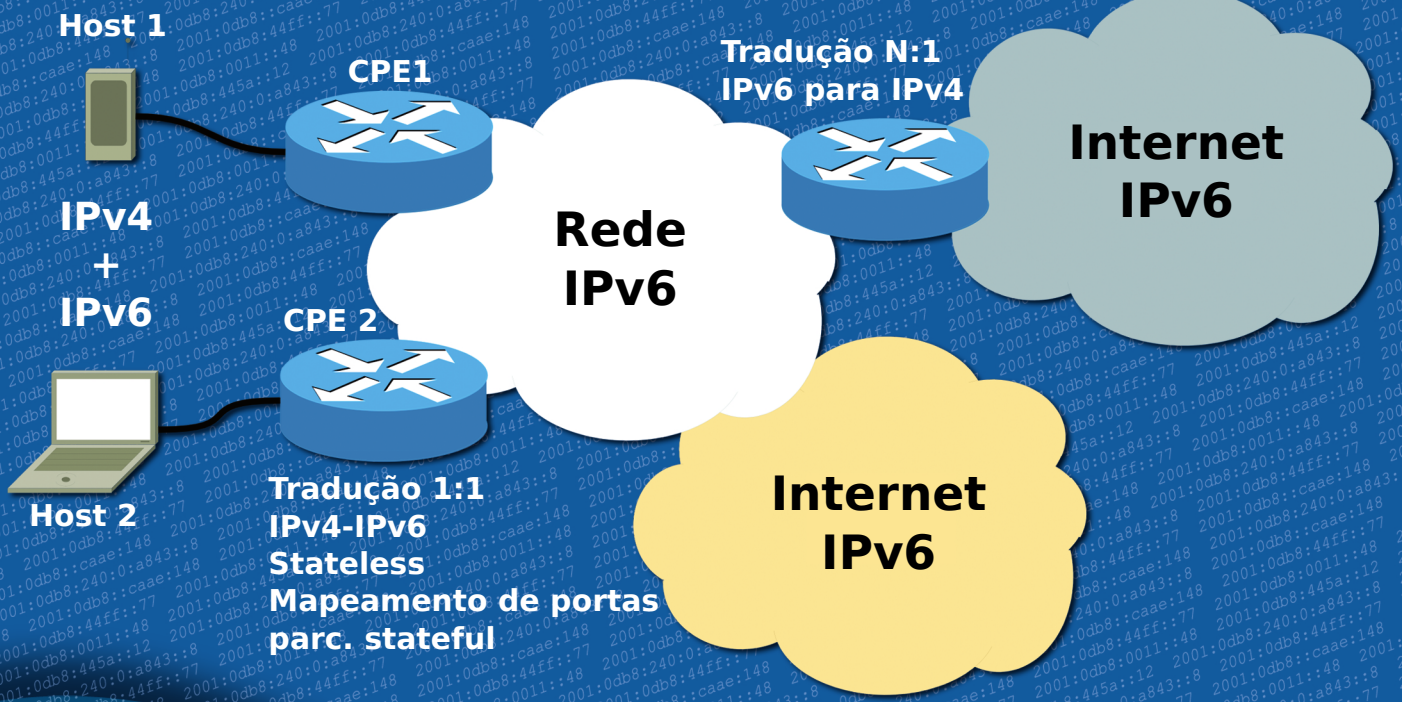
Roteiros para o laboratório



ceptro.br
Centro de Estudos e Pesquisas em Tecnologia de Redes e Operações

nic.br
Núcleo de Informação e Coordenação do Ponto BR

cgi.br
Comitê Gestor da Internet no Brasil

Índice

Transição

Exercícios de Endereçamento IPv6

1) Indicar a que tipo pertence cada um dos seguintes endereços:

Endereço	Tipo
2001:db8:cafe:f0ca:faca:2:3	
2804:1:2:b0ca:2c0:17ff:fe00:d1ca	
fe80::dad0:baba:ca00:a7a2	
fe80::2c0:17ff:fe00:d1ca	
2002:c8A0:79c::b010:de:c0c0	
::1	
fd00:ada:2345:b0ba::1	
ff0e::beba:d012:3:4	
ff05::baba:bebe:baba	

2) Comprimir ao máximo os seguintes endereços:

-2001:0db8:0000:1200:0fe0:0000:0000:0003
-2001:0db8::ca5a:0000:2000
-2001:0db8:face:b00c:0000:0000:0100:00ab

3) Descomprimir ao máximo os seguinte endereços:

-2001:db8:0:ca1::1:abcd
-2001:db8:4::2
-2001:db8:200::bdb:110

4) Utilizando o padrão EUI-64, crie endereços IPv6 a partir do prefixo 2001:db8:ba1a:d0ce::/64 baseados nos seguintes endereços MAC:

- 00:e0:4c:70:89:8d
- 5c:1d:e0:8c:e7:e7
- 07:00:27:00:e8:8b

5) Divida o prefixo 2001:db8::/32 na metade para que sejam gerados dois sub-prefixos.

