



Protocolo IPv4 e IP

J.G¹, M.F², P.G.³, T.C.⁴

Licenciatura em Ciências da Computação (LCC)

Universidade do Minho, R. da Universidade, 4710-057 Braga
Licenciatura em Ciências da Computação
gci@reitoria.uminho.pt

¹ João Guedes - A94013

² Miguel Freitas - A91635

³ Pedro Gomes - A91647

⁴ Tomás Campinho - A91668

TP4: Protocolo IPv4 e IP

TP4: Protocolo IPv4 (Parte I – Respostas)

Captura de tráfego IP

1. Prepare uma topologia CORE para verificar o comportamento do traceroute.

Ligue um host n1 a um router n2, que se liga a um router n3 que, por sua vez, se liga a um host n4.

- a) Active o wireshark ou o tcpdump no nó 4. Numa shell de n4, execute o comando `traceroute -I` para o endereço IP do n1.

```
root@n4:/tmp/pycore.45400/n4.conf# traceroute -I 10.0.0.10
Specify "host" missing argument.
root@n4:/tmp/pycore.45400/n4.conf# traceroute 10.0.0.10
traceroute to 10.0.0.10 (10.0.0.10), 30 hops max, 60 byte packets
 1 10.0.2.1 (10.0.2.1)  0.175 ms  0.011 ms  0.006 ms
 2 10.0.1.1 (10.0.1.1)  0.034 ms  0.022 ms  0.015 ms
 3 * * *
 4 * * *
 5 * * *
 6 * * *
 7 * * *
 8 * 10.0.0.10 (10.0.0.10)  0.053 ms  0.011 ms
```

Como se pode observar na figura 2, os pacotes passam por 2 routers, cujos IPs das interfaces ativas de cada um são, respetivamente, 10.0.0.10 e 10.0.2.1, até chegarem ao destino cujo IP da sua interface ativa é 10.0.1.1 .

TP4: Protocolo IPv4 e IP

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

```
10.0.1.1 > 224.0.0.5: OSPFv2, Hello, length 48
Router-ID 10.0.0.1, Backbone Area, Authentication Type: none (0)
Options [External]
Hello Timer 10s, Dead Timer 40s, Mask 255.255.255.0, Priority 1
Designated Router 10.0.1.2, Backup Designated Router 10.0.1.1
Neighbor List:
10.0.1.2
16:06:47.757732 IP (tos 0xc0, ttl 1, id 18272, offset 0, flags [none], proto OSPF (89), length 68)
10.0.1.2 > 224.0.0.5: OSPFv2, Hello, length 48
Router-ID 10.0.1.2, Backbone Area, Authentication Type: none (0)
Options [External]
Hello Timer 10s, Dead Timer 40s, Mask 255.255.255.0, Priority 1
Designated Router 10.0.1.2, Backup Designated Router 10.0.1.1
Neighbor List:
10.0.0.1
16:06:47.782998 IP6 (class 0xc0, hlim 1, next-header OSPF (89) payload length: 40) fe80::200:ff:feaa:3 > ff02::5: OSPFv3, Hello, length 40
Router-ID 10.0.1.2, Backbone Area
Options [V6, External, Router]
Hello Timer 10s, Dead Timer 40s, Interface-ID 0.0.0.243, Priority 1
Designated Router 10.0.1.2, Backup Designated Router 10.0.0.1
Neighbor List:
10.0.0.1
16:06:47.863642 IP6 (class 0xc0, hlim 1, next-header OSPF (89) payload length: 40) fe80::200:ff:feaa:2 > ff02::5: OSPFv3, Hello, length 40
Router-ID 10.0.0.1, Backbone Area
Options [V6, External, Router]
Hello Timer 10s, Dead Timer 40s, Interface-ID 0.0.0.241, Priority 1
Designated Router 10.0.1.2, Backup Designated Router 10.0.0.1
Neighbor List:
10.0.1.2
16:06:52.465188 IP (tos 0x0, ttl 1, id 13977, offset 0, flags [none], proto UDP (17), length 60)
10.0.2.10.36510 > 10.0.0.10.33456: UDP, length 32
16:06:52.465215 IP (tos 0xc0, ttl 63, id 2742, offset 0, flags [none], proto ICMP (1), length 88)
```

ICMP verifica-se quando dois aparelhos numa rede comunicam um entre outro, este protocolo pertence à camada de rede que visa diagnosticar erros de ligação. [Referência]

Como predefinição o traceroute envia 3 datagramas com o mesmo TTL pelo que capturamos 3 vezes mais mensagens ICMP.

Neste caso verifica-se se é utilizado para verificar se o pedido pretendido chega ao destino. Como visto em A, avança todos os aparelhos, por fim chegando ao destino.

Este resultado era esperado pois “traceroute” é um processo dinâmico que permite descobrir a rota até ao destino.

TP4: Protocolo IPv4 e IP

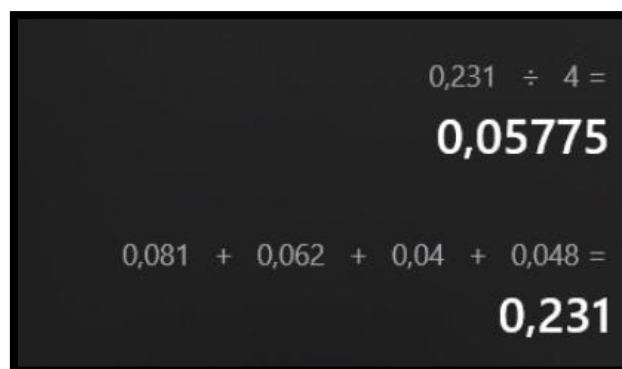
c) Qual deve ser o valor inicial mínimo do campo TTL para alcançar o destino n1?

TTL (time-to-live) indica durante quanto tempo um registo será guardado no servidor DNS. O valor mínimo do campo TTL para alcançar o destino s4 deve ser 3, uma vez que o TTL é decrementado aquando da passagem em cada router. Quando utilizamos o comando “traceroute” verificamos que quando o TTL é usado a 3 consegue atingir o destino.

d) Qual o tempo médio de ida-e-volta (RTT - round-trip time) obtido?

```
time=0,081 ms  
time=0,062 ms  
time=0,040 ms  
time=0,048 ms
```

Como se pode verificar pela figura, os valores do RoundTrip Time (rtt), é o valor correspondente ao valor médio, sendo assim o valor médio é de que é 0,05775 milisegundos.



The image shows a handwritten calculation on a dark background. At the top, the sum of four RTT values is calculated: $0,081 + 0,062 + 0,04 + 0,048 = 0,231$. Below this, the average is calculated by dividing the sum by 4: $0,231 \div 4 = 0,05775$. The final result, 0,05775, is displayed in a larger font.

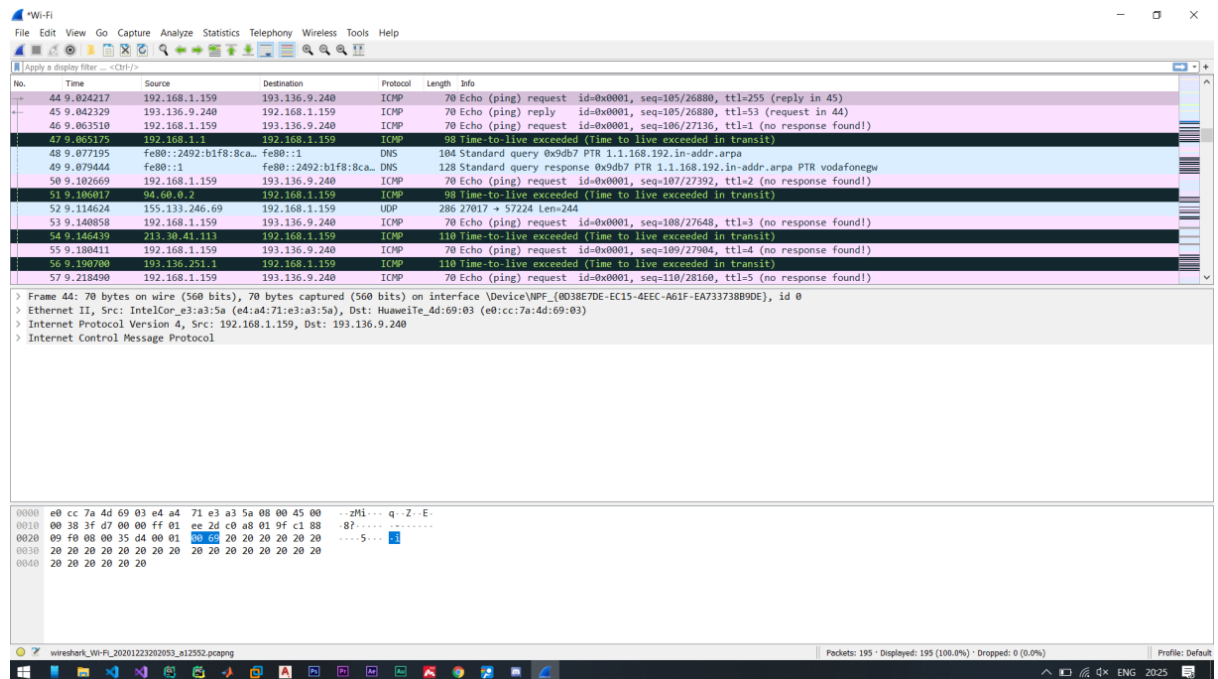
$$0,081 + 0,062 + 0,04 + 0,048 = 0,231$$
$$0,231 \div 4 = 0,05775$$

Este valor que foi calculado é um valor estimado. Como grupo denotamos que vão existir erros associados a este valor, uma vez que a amostra que retiramos pouco extensiva. De modo a diminuir o erro associado a este cálculo devemos repetir mais vezes o ensaio.

TP4: Protocolo IPv4 e IP

2. Pretende-se agora usar o traceroute na sua máquina nativa, e gerar datagramas IP de diferentes tamanhos.

O programa tracert disponibilizado no Windows não permite mudar o tamanho das mensagens. Como alternativa, o grupo utilizou o programa pingplotter na sua versão livre ou shareware (<http://www.pingplotter.com>) para efetuar traceroute.



a) Qual é o endereço IP da interface ativa do seu computador?

- ✓ Internet Protocol Version 4, Src: 192.168.1.159, Dst: 193.136.9.240
 - 0100 = Version: 4
 - 0101 = Header Length: 20 bytes (5)
 - > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
 - Total Length: 56
 - Identification: 0x3fd7 (16343)
 - > Flags: 0x0000
 - Fragment offset: 0
 - Time to live: 255
 - Protocol: ICMP (1)
 - Header checksum: 0xee2d [validation disabled]
 - [Header checksum status: Unverified]
 - Source: 192.168.1.159
 - Destination: 193.136.9.240
 - > Internet Control Message Protocol

TP4: Protocolo IPv4 e IP

```
-----  
Source: 192.168.1.159  
Destination: 193.136.9.240
```

Selecionando a primeira mensagem ICMP capturada é possível observar a secção do IPv4 com o endereço IP da interface ativa do nosso computador, 192.168.1.159.

Concluimos que é este o endereço pois na secção do ICMP é possível visualizar que esta mensagem possui tipo 8 que descreve a mensagem de echo request. Este protocolo ICMP serve para testar a **conectividade entre equipamentos**. Por isso, de acordo com estas referências sabemos que o nosso computador é o Source desta mensagem.

b) Qual é o valor do campo protocolo? O que identifica?

O valor do campo protocolo é ICMP (1). Este campo identifica o protocolo usado para a transmissão do datagrama. Neste caso, o ICMP (Internet Control Message Protocol) é usado para testar a conectividade entre a origem e o destino.

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

```
> Frame 44: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \Device\NPF_{0D38E7DE-EC15-4EEC-A61F-EA733738B9DE}, id 0  
> Ethernet II, Src: IntelCor_e3:a3:5a (e4:a4:71:e3:a3:5a), Dst: HuaweiTe_4d:69:03 (e0:cc:7a:4d:69:03)  
v Internet Protocol Version 4, Src: 192.168.1.159, Dst: 193.136.9.240  
  0100 .... = Version: 4  
  .... 0101 = Header Length: 20 bytes (5)
```

Os bytes do cabeçalho IP(v4) podem ser visualizados, neste caso 20 bytes. O payload é parte principal dos dados transmitidos. Os bytes do payload são calculados subtraindo o número de bytes totais do datagrama (Total length, que podemos verificar na imagem do exercício anterior) pelo número de bytes do cabeçalho antes mencionado. Portanto, o número de bytes do payload são $(56 - 20 =) 36$ bytes.

TP4: Protocolo IPv4 e IP

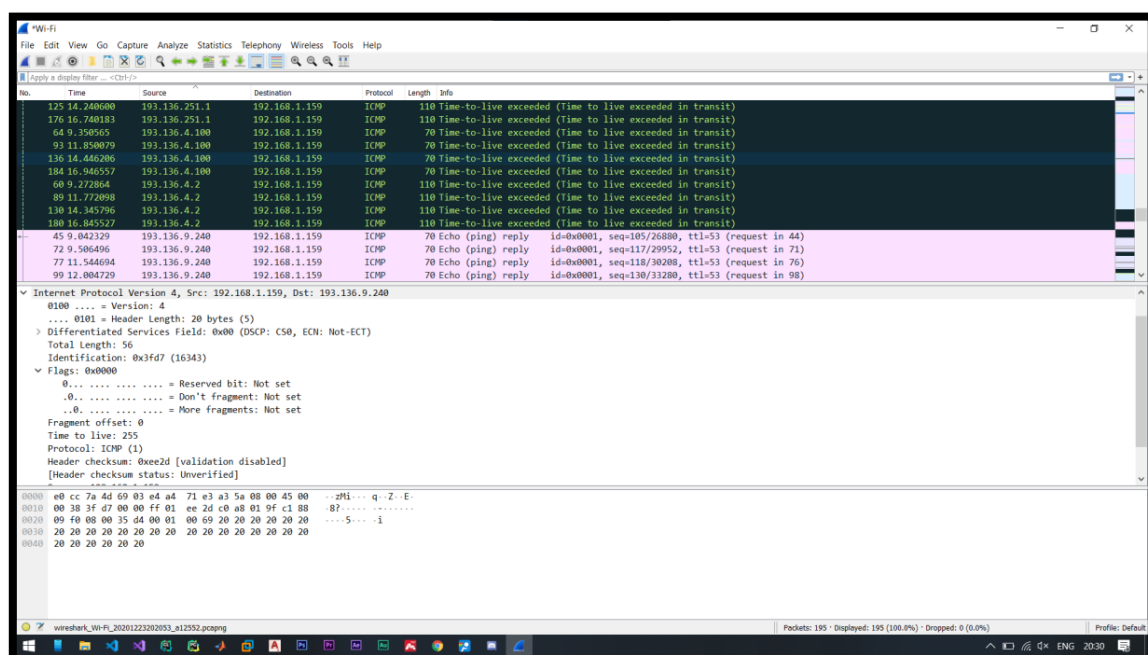
d) O datagrama IP foi fragmentado? Justifique.

```
Internet Protocol Version 4, Src: 192.168.1.159, Dst: 193.136.9.240
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 56
    Identification: 0x3fd7 (16343)
  < Flags: 0x0000
    0... .. = Reserved bit: Not set
    .0.. .. = Don't fragment: Not set
    ..0. .. = More fragments: Not set
    Fragment offset: 0
```

Como se pode observar, não se verificam ações de fragmentação do datagrama. Se existisse fragmentação podíamos ver ações consecutivas com datagramas fragmentados então sendo assim vemos que o parâmetro “More fragments” não é atribuído.

Tendo isto em conta vemos que também o parâmetro offset do fragmento está a 0.

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 com base no IP gerado na sua máquina. Observa algum padrão nos valores do campo de Identificação do datagrama IP e TTL?



TP4: Protocolo IPv4 e IP

180 16.845527	193.136.4.2	192.168.1.159	ICMP	110 Time-to-live exceeded (Time to live exceeded in transit)	
45 9.042329	193.136.9.240	192.168.1.159	ICMP	70 Echo (ping) reply	id=0x0001, seq=105/26880, ttl=53 (request in 44)
72 9.506496	193.136.9.240	192.168.1.159	ICMP	70 Echo (ping) reply	id=0x0001, seq=117/29952, ttl=53 (request in 71)
77 11.544694	193.136.9.240	192.168.1.159	ICMP	70 Echo (ping) reply	id=0x0001, seq=118/30208, ttl=53 (request in 76)
99 12.004729	193.136.9.240	192.168.1.159	ICMP	70 Echo (ping) reply	id=0x0001, seq=130/33280, ttl=53 (request in 98)
Internet Protocol Version 4, Src: 193.136.4.2, Dst: 192.168.1.159					
0100 = Version: 4					
.... 0101 = Header Length: 20 bytes (5)					
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)					
Total Length: 96					
Identification: 0x7a46 (31302)					
Flags: 0x0000					
0... .. = Reserved bit: Not set					
.0.. .. = Don't fragment: Not set					
..0. = More fragments: Not set					

45 9.042329	193.136.9.240	192.168.1.159	ICMP	70 Echo (ping) reply	id=0x0001, seq=105/26880, ttl=53 (request in 44)
72 9.506496	193.136.9.240	192.168.1.159	ICMP	70 Echo (ping) reply	id=0x0001, seq=117/29952, ttl=53 (request in 71)
77 11.544694	193.136.9.240	192.168.1.159	ICMP	70 Echo (ping) reply	id=0x0001, seq=118/30208, ttl=53 (request in 76)
99 12.004729	193.136.9.240	192.168.1.159	ICMP	70 Echo (ping) reply	id=0x0001, seq=130/33280, ttl=53 (request in 98)
Internet Protocol Version 4, Src: 193.136.9.240, Dst: 192.168.1.159					
0100 = Version: 4					
.... 0101 = Header Length: 20 bytes (5)					
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)					
Total Length: 56					
Identification: 0xf40d (62477)					
Flags: 0x0000					
0... .. = Reserved bit: Not set					
.0.. .. = Don't fragment: Not set					
..0. = More fragments: Not set					
Fragment offset: 0					
Time to live: 53					

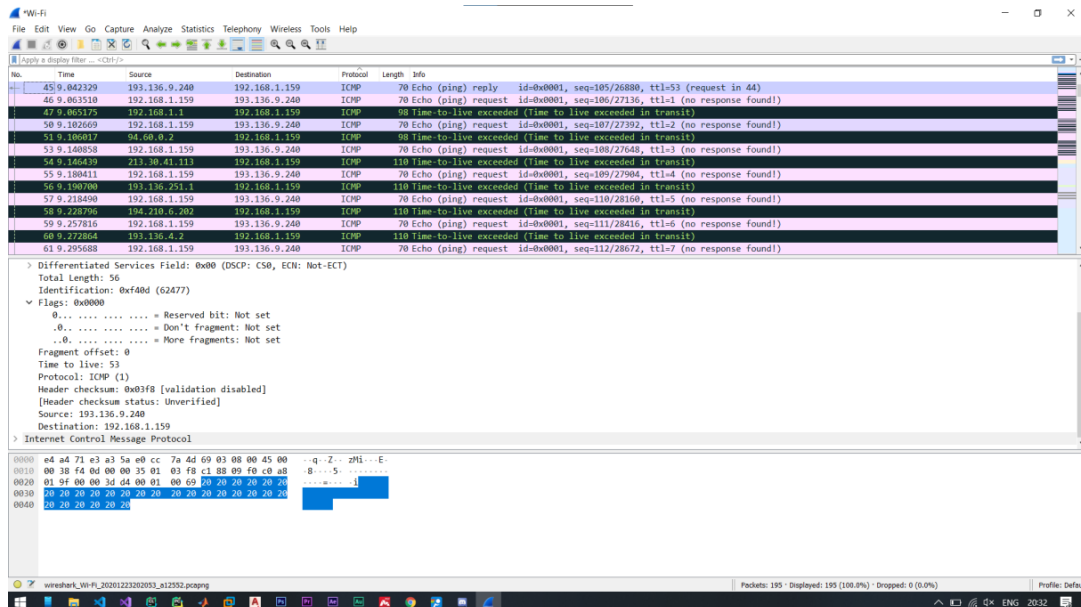
Podemos concluir que os campos do cabeçalho IP que variam de pacote para pacote são o campo *Identificação*, o campo *Header Checksum* e o campo *Time-to-live* (apesar de no print não conseguir identificar este campo, o grupo verificou que sim, existiam mudanças) na secção do IP.

À pergunta: “**Observa algum padrão nos valores do campo de Identificação do datagrama IP e TTL?**” **Sim** o grupo identifica alguns padrões na variação dos valores como por exemplo:

- O campo de identificação do datagrama IP aumenta uma unidade por cada datagrama enviado pelo nosso computador.
- Relativamente ao campo TTL, como usamos o comando traceroute padrão, ele envia 3 datagramas com o mesmo TTL pelo que o TTL aumenta uma unidade em cada 3 datagramas enviados pelo nosso computador.

TP4: Protocolo IPv4 e IP

- f) A seguir (com os pacotes ordenados por endereço destino) encontre a série de respostas ICMP TTL exceeded enviadas ao seu computador pelo primeiro router. Qual é o valor do campo TTL? Esse valor permanece constante para todas as mensagens de resposta ICMP TTL exceeded enviadas? Porquê?



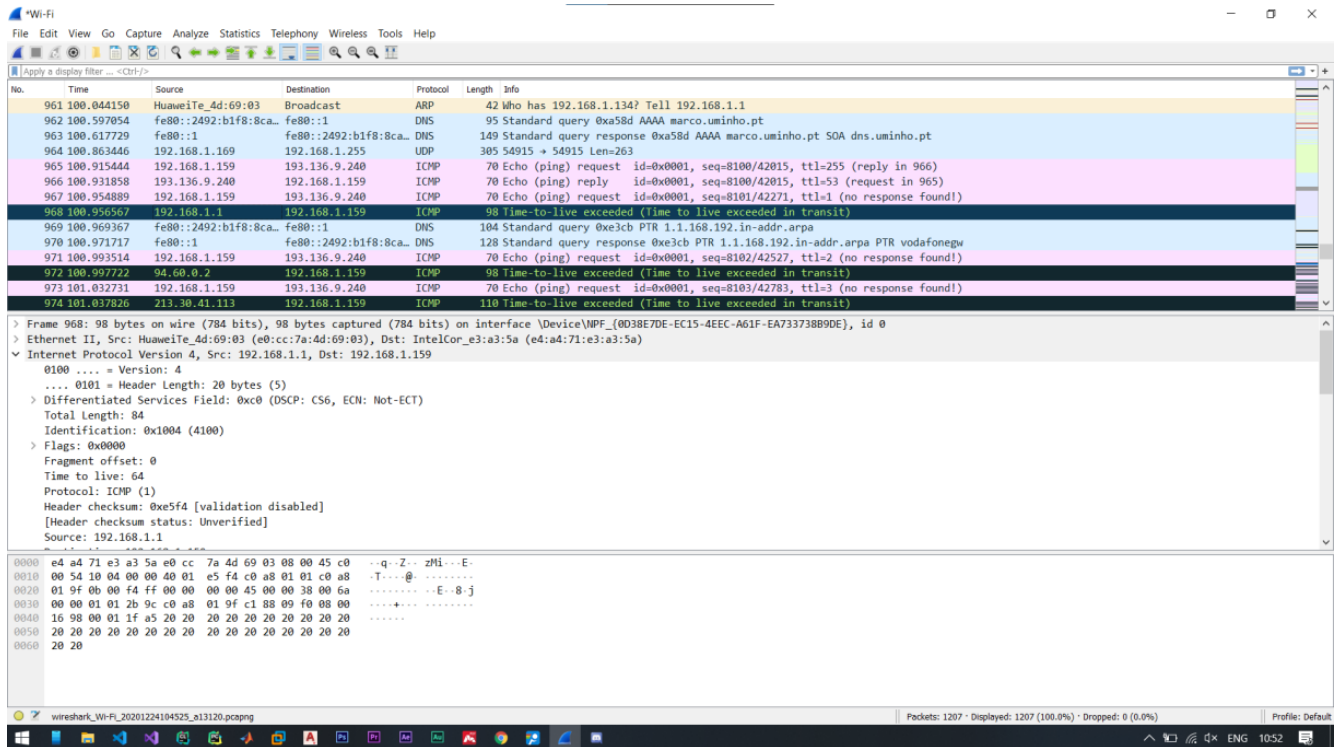
- Internet Protocol Version 4, Src: 192.168.1.159, Dst: 193.136.9.240
 - 0100 = Version: 4
 - 0101 = Header Length: 20 bytes (5)
 - Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
 - Total Length: 56
 - Identification: 0x3fd8 (16344)
 - Flags: 0x0000
 - 0... = Reserved bit: Not set
 - .0... = Don't fragment: Not set
 - ..0... = More fragments: Not set
 - Fragment offset: 0
 - Time to live: 1
 - Protocol: ICMP (1)
 - Header checksum: 0xec2d [validation disabled]
 - [Header checksum status: Unverified]

O primeiro valor do campo TTL é 1, devia-se manter enquanto a fonte for a mesma.

Contudo o grupo não verificou, não sabemos o que desencadeou este erro. Por isso não conseguimos responder com precisão. Se este erro não fosse verificado, pela teoria iríamos conseguir verificar que quando a fonte variar este campo era decrementado uma unidade (estes valores iriam ser altos) uma vez que cada router devia assegurar que a mensagem ICMP chega ao seu destino. O facto de ser decrementado deve-se ao normal funcionamento de trânsito de datagramas entre routers, cada router diferente decrementa uma unidade ao TTL até este chegar ao destino.

TP4: Protocolo IPv4 e IP

3. 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 em 30XX bytes.



- a) Localize a primeira mensagem ICMP. A mensagem foi fragmentada? Porque é que houve (ou não) necessidade de o fazer?

Como podemos verificar na figura acima ilustrada não foi detetada nenhuma mensagem fragmentada uma vez que aparece “Fragment offset: 0”. Pois a quantidade máxima de dados que um frame da camada Física consegue transportar chama-se MTU (maximum transmission unit). Só conseguem carregar até 1500 bytes de dados, e como nós queríamos capturar tráfego para pacotes com 3028 bytes, então ter-se-ia que se dividir o pacote inicial em 3 fragmentos.

TP4: Protocolo IPv4 e IP

- 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 se trata do primeiro fragmento? Qual é o tamanho deste datagrama IP?

Não conseguimos encontrar nenhum fragmento do datagrama IP segmentado, então sendo assim e para não deixar o resto de todos os exercícios em branco decidimos como grupo discutir e levantar algumas hipóteses com outros dados para dar segmento ao exercício.

```
► Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 1500
  Identification: 0xa25a (41562)
▼ Flags: 0x2000, More fragments
  0... .. = Reserved bit: Not set
  .0.. .. = Don't fragment: Not set
  ..1. .... = More fragments: Set
  ...0 0000 0000 0000 = Fragment offset: 0
```

Como se pode verificar, sublinhado a azul escuro, existe uma flag no cabeçalho que indica a existência de mais fragmentos, se existem mais é porque o que está a ser analisado é um fragmento. Trata-se do primeiro fragmento uma vez que o campo sublinhado a preto “Fragment offset”, indica o offset do datagrama e este está a 0. Total length corresponde ao tamanho do datagrama que é 1500 bytes

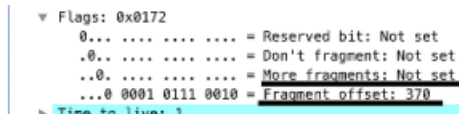
- c) Imprima o segundo fragmento do datagrama IP original. Que informação do cabeçalho IP indica que não se trata do 1º fragmento? Há mais fragmentos? O que nos permite afirmar isso?

```
▼ Flags: 0x20b9, More fragments
  0... .. = Reserved bit: Not set
  .0.. .. = Don't fragment: Not set
  ..1. .... = More fragments: Set
  ...0 0000 1011 1001 = Fragment offset: 185
```

Uma vez que o Fragment offset é 185, logo é diferente de 0 pelo que não se trata do primeiro fragmento. Existem mais fragmentos. No campo de more fragments está selecionado como Set, logo é verdadeira que existem mais fragmentos.

TP4: Protocolo IPv4 e IP

- d) Quantos fragmentos foram criados a partir do datagrama original? Como se detecta o último fragmento correspondente ao datagrama original?



```
▼ Flags: 0x0172
 0... .. = Reserved bit: Not set
.0... .. = Don't fragment: Not set
..0... .. = More fragments: Not set
...0 0001 0111 0010 = Fragment offset: 370
```

Foram criados 3 fragmentos, esta é informação que é fornecida pelo Wireshark e não está presente no cabeçalho IP. O campo More fragments não está assinalado e o campo Fragment offset é 370, que é logicamente diferente de 0.

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

Uma vez que existem 3 fragmentos relativos ao datagrama original podemos comparar os primeiros dois devido às suas semelhanças e por fim comparar as principais diferenças de um dos dois primeiros fragmentos para com o último fragmento.

Comparando os dois primeiros fragmentos, podemos notar que as principais diferenças estão nos campos Fragment offset e no Header checksum.

Comparando o segundo fragmento com o último, as principais diferenças estão nos campos Total Length, More fragments, Fragment offset e no Header checksum. O campo Identification é igual em todos os fragmentos pelo que no destino é possível saber quais dos fragmentos vão originar um datagrama.

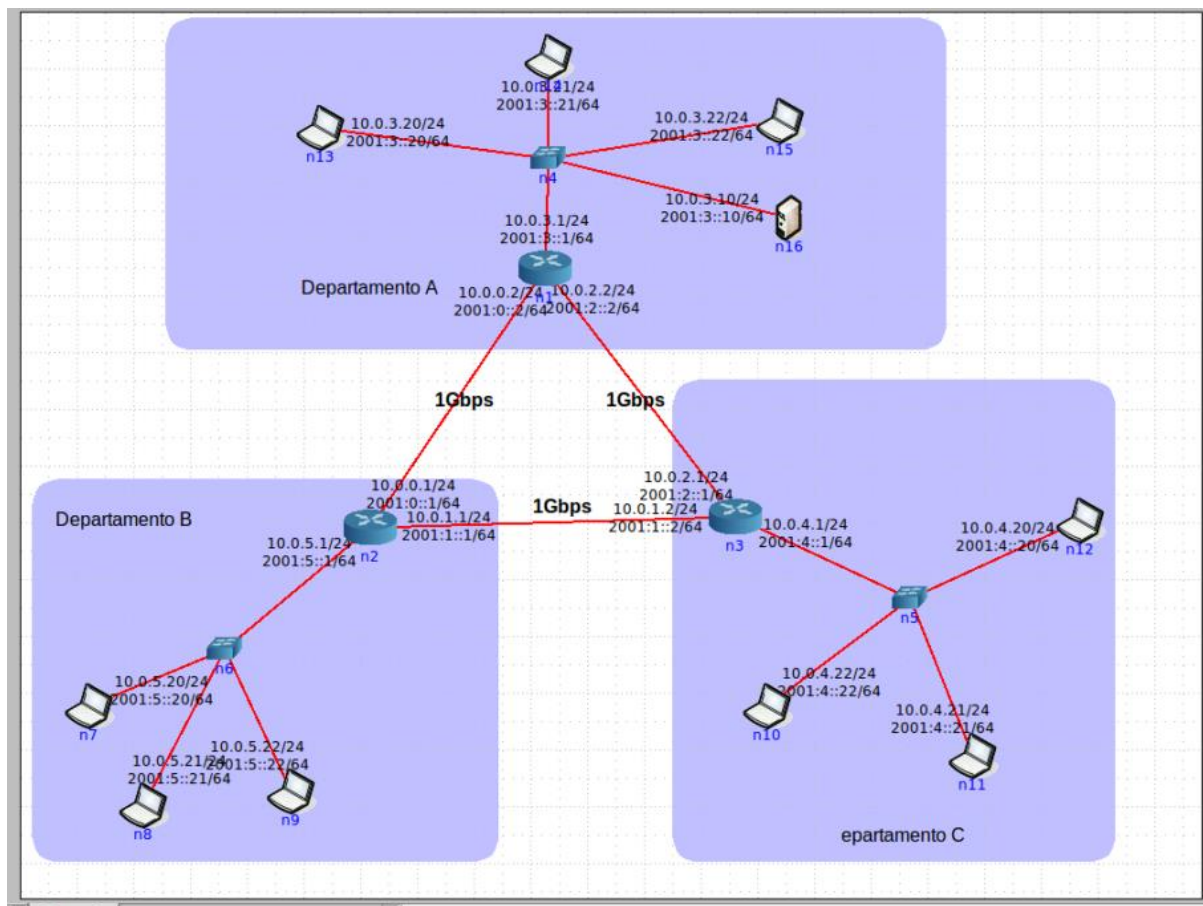
TP4: Protocolo IPv4 e IP

TP4: Protocolo IPv4 (Parte II – Respostas)

Endereçamento e Encaminhamento IP

1. Atenda aos endereços IP atribuídos automaticamente pelo CORE aos diversos equipamentos da topologia.

- a) 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.



TP4: Protocolo IPv4 e IP

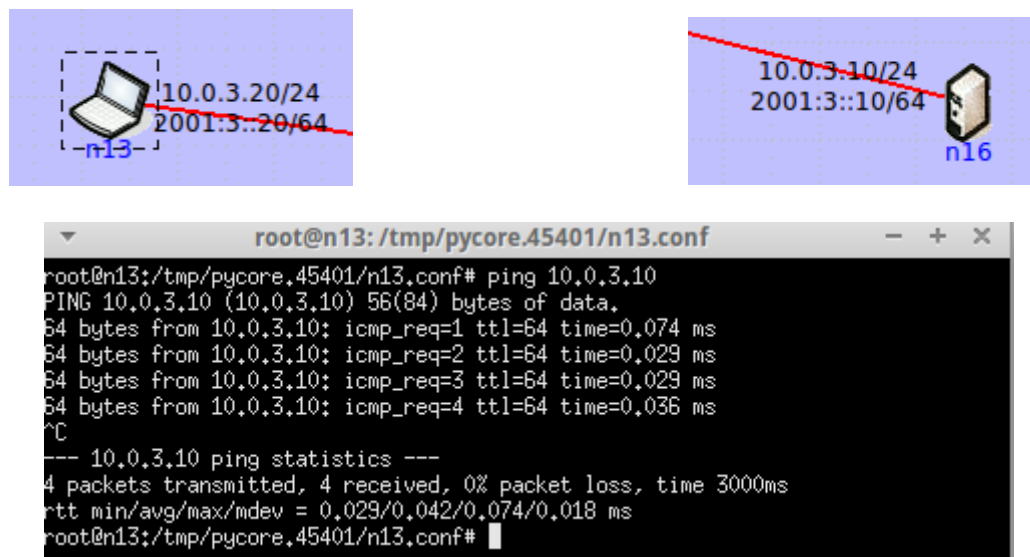
b) Trata-se de endereços públicos ou privados? Porquê?

De acordo com RFC1918, estes espaços de endereçamento são reservados para internets privadas 10.0.0.0 - 10.255.255.255. Como se pode observar, todos os equipamentos estão endereçados dentro deste espaço, conclui-se então que os endereços são privados

c) Quantos bytes tem o campo de dados (payload) do datagrama? Como se calcula o tamanho do payload? Porque razão não é atribuído um endereço IP aos switches?

Um endereço IP é um requisito para a camada de rede (nível 3), os switches trabalham estritamente na camada link (nível 2) e, portanto, não precisam de um endereço IP.

d) 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).



Existe conectividade entre os laptops e o servidor do Departamento A, podemos verificar isso na figura 27 onde conectamos o os laptops **n13** e **n16** (10.0.3.10).

Também existe conectividade entre os laptops departamento C também tem conectividade com o servidor (n16) e consecutivamente os laptops do departamento B também.

TP4: Protocolo IPv4 e IP

2. Atenda aos endereços IP atribuídos automaticamente pelo CORE aos diversos equipamentos da topologia.

- a) 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`).

```
root@n13:/tmp/pycore.45401/n13.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
0.0.0.0 10.0.3.1 0.0.0.0 UG 0 0 0 eth0
10.0.3.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
root@n13:/tmp/pycore.45401/n13.conf#
```

N13

```
root@n1:/tmp/pycore.45401/n1.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.0.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
10.0.1.0 10.0.0.1 255.255.255.0 UG 0 0 0 eth0
10.0.2.0 0.0.0.0 255.255.255.0 U 0 0 0 eth1
10.0.3.0 0.0.0.0 255.255.255.0 U 0 0 0 eth2
10.0.4.0 10.0.2.1 255.255.255.0 UG 0 0 0 eth1
10.0.5.0 10.0.0.1 255.255.255.0 UG 0 0 0 eth0
root@n1:/tmp/pycore.45401/n1.conf#
```

N1

Pode-se ver a tabela de encaminhamento do laptop **n13**, e a tabela de encaminhamento do router **n1**.

Na tabela de encaminhamento do laptop **n13**, os únicos endereços de saída são o respetivo router do departamento por onde saem todos os packets destinados a outros departamentos ou outras redes, e no caso de querer comunicar com laptops no próprio departamento, os datagramas serão entregues na interface de endereço próprio que corresponde ao gateway 0.0.0.0, sendo isto feito com auxílio do Ethernet Switch.

Na tabela de encaminhamento do router do departamento A, dependendo para onde se quer enviar os datagramas usamos endereços de saída diferentes. No caso de querer enviar datagramas para o departamento B e o departamento C usa os endereços. No caso de se querer comunicar com as redes adjacentes usa-se a interface de endereço default (0.0.0.0).

TP4: Protocolo IPv4 e IP

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

```
root@n13:/tmp/pycore.45401/n13.conf# ps -ef
UID      PID  PPID  C STIME TTY          TIME CMD
root      1    0    0 11:21 ?        00:00:00 /usr/sbin/vnoded -v -c /tmp/pyco
root      74    1    0 11:26 pts/7    00:00:00 /bin/bash
root     128    74    0 11:27 pts/7    00:00:00 ps -ef
root@n13:/tmp/pycore.45401/n13.conf#
```

```
root@n1:/tmp/pycore.45401/n1.conf# ps -ef
UID      PID  PPID  C STIME TTY          TIME CMD
root      1    0    0 11:21 ?        00:00:00 /usr/sbin/vnoded -v -c /tmp/pyco
root      62    1    0 11:21 ?        00:00:00 /usr/local/sbin/zebra -u root -g
root      71    1    0 11:21 ?        00:00:00 /usr/local/sbin/ospf6d -u root -
root      72    1    0 11:21 ?        00:00:00 /usr/local/sbin/ospfd -u root -g
root     132    1    6 11:28 pts/7    00:00:00 /bin/bash
root     186   132    0 11:28 pts/7    00:00:00 ps -ef
root@n1:/tmp/pycore.45401/n1.conf#
```

Este sistema possui encaminhamento dinâmico e estático. Na topologia de rede em anel (Router A - Router B - Router C) o encaminhamento é dinâmico. Pois se existir uma falha entre os routers, vão tentar adaptar um novo caminho.

Podemos verificar na primeira imagem correspondente ao **n13** que apenas temos 3 processos a decorrer na máquina: o processo-pai, a bash e o comando ps.

Podemos verificar na segunda imagem que corresponde ao **n1** temos 6 processos a decorrer na máquina: o processo-pai, a bash, o comando ps e 3 deamons, 2 dois quais são protocolos de roteamento para IPv4 e IPv6. **Concluimos que nos routers existe encaminhamento dinâmico, e nos laptops encaminhamento estático.**

TP4: Protocolo IPv4 e IP

- c) 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.

```
root@n16:/tmp/pycore.45401/n16.conf# route del -net 0.0.0.0
root@n16:/tmp/pycore.45401/n16.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.3.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
root@n16:/tmp/pycore.45401/n16.conf#
```



```
root@n13:/tmp/pycore.45401/n13.conf# ping 10.0.0.10
PING 10.0.0.10 (10.0.0.10) 56(84) bytes of data.
From 10.0.3.1 icmp_seq=1 Destination Host Unreachable
From 10.0.3.1 icmp_seq=2 Destination Host Unreachable
From 10.0.3.1 icmp_seq=3 Destination Host Unreachable
From 10.0.3.1 icmp_seq=4 Destination Host Unreachable
From 10.0.3.1 icmp_seq=5 Destination Host Unreachable
From 10.0.3.1 icmp_seq=6 Destination Host Unreachable
^C
--- 10.0.0.10 ping statistics ---
7 packets transmitted, 0 received, +6 errors, 100% packet loss, time 6012ms
pipe 4
root@n13:/tmp/pycore.45401/n13.conf#
```

O comando `route delete` no servidor **n16**, assim como a tabela de encaminhamento antes e depois do sucedido, que comprova o sucesso do comando realizado.

Ao retirar a rota por defeito da tabela de encaminhamento do servidor **n16**, os utilizadores da empresa que por norma acedem ao servidor deixam de poder fazer tal acesso, pois foi retirado o endereço de saída 10.0.3.1 com que o servidor se ligava ao switch que por sua vez se ligava ao Router A.

Sempre que os utilizadores tentam comunicar com o servidor **n16**, enviam os seus packets mas não recebem resposta do servidor uma vez que este já não tem como comunicar com o exterior do departamento A, logo a rota não é definida e a comunicação não é estabelecida. Um exemplo disso é o teste de conexão do laptop **n13** ao servidor **n16**. O servidor apenas consegue conectar com laptops dentro do departamento C uma vez que estes tem o endereço que combina com 10.0.3.0.

TP4: Protocolo IPv4 e IP

- d) 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 registre os comandos que usou.

```
root@n16:/tmp/pycore.45401/n16.conf# route add default gw 10.0.3.1 eth0
root@n16:/tmp/pycore.45401/n16.conf# netstat -rn
Kernel IP routing table
Destination      Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0          10.0.3.1       0.0.0.0         UG        0 0        0 eth0
10.0.3.0         0.0.0.0        255.255.255.0   U        0 0        0 eth0
root@n16:/tmp/pycore.45401/n16.conf#
```

A utilização do comando Route Add no servidor **n16** está representado na figura acima. Assim comparando a tabela de encaminhamento do exercício anterior e a deste exercício podemos ver que o comando é realizado com sucesso.

- e) 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.

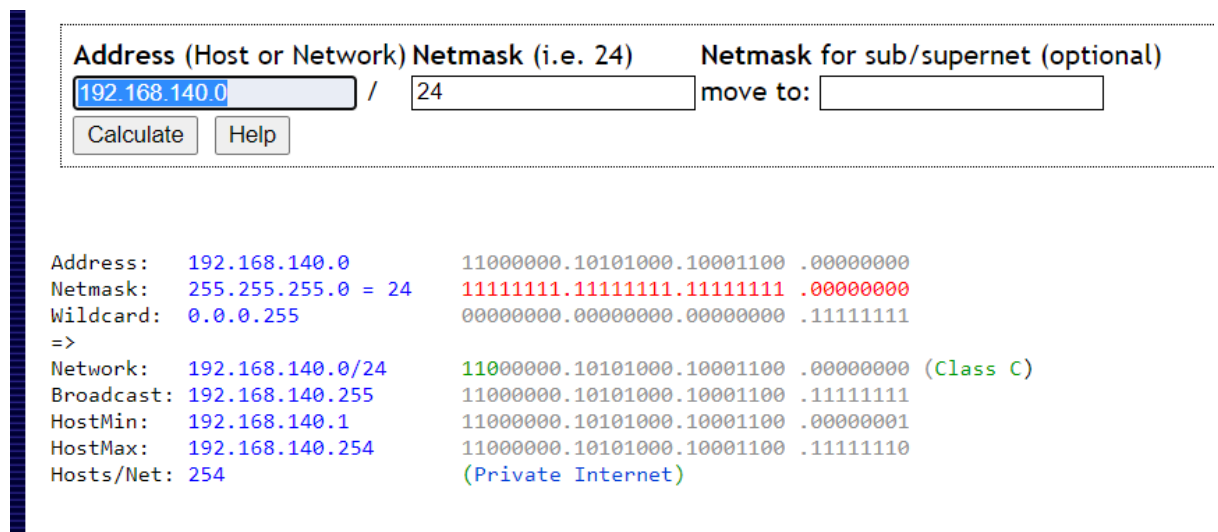
```
root@n13:/tmp/pycore.45401/n13.conf# ping 10.0.3.10
PING 10.0.3.10 (10.0.3.10) 56(84) bytes of data:
64 bytes from 10.0.3.10: icmp_req=1 ttl=64 time=0.063 ms
64 bytes from 10.0.3.10: icmp_req=2 ttl=64 time=0.031 ms
64 bytes from 10.0.3.10: icmp_req=3 ttl=64 time=0.033 ms
64 bytes from 10.0.3.10: icmp_req=4 ttl=64 time=0.092 ms
64 bytes from 10.0.3.10: icmp_req=5 ttl=64 time=0.049 ms
64 bytes from 10.0.3.10: icmp_req=6 ttl=64 time=0.051 ms
64 bytes from 10.0.3.10: icmp_req=7 ttl=64 time=0.027 ms
64 bytes from 10.0.3.10: icmp_req=8 ttl=64 time=0.052 ms
64 bytes from 10.0.3.10: icmp_req=9 ttl=64 time=0.030 ms
64 bytes from 10.0.3.10: icmp_req=10 ttl=64 time=0.032 ms
64 bytes from 10.0.3.10: icmp_req=11 ttl=64 time=0.060 ms
64 bytes from 10.0.3.10: icmp_req=12 ttl=64 time=0.066 ms
^C
--- 10.0.3.10 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11003ms
rtt min/avg/max/mdev = 0.027/0.048/0.092/0.020 ms
root@n13:/tmp/pycore.45401/n13.conf#
```

A nova política de encaminhamento funciona corretamente e que o servidor está novamente acessível, através do teste de conectividade com o laptop de outro departamento, neste caso o laptop 10.0.3.10 do departamento A. Logo na figura acima podemos ver que foi criada a nova tabela de encaminhamento do servidor **n16**.

TP4: Protocolo IPv4 e IP

Definição de Sub-redes

1. Assumindo que dispõe apenas de um único endereço de rede IP classe C 192.168.140.0/24, 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.



The screenshot shows a network calculator interface. At the top, there are input fields for 'Address (Host or Network)' with the value '192.168.140.0', 'Netmask (i.e. 24)' with the value '24', and 'Netmask for sub/supernet (optional)' with a 'move to:' field. Below these are 'Calculate' and 'Help' buttons. The results section displays the following information:

Address:	192.168.140.0	11000000.10101000.10001100 .00000000
Netmask:	255.255.255.0 = 24	11111111.11111111.11111111 .00000000
Wildcard:	0.0.0.255	00000000.00000000.00000000 .11111111
=>		
Network:	192.168.140.0/24	11000000.10101000.10001100 .00000000 (Class C)
Broadcast:	192.168.140.255	11000000.10101000.10001100 .11111111
HostMin:	192.168.140.1	11000000.10101000.10001100 .00000001
HostMax:	192.168.140.254	11000000.10101000.10001100 .11111110
Hosts/Net:	254	(Private Internet)

De acordo com o que aprendemos nas fuckings aluas metemos o endereço mais alto no router e os restantes aparelhos pegando no endereço mais baixo.

2. Qual a máscara de rede que usou (em formato decimal)? Justifique

De acordo com os nossos dados (mostrados na figura acima), a mascara de rede que usou em formato decimal é: 255.255.255.0

3. Quantos hosts IP pode interligar em cada departamento? Justifique.

Podem interligar 254 hosts IP em cada departamento. Utilizamos um webservice para ter acesso a todos estes dados

TP4: Protocolo IPv4 e IP

4. Garanta que conectividade IP entre as várias redes locais da empresa LCCnet é mantida.

A conectividade IP é garantida através do uso de endereços ip estáticos externos diferentes dos endereços internos. Com isto simulamos comunicação exterior entre routers. Por isso a conectividade IP é mantida na empresa LCCnet.

TP4: Protocolo IPv4 e IP

Conclusão

Os resultados obtidos no desenvolver desta atividade de trabalho prático laboratorial, permitiram ao grupo compreender e consolidar questões relacionadas com o formato dos datagramas que são enviados entre uma origem e um destino com ajuda do comando traceroute. Na segunda fase tratamos de assuntos tais como o endereçamento e encaminhamento IP, e subnetting.

Neste mesmo estudo, o grupo apresentou algumas dificuldades em resolver principal o exercício 3 da parte I uma vez que não foram os resultados que esperávamos, e no desenvolver de algumas questões relacionadas com a parte II. No entanto, achamos que conseguimos cumprir todos os objetivos da atividade, sendo esta bem-sucedida, uma vez que conseguimos alcançar o principal objetivo da experiência.

Na nossa opinião, achamos que esta atividade decorreu mais ou menos como previsto, devido a todos estes dados serem coincidentes com todas as afirmações acima referidas, contribuindo para uma coerência entre a teoria e a prática.

Relembramos que todos os dados foram calculados de forma rigorosa, por isso pensamos que todos os erros que possam surgir possam ter prevenido da realização de capturas em wireshark em diferentes dias e diferentes lugares.

.

TP4: Protocolo IPv4 e IP

Referências

bharnden. (12 de setembro de 2020). *CORE Documentation*. Obtido de GitHub:

<https://github.com/coreemu/core/blob/master/docs/index.md>

networksorcery. (s.d.). *ARP, Address Resolution Protocol*. Obtido de networksorcery:

<http://www.networksorcery.com/enp/protocol/arp.htm>

Internetworking - Protocolo IP (Notas de Apoio das Aulas Teóricas) traceroute:

<http://tools.ietf.org/html/rfc2151> (seção 3.4)

Internet Protocol (IP): <http://tools.ietf.org/html/rfc791>

Internet Message Control Protocol (ICMP): <http://tools.ietf.org/html/rfc792>

Internetworking - Protocolo IP (Notas de Apoio das Aulas Teóricas)

Internet Protocol (IP): <http://tools.ietf.org/html/rfc791>