou o *tcpdump* no *host* n4. Numa *shell* de n4, execute o comando *traceroute* -I para o endereço IP do *host* n1.**

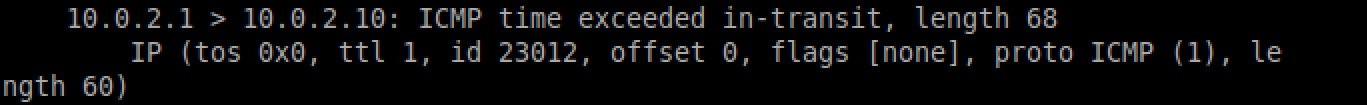


1. **Registe e analise o tráfego ICMP enviado por n4 e o tráfego ICMP recebido como resposta. Comente os resultados face ao comportamento esperado.**

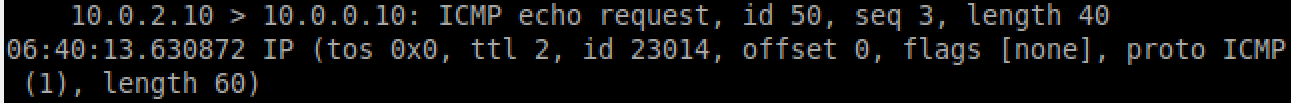
Primeiramente, n4 envia três datagramas (correspondentes a um *echo request*) com o campo TTL (*Time To Live*) igual a 1.

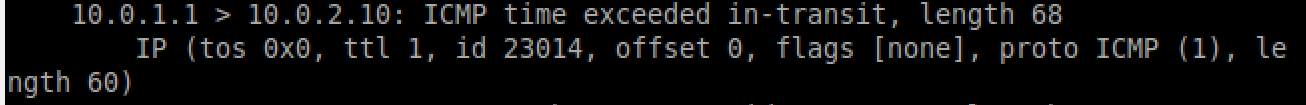


Como o número de saltos é inferior ao mínimo necessário para chegar de n4 a n1, é recebida uma mensagem de controlo ICMP (*Internet Control Message Protocol*) informando da falha no envio (i.e. *time exceeded in-transit*), procedente de n3. É de notar, no entanto, que apesar de a mensagem ICMP fazer 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.

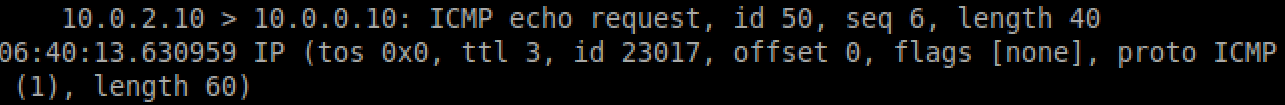


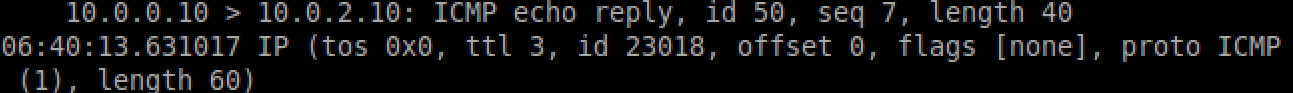
De seguida, são enviados mais três datagramas com o campo TTL igual a 2, ocorrendo o mesmo (sendo que a mensagem ICMP, *time exceeded in-transit*, é enviada por n2 e não por n3).



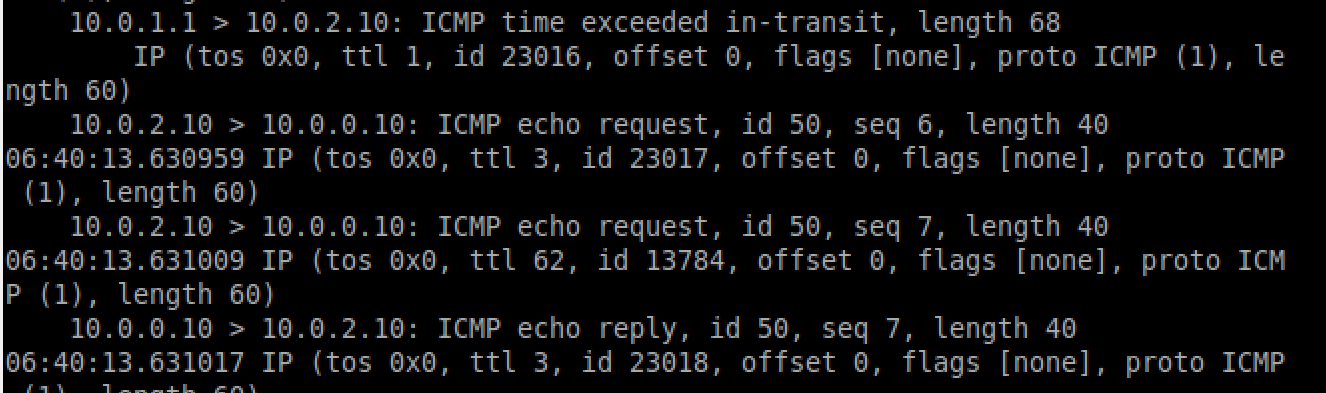


Por último, são enviados outros três datagramas mas agora com o TTL igual a 3 pelo que, como já se atingiu número mínimo de saltos para chegar de n4 a n1, é recebida a resposta ao *echo* *request* (i.e. um *echo* *reply*) procedente de n1.





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?**

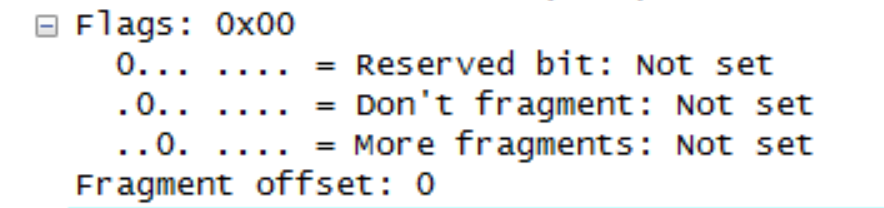
C:\Users\win8\AppData\Local\Microsoft\Windows\INetCacheContent.Word\2-b-PROTOCOL.PNG

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*?**

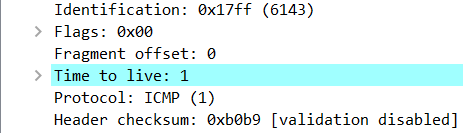
O cabeçalho IPv4 tem 20 bytes. O campo de dados (*payload*) do datagrama tem 28 bytes. O *payload* é dado pelo valor do cabeçalho somado com os primeiros 8 bytes do campo de dados do datagrama.

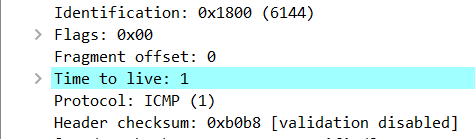
1. **O datagrama IP foi fragmentado? Justifique.**

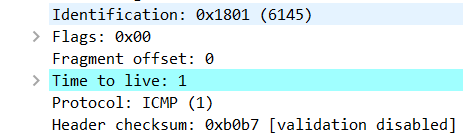


O datagrama não foi fragmentado visto que o campo *More fragments* está a 0 e o *Fragment offset* também está a 0. Ou seja, como o payload nesta trama começa na posição 0 (do datagrama original) – segundo o *Fragment* *offset* – e como não se esperam mais fragmentos desse datagrama original – segundo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

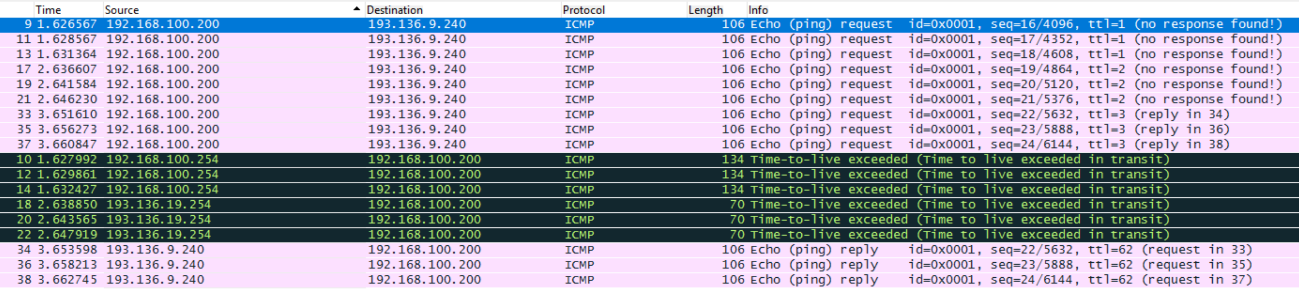






Segundo o *Wireshark*, os campos que variam são: a identificação (*Identification*), o TTL e o *header checksum*.

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.

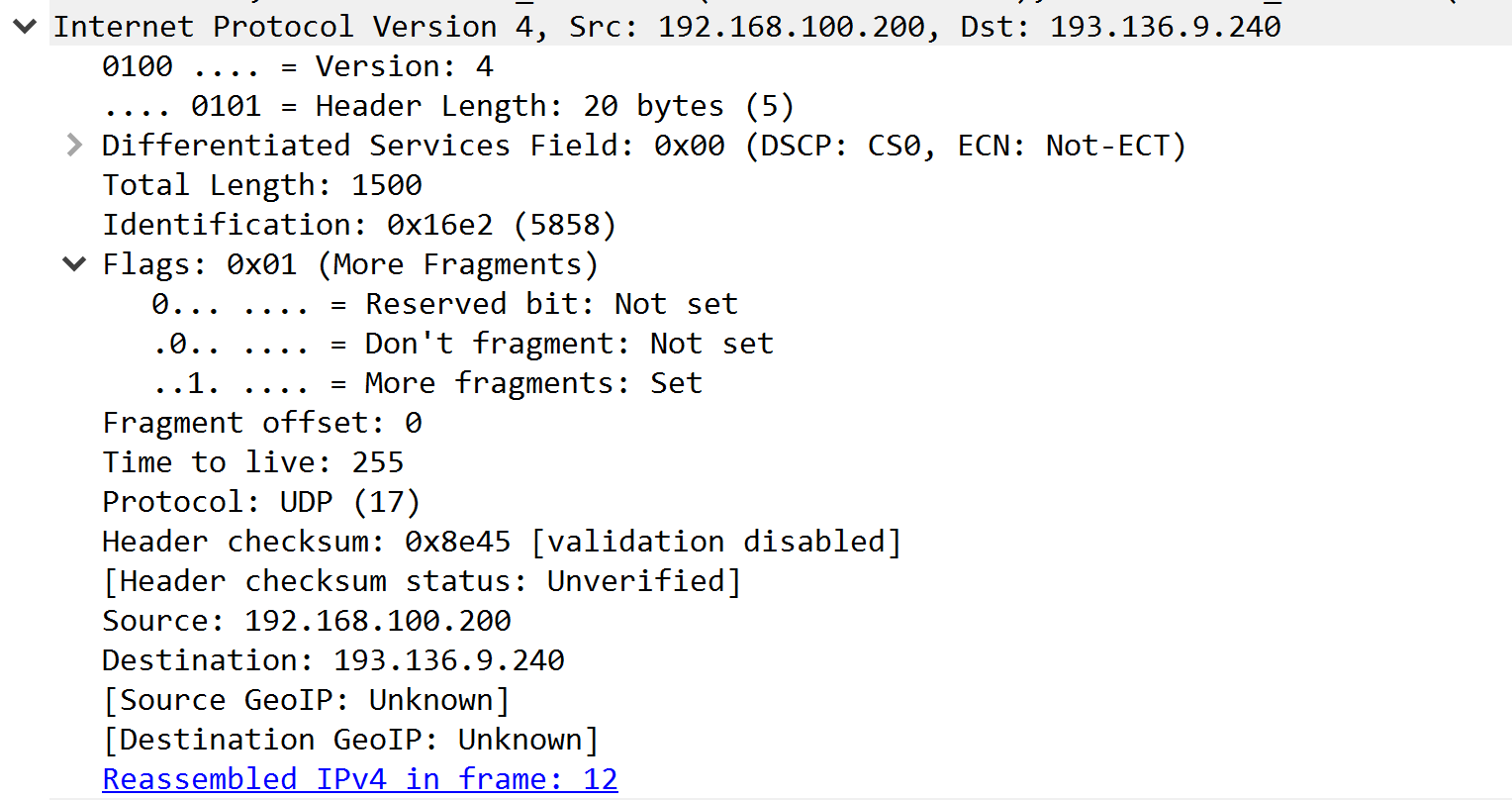
Porque é que varia? Router não sabe quando lhe chega TTL = 0 (quantos saltos houve)

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?**

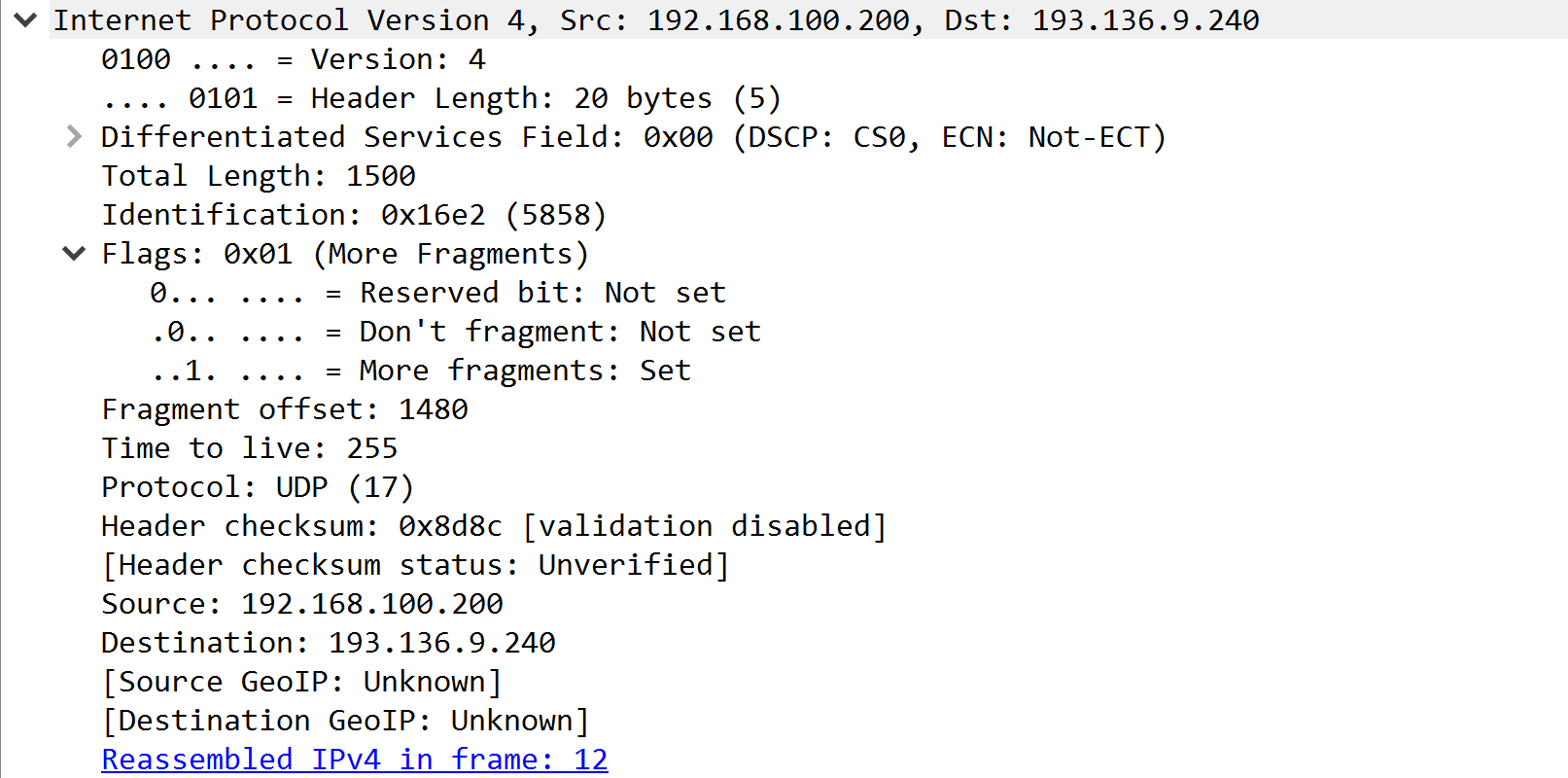


A informação do cabeçalho que indica que o datagrama foi fragmentada é a *flag* *More fragments*. Verifica-se que se trata do primeiro fragmento devido ao campo do *fragment offset* estar a zero.



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?**

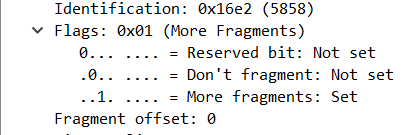


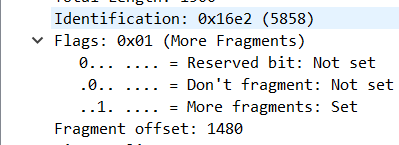
Podemos verificar que não se trata do primeiro fragmento devido ao campo do *fragment offset* que está a 1480. Sim, ainda existem mais fragmentos. Podemos confirmar tal informação a partir do valor da *flag* relativa à existência de mais fragmentos (“*More fragments”*) que está a 1.

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

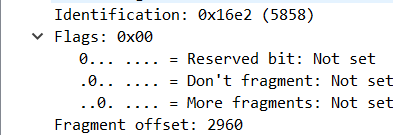
1. **Quantos fragmentos foram criados a partir do datagrama original? Como se deteta o último fragmento correspondente ao datagrama original?**

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**

**2 – Endereçamento e Encaminhamento IP**

1. **Atenda aos endereços IP atribuídos automaticamente pelo CORE aos diversos equipamentos da topologia.**
2. **Indique que endereços IP e máscaras de rede foram atribuídos pelo CORE a cada equipamento. Se preferir, pode incluir uma imagem que ilustre de forma clara a topologia e o endereçamento.**
3. **Tratam‐se de endereços públicos ou privados? Porquê?**

Tratam-se de endereços privados pois correspondem a endereços IP da classe A (entre 10.0.0.0 e 10.255.255.255). São também designados de *unique local access* (ULA), sendo endereços reservados à rede local dos departamentos do MIEInet, não sendo possível aceder-lhes diretamente através da rede internet global.

1. **Porque razão não é atribuído um endereço IP aos *switches*?**

Não é atribuído um endereço IP aos *switches* pois estes são apenas dispositivos de ligação da rede (e não um *host*), que atua ao nível da Ethernet, enviando tramas deste tipo e encaminhando o tráfego da rede.

1. **Usando o comando *ping* certifique‐se que existe conectividade IP entre os laptops dos utilizadores e o servidor do departamento A (basta certificar a conectividade de um laptop por departamento).**

Para testar a conectividade, enviámos múltiplos *pings* de um portátil de cada departamento (n5 para A, n7 para B e n8 para C) para o servidor presente no departamento A (n4).

1. **Para o router e um laptop do departamento A:**
2. **Execute o comando *netstat -rn* por forma a poder consultar a tabela de encaminhamento *unicast* (IPv4). Inclua no seu relatório as tabelas de encaminhamento obtidas; interprete as várias entradas de cada tabela. Se necessário, consulte o manual respetivo (*man* *netstat*).**

Para o router n1**:**

Cada uma das entradas da tabela diz-nos que para enviar um datagrama destinado a *Destination*, este tem de ser entregue a *Gateway* e sair pela interface *Iface*. Mais concretamente:

* A primeira linha diz-nos que para enviar um datagrama com destino 10.0.0.0, este tem de ser entregue a 0.0.0.0 e sair pela interface eth0.
* A segunda linha diz-nos que para enviar um datagrama com destino 10.0.1.0, este tem de ser entregue a 0.0.0.0 e sair pela interface eth1.
* A terceira linha diz-nos que para enviar um datagrama com destino 10.0.2.0, este tem de ser entregue a 10.0.0.2 e sair pela interface eth0.
* A quarta linha diz-nos que para enviar um datagrama com destino 10.0.3.0, este tem de ser entregue a 0.0.0.0 e sair pela interface eth2.
* A quinta linha diz-nos que para enviar um datagrama com destino 10.0.4.0, este tem de ser entregue a 0.0.0.0 e sair pela interface eth3.
* A sexta linha diz-nos que para enviar um datagrama com destino 10.0.5.0, este tem de ser entregue a 10.0.1.2 e sair pela interface eth1.
* A sétima linha diz-nos que para enviar um datagrama com destino 10.0.6.0, este tem de ser entregue a 10.0.0.2 e sair pela interface eth0.

Para o laptop n5:

* A primeira linha diz-nos que para enviar um datagrama com destino 0.0.0.0 (*default*), este tem de ser entregue a 10.0.4.1 e sair pela interface eth0.
* A segunda linha diz-nos que para enviar um datagrama com destino 10.0.4.0, este tem de ser entregue a 0.0.0.0 e sair pela interface eth0.

**INSERIR PRINTS LAPTOP**

Quando o *gateway* é 0.0.0.0, isto indica-nos que o destino já está ligado, diretamente, ao própriodispositivo de origem (*router/laptop*). No contexto do problema, isto acontece (i.e. *gateway* a 0.0.0.0) quando o *router* n1 comunica com n2, n3, n4 e, através do *switch* n11, com n5, n6 e n14 ou quando o laptop n5 comunica com n1, n6 e n14.

Quando a *destination* é 0.0.0.0, estamos perante um caminho por defeito (ou *default*) – como, por exemplo, na segunda entrada da tabela de encaminhamento de n5 – que nos indica, no caso de n5, que quando for necessário enviar um datagrama cujo IP de destino não esteja na subrede 10.0.4.0 (a única outra entrada da tabela), deverá ser enviado, por defeito, através do dispositivo correspondente ao *gateway* 10.0.4.1 (*router* n1), que se encarregará de encaminhar o datagrama ao destino.

Na terceira coluna das tabelas de encaminhamento podemos ainda verificar a presença das *flags* U e G que fazem referência, respetivamente, à validade de um caminho (a *source* e a *destination* estão, de facto, conectadas – diretamente ou não) *e* ao facto do encaminhamento passar por um *gateway* intermédio. Quando a flag G não aparece nalguma entrada, podemos dizer que o dispositivo de destino se encontra ligado diretamente ao de origem.

1. **Diga, justificando, se está a ser usado encaminhamento estático ou dinâmico (sugestão: analise que processos estão a correr em cada sistema).**

Está a ser usado encaminhamento estático, visto que na alínea anterior ao verificar ao usar *netstat* *-rn* no router, nas *flags* não aparece nenhum D (que corresponde a encaminhamento dinâmico).

1. **Admita que, por questões administrativas, a rota por defeito (0.0.0.0 ou *default*) deve ser retirada definitivamente da tabela de encaminhamento do servidor localizado no departamento A. Use o comando *route* *delete* para o efeito. Que implicações tem esta medida para os utilizadores da empresa que acedem ao servidor. Justifique.**

Ao utilizar o comando *route* *delete* *default* no servidor n4, a *route* por *default* (0.0.0.0) foi eliminada como podemos ver no *netstat* *-rn* anteriormente executado. As implicações que esta medida irá ter para os utilizadores é a impossibilidade de o servidor conseguir encaminhar datagramas cujo IP destino não seja pertencente à subrede 10.0.3.0. Como podemos ver nas imagens, é impossível fazer *ping* desde o servidor até máquinas nas subredes dos departamentos B e C.

1. **Adicione as rotas estáticas necessárias para restaurar a conectividade para o servidor, por forma a contornar a restrição imposta em c). Utilize para o efeito o comando *route* *add* e registe os comandos que usou.**

Tentámos executar os comandos:

* route ADD 10.0.6.0 MASK 255.255.255.0 10.0.3.1 (para restaurar a conexão com a subrede do departamento B)
* route ADD 10.0.5.0 MASK 255.255.255.0 10.0.3.1 (para restaurar a conexão com a subrede do departamento C)

1. **Teste a nova política de encaminhamento garantindo que o servidor está novamente acessível, utilizando para o efeito o comando *ping*. Registe a nova tabela de encaminhamento do servidor.**

**3 – Definição de Subredes**

**Considere a topologia definida anteriormente. Assuma que o endereçamento entre os routers se mantém inalterado, contudo, o endereçamento em cada departamento deve ser redefinido.**

1. **Assumindo que dispõe apenas de um único endereço de rede IP classe C 192.168.128.0/24, defina um novo esquema de endereçamento para as redes dos departamentos (mantendo a rede de core inalterada) e atribua endereços às interfaces dos vários sistemas envolvidos. Deve justificar as opções usadas.**
2. **Qual a máscara de rede que usou (em formato decimal)? Justifique.**
3. **Quantos *hosts* IP pode interligar em cada departamento? Justifique.**
4. **Garanta que conectividade IP entre as várias redes locais da empresa MIEInet é mantida.**

**Conclusões**