_____
_____

6) Divida o prefixo 2001:db8:c000::/34 nos seguintes tamanhos:

- /35

_____
_____

- /36

_____
_____
_____
_____

7) Divida o prefixo 2001:db8:a000::/35 no seguinte tamanho:

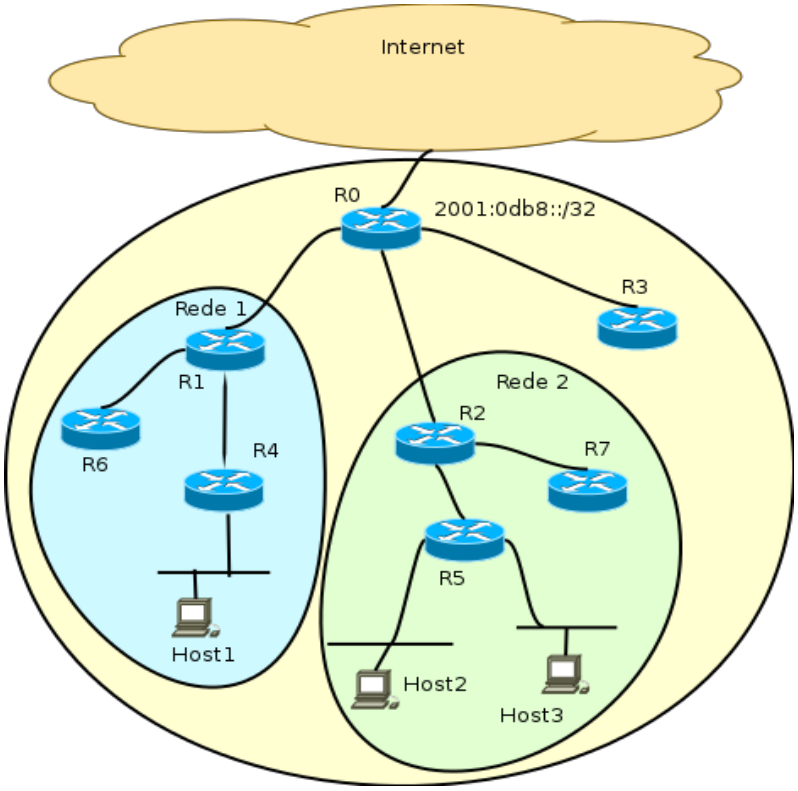
- /37

_____
_____
_____
_____

8) Você precisa subdividir o prefixo 2001:db8::/32 para atender a diversas subredes em sua organização. Para cada caso, diga qual é o prefixo mais curto que atende sua necessidade, o número de subredes geradas e o número de redes /64 possíveis em cada subrede. Diga também qual é o prefixo mais curto possível sem “cortar caracteres” (indo de 4 em 4 bits), o número de subredes geradas e o número de redes /64 possíveis em cada subrede.

necessidade	prefixo	no. subredes	redes /64		prefixo	no. subredes	redes /64
2 redes	/33	2	2^31		/36	16	2^28
56 redes							
1500 redes							
30000 redes							

9) A partir do prefixo 2001:0db8::/32, atribuir os prefixos às redes e computadores da organização ilustrada na figura a seguir:



Descrição	Endereço/Prefixo		Descrição	Endereço Prefixo
Infraestrutura rot.	/48		Rede 7 (R7)	/56
Gestão e Monit.	/48		Rede do Host1	/64
Rede 1 (R1)	/48		Rede do Host2	/64
Rede 2 (R2)	/48		Rede do Host3	/64
Rede 3 (R3)	/48		Host1	/64
Rede 4 (R4)	/56		Host2	/64
Rede 5 (R5)	/56		Host3	/64
Rede 6 (R6)	/56			

Referências

Vives, Alvaro. Practice - IPv6 Addressing. IPv6 Workshop, Bogotá, 2010, Consulintel.
http://www.6deploy.org/workshops/20100927_bogota_colombia/DIA1-3-PRACTICA-Direcciones-v0.1.pdf

IPV6 - Neighbor Discovery Protocol

Experiência 1 - Neighbor Solicitation e Neighbor Advertisement

Objetivo

Esta experiência tem como objetivo apresentar o funcionamento do mecanismo de resolução de endereços do protocolo Neighbor Discovery IPv6, através do estudo das mensagens de Neighbor Solicitation e Neighbor Advertisement.

Para o presente exercício será utilizada a topologia descrita no arquivo: **Funcionalidade-Neighbor-Discovery-E1.imn**.

Introdução Teórica

A resolução de endereços é um procedimento realizado pelos nós de uma rede para descobrir endereços físicos dos dispositivos vizinhos presentes no mesmo enlace que sua interface de rede.

O procedimento é iniciado quando um dispositivo tenta enviar um pacote cujo o endereço físico de destino é desconhecido. Nele, o nó solicitante irá enviar uma mensagem Neighbor Solicitation para todos os nós do enlace com endereço de destino IPv6 marcado como *Multicast Solicited-Node (FF02::1:FFXX:XXXX)* e o endereço a ser resolvido marcado no campo Target do cabeçalho ICMPv6. Além desses dados, o protocolo opcional Source Link-Layer Address também é enviado com intuito de fornecer o seu endereço físico ao destino e, assim, evitar novas trocas de mensagens.

O nó de destino, dono do IPv6 requisitado, ao receber esse pacote cria uma mensagem Neighbor Advertisement e a envia como resposta diretamente ao nó requisitante. O endereço de destino IPv6 dessa mensagem será marcado com o endereço de link local do nó requisitante e o seu endereço físico será utilizado para preencher o campo de Protocolo opcional *Target Link-Layer Address.*

Com a chegada dessa informação ao nó, o *neighbor cache* é atualizado, com o status de alcance e a comunicação pode ser começada.

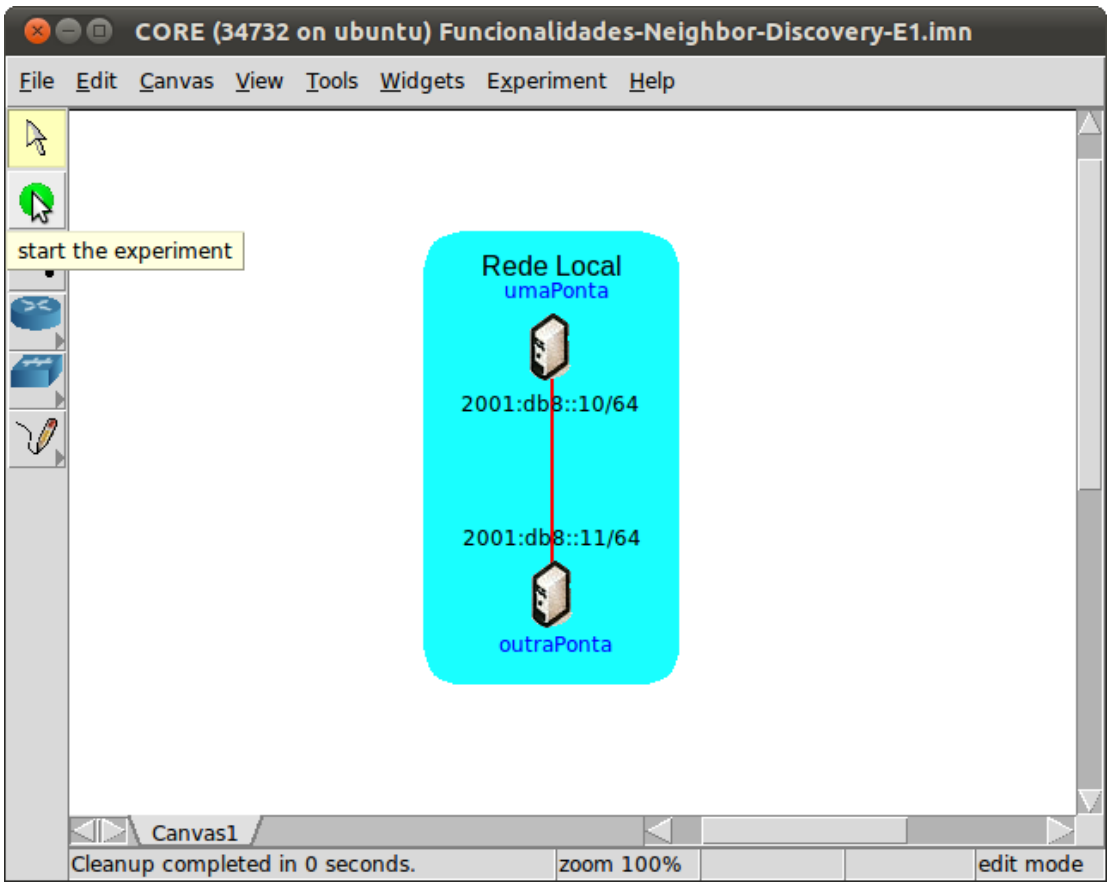
Roteiro Experimental

1. Caso não esteja utilizando a máquina virtual fornecida pelo NIC.br é preciso, antes de começar a experiência, instalar alguns softwares para auxiliar no aprendizado (caso contrário vá para o passo 2).
   Siga o passo a seguir para realizar a instalação:
   a. Para fazer algumas verificações durante o experimento será necessário a utilização do programa **Wireshark** que realiza a verificação dos pacotes que são enviados na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

   Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

2. Inicie o CORE e abra o arquivo “**Funcionalidade-Neighbor-Discovery-E1.imn**” localizado no diretório do desktop “Funcionalidades/NeighborDiscovery”, da máquina virtual do NIC.br. A seguinte topologia inicial de rede deve aparecer deve aparecer:



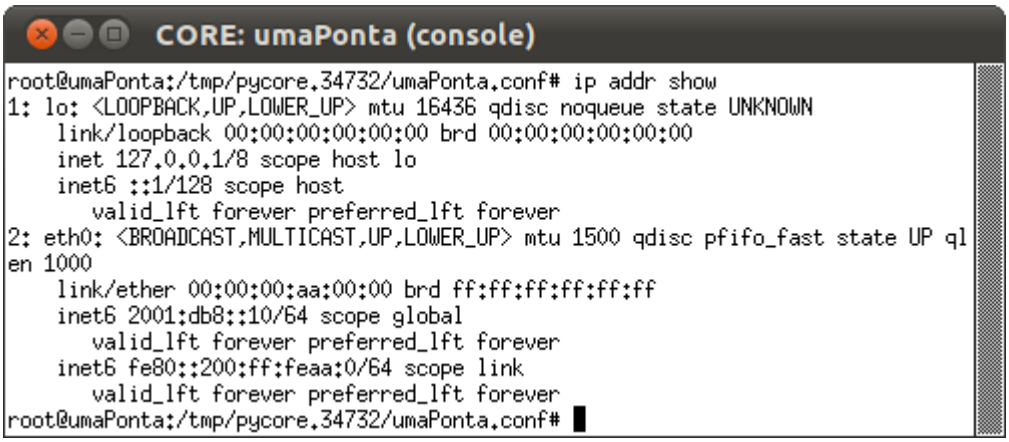
3. Verifique a configuração dos nós da topologia.
   a. Inicie a simulação realizando um dos seguintes passos:
      i. aperte o botão  ;
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do umaPonta, através do duplo-clique.

   c. Verifique que a configuração do umaPonta através do seguinte comando:

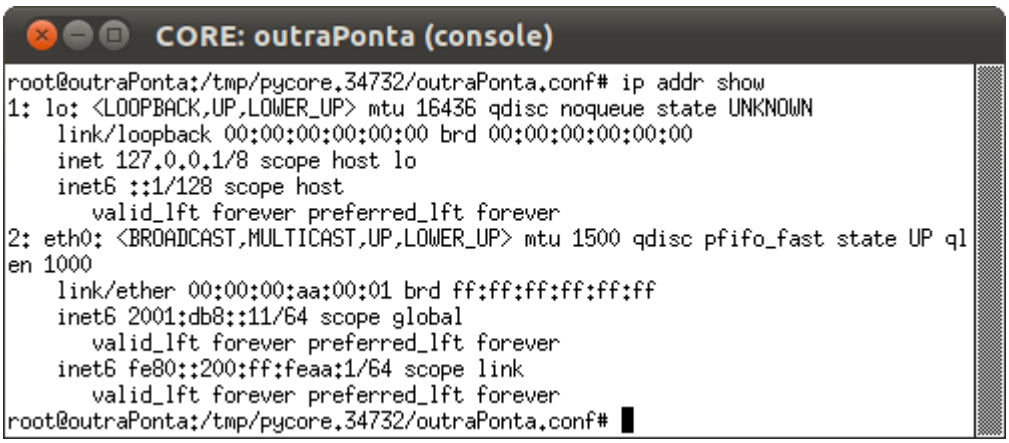
```
# ip addr show
```

   O resultado deve ser:



   ***Obs**: A partir desse comando é possível observar os endereços das interfaces.

   d. Verifique que a configuração do outraPonta através do mesmo comando.
   O resultado deve ser:



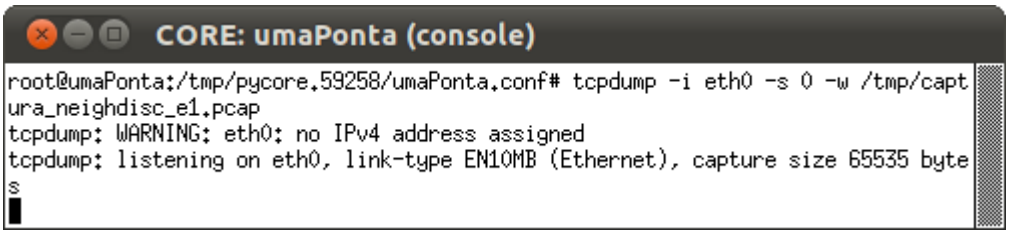
   ***Obs**: A partir desse comando é possível observar os endereços das interfaces.

4. Teste a conectividade IPv6 de um no ao outro.
    a. Abra o terminal do umaPonta, através do duplo-clique.

    b. Utilize o seguinte comando para iniciar a captura de pacotes do roteador:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_neighdisc_e1.pcap
```

    O resultado deve ser:

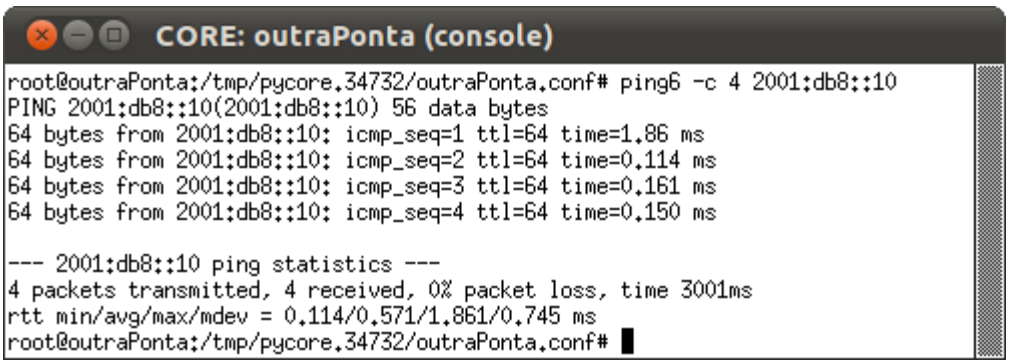


    ***Obs**: Não feche esse terminal até o final do experimento, uma vez que, isso ocasionará no término da execução do comando “tcpdump” e prejudicará o andamento da experiência.

    c. No terminal do outraPonta, utilize o seguinte comando :

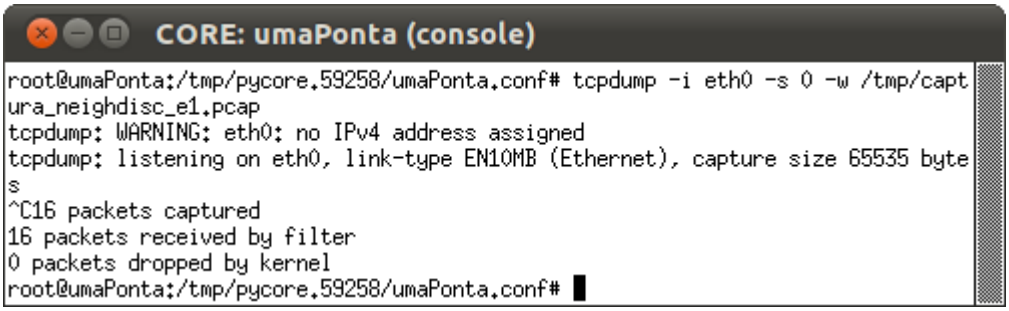
```
# ping6 -c 4 2001:db8::10
```

    O resultado deve ser:



    ***Obs**: o endereço de destino deve ser o do umaPonta. Obtido pelo comando “ip addr” anteriormente.

d. No terminal do umaPonta, encerre a captura de pacotes através da sequência Ctrl+C.
O resultado deve ser:



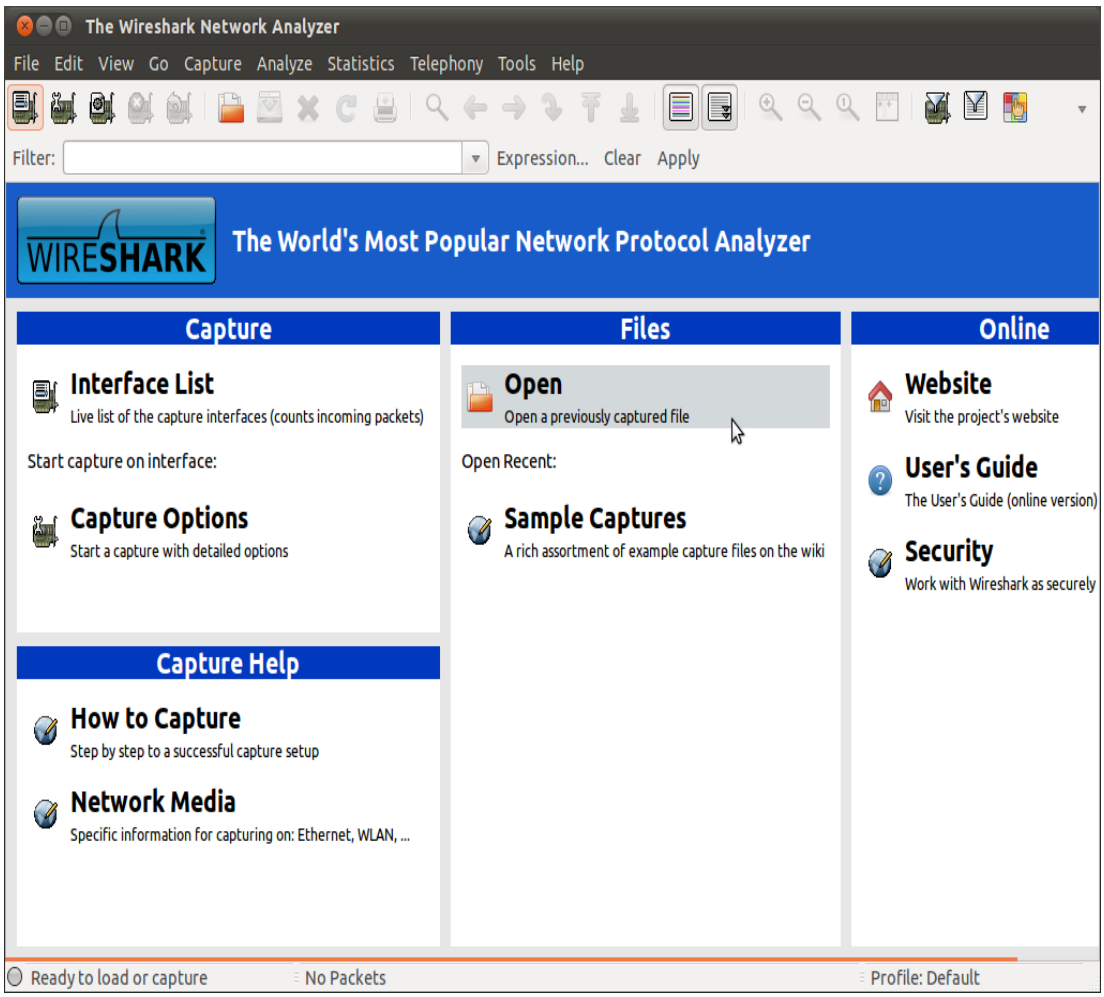
***Obs**: A quantidade de pacotes pode variar de acordo com o tempo esperado para dar o comando Ctrl+C.

5. Encerre a simulação com uma das seguintes ações:

   i. aperte o botão  ;
   ii. utilize o menu Experiment > Stop.

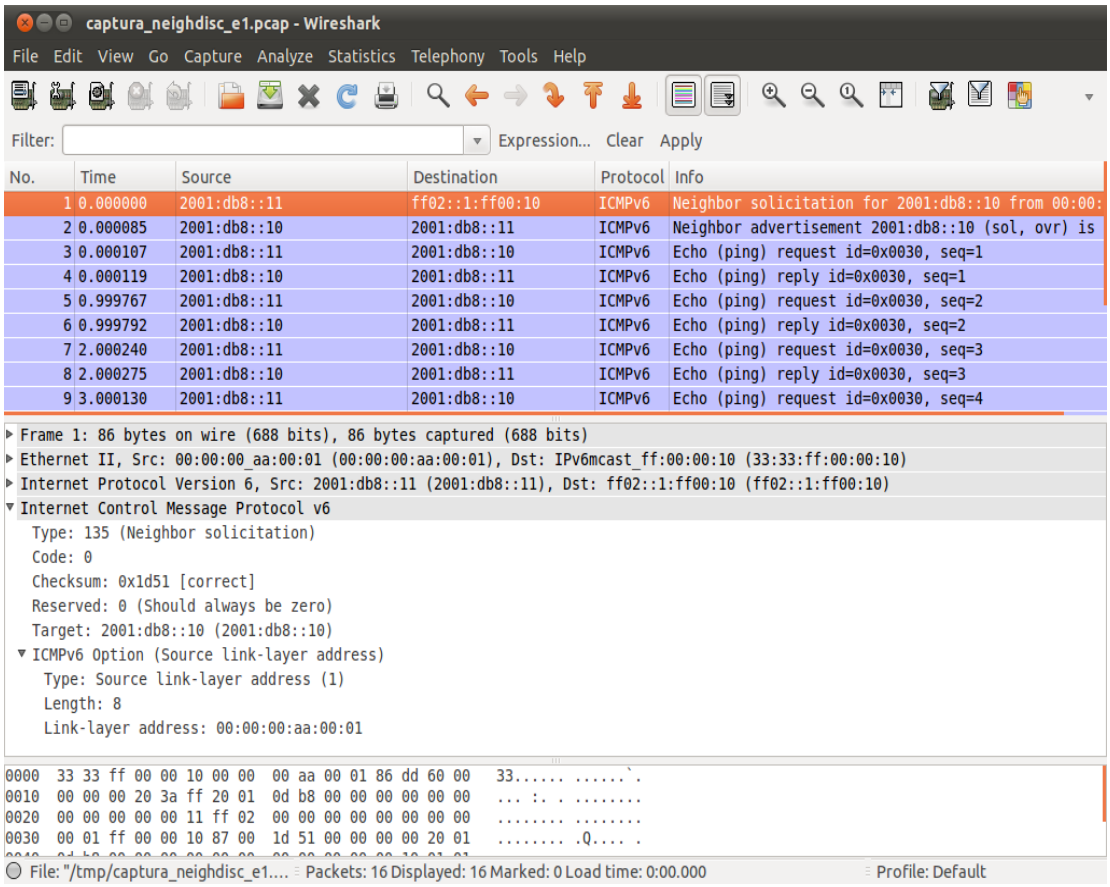
6. A verificação dos pacotes capturados será realizada através do programa Wireshark. Para iniciá-lo execute o seguinte comando em um terminal da máquina virtual:

```
$ wireshark
```



a. Abra o arquivo **/tmp/captura_neighdisc_e1.pcap** com o menu File>Open:
b. Procure pelos pacotes de *Neighbor Solicitation* e *Neighbor Advertisement*. Analise-os e veja se os dados contidos nos pacotes conferem com o que foi passado na teoria.

Neighbor Solicitation:

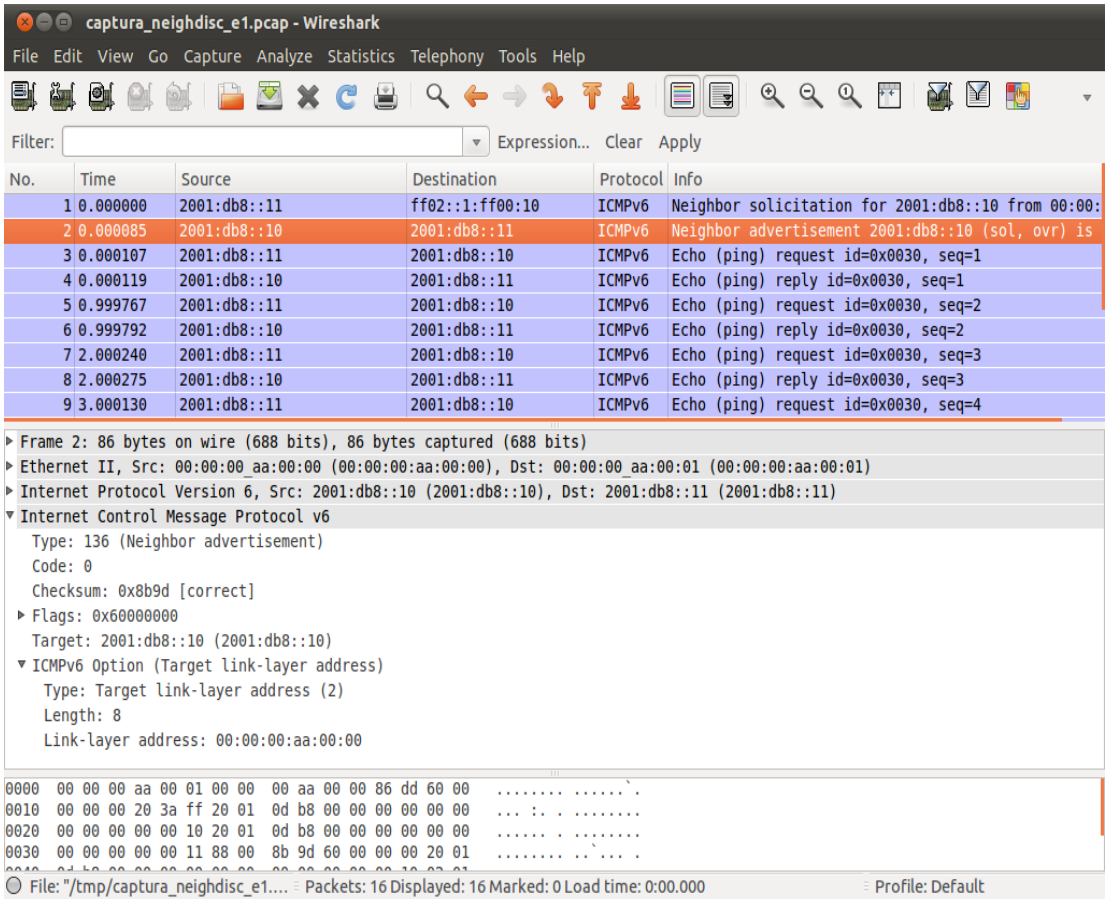


***Obs**: o filtro icmpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço (33:33:ff:00:00:10), sendo que o prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet. O sufixo ff:00:00:10 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a fonte é o MAC address da interface do dispositivo que enviou a solicitação (00:00:00:aa:00:01).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho (de extensão do IPv6), no caso, o valor 58(0x3a) refere-se à uma mensagem ICMPv6.
- **Source (IPv6)**: a fonte é o endereço IP da interface diretamente ligada ao enlace em que se faz a requisição (2001:db8::11).
- **Destination (IPv6)**: o destino é o endereço multicast Solicited-Node (ff02::1:ff00:10).
- **Type (ICMPv6)**: indica que a mensagem é do tipo 135 (Neighbor Solicitation).
- **ICMPv6 Option (ICMPv6)**: indica as opções do pacote ICMPv6:
    - Source Link Layer Address
        - *Type*: indica o tipo de dado da mensagem ICMPv6. Em nosso caso, ela é do tipo "*Source link-layer address*".
        - *Link-layer address*: indica o MAC address da interface de origem da mensagem.

Neighbor Advertisement:



***Obs**: o filtro icmpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: MAC address do nó requisitante que foi obtido através da mensagem de *Neighbor Solicitation* enviada anteriormente (00:00:00:aa:00:01).
- **Source (Ethernet)**: a origem é o MAC address da interface do dispositivo umaPonta que enviou a resposta (00:00:00:aa:00:00).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho (de extensão do IPv6), no caso, o valor 58(0x3a) refere-se à uma mensagem ICMPv6.
- **Destination (IPv6)**: diferentemente da mensagem *Neighbor Solicitation*, a Neighbor Advertisement possui como destino o endereço IPv6 global do nó requisitante (2001:db8::11).
- **Source (IPv6)**: a origem é o endereço IP da interface diretamente ligada ao enlace em que a requisição foi recebida (2001:db8::10).
- **Type (ICMPv6)**: indica que a mensagem é do tipo 136 (*Neighbor Advertisement*).
- **Flags (ICMPv6)**: uma mensagem do tipo *Neighbor Advertisement* possui 3 flags:
    - A primeira indica se quem está enviando é um roteador. No nosso caso, ela está marcada com 0.
    - A segunda indica se a mensagem é uma resposta a um Neighbor Solicitation.

No nosso caso, ela está setada em 1.
    - A terceira indica se a informação carregada na mensagem é uma atualização de endereço de algum nó da rede. No nosso caso, ela está setada em 1.

- **Target Address (ICMPv6)**: indica o endereço IP sobre o qual as flags informam. No nosso caso, é o próprio endereço da interface do dispositivo umaPonta (2001:db8::10).
- **ICMPv6 Option (ICMPv6)**: indica as opções do pacote ICMPv6:
    - Target Link Layer Address
        - *Type*: indica o tipo de dado da mensagem ICMPv6. Em nosso caso, ela é do tipo "*Target link-layer address*".
        - *Link-layer address*: indica o MAC address da interface do dispositivo umaPonta da mensagem.

Exercícios

1) Qual é o campo da mensagem *Neighbor Solicitation* que identifica o destino procurado?

_____

2) Qual é o campo que transmite a informação do MAC address do destino?

_____

3) Por que é enviado o endereço MAC address da origem?

_____

4) Faça um desenho esquemático que representa a funcionalidade de descoberta de endereços de camada 2.

5) Crie uma nova topologia através da adição de dois elementos, um switch e um PC. A partir disso, capture numa só interface de máquina duas mensagens *Neighbor Solicitations* distintas.

IPV6 - Neighbor Discovery Protocol

Experiência 2 e 3 - Router Solicitation e Router Advertisement

Objetivo

Esta experiência possui como objetivo apresentar o funcionamento do mecanismo de descoberta de roteadores. Para isso, ela foi dividida em duas partes.

A primeira focada no envio da mensagem *Router Solicitation* com resposta *Router Advertisement*. E a segunda, mostra o anuncio do roteador para a rede com o uso da mensagem *Router Advertisement*.

O presente exercício utiliza as topologias descritas nos seguintes arquivos:

- **Funcionalidade-Neighbor-Discovery-E2.imn**;
- **Funcionalidade-Neighbor-Discovery-E3.imn**.

Introdução Teórica

A descoberta de roteadores é um procedimento realizado pelos nós da rede quando configuram seu endereço link local, ou seja, no momento que se conectam ou se reconectam em uma rede, para descobrir características do enlace e rotas de comunicação.

O mecanismo começa com o envio da mensagem *Router Solicitation* direcionado a todos os roteadores no enlace, utilizando o endereço de destino *All-Router (FF02::2) multicast Group.* Além de procurar por roteadores, essa mensagem já contém o endereço físico do próprio dispositivo de forma a evitar novas trocas de pacotes.
O roteador, ao receber o *Router Solicitation*, gera uma resposta *Router Advertisement* que é enviada diretamente ao nó solicitante através de seu endereço de link local. Porém, é possível que essa resposta seja enviada em multicast *All Node*, realizando o envio do *Router Advertisement* para todos os nós.
O *Router Adversiment* serve para informar ao nó sobre a existência do roteador.

De tempos em tempos os roteadores também enviam *Router Advertisements* para informar que continuam operando corretamente.

Roteiro Experimental

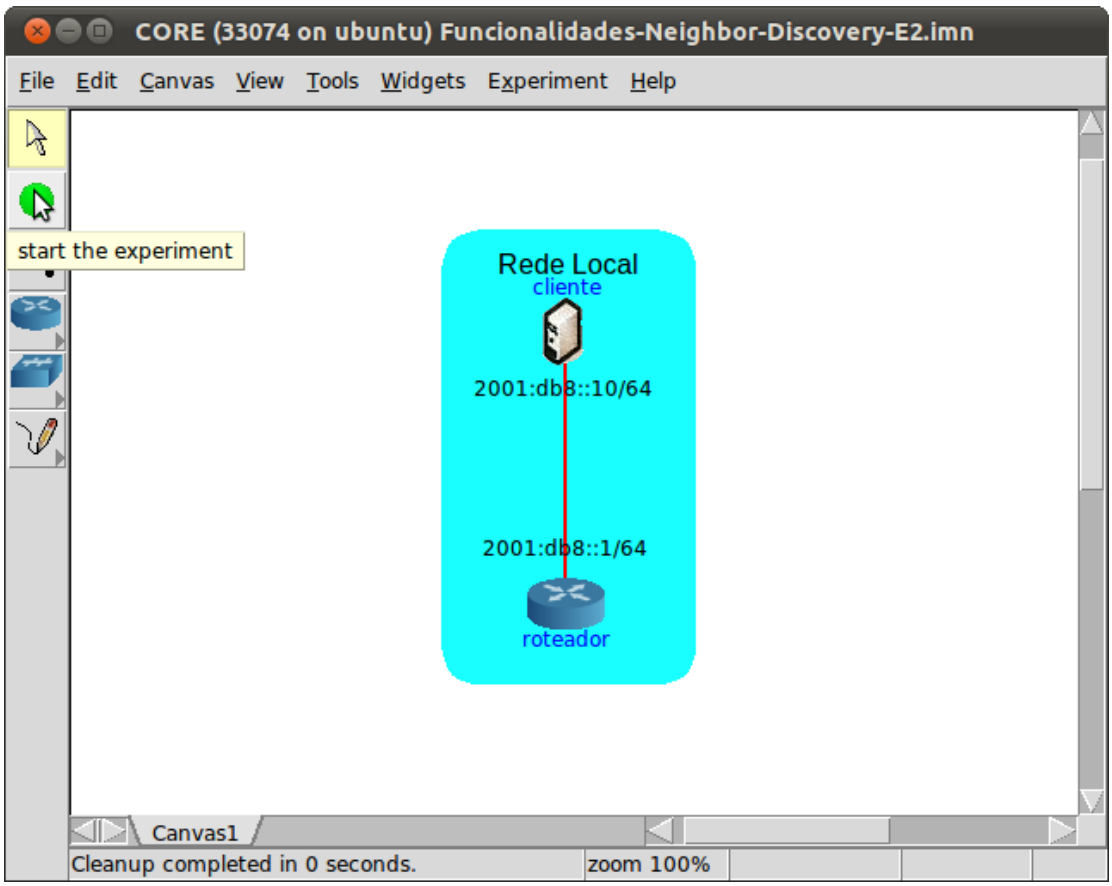
Experiência 2 - Router Solicitation

1. Caso não esteja utilizando a máquina virtual fornecida pelo NIC.br é preciso, antes de começar a experiência, instalar alguns softwares para auxiliar no aprendizado (caso contrário vá para o passo 2).
   Siga o passo a seguir para realizar a instalação:
   a. Para fazer algumas verificações durante o experimento será necessário a utilização do programa **Wireshark** que realiza a verificação dos pacotes que são enviados na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

   Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

2. Inicie o CORE e abra o arquivo “**Funcionalidade-Neighbor-Discovery-E2.imn**” localizado no diretório do desktop “Funcionalidades/NeighborDiscovery”, da máquina virtual do NIC.br. A seguinte topologia inicial de rede deve aparecer:



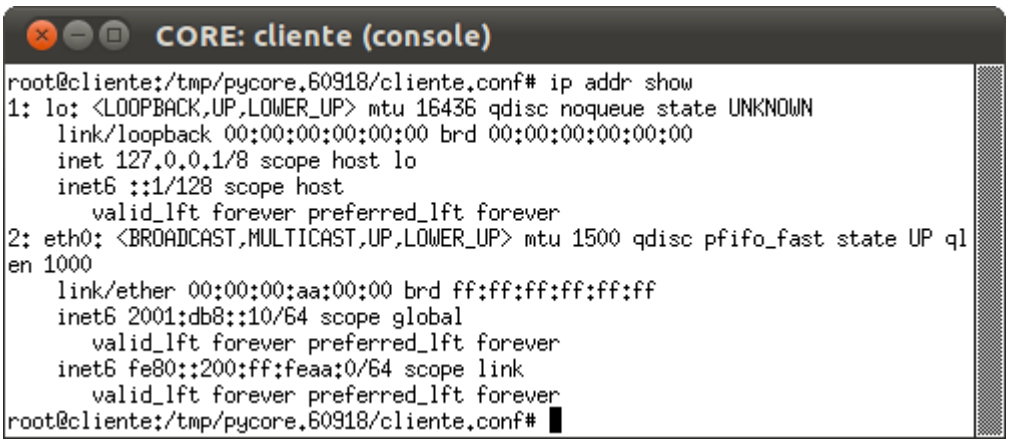
3. Verifique a configuração dos nós da topologia.
   a. Inicie a simulação utilizando uma das seguintes maneiras:
      i. aperte o botão  ;
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do cliente, através de duplo-clique.

   c. Verifique a configuração do cliente através do seguinte comando:

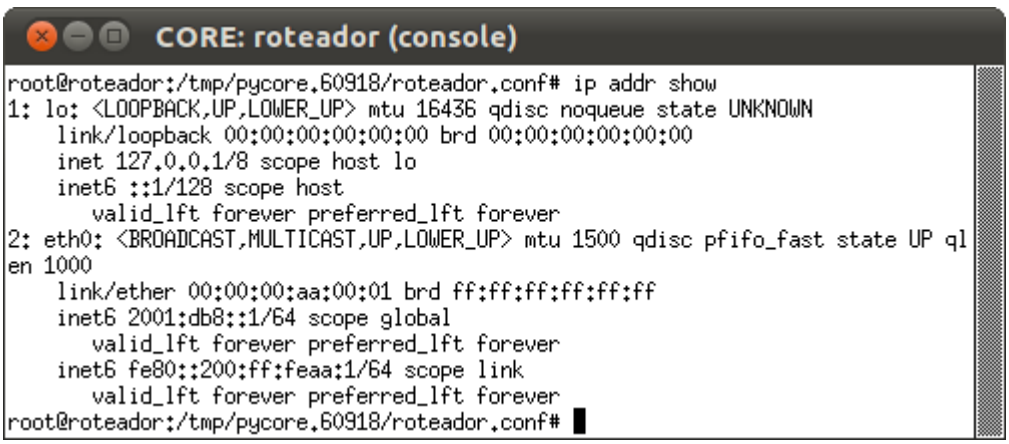
```
# ip addr show
```

   O resultado deve ser:



   ***Obs**: A partir desse comando é possível observar os endereços das interfaces.

   d. Verifique a configuração do roteador com o mesmo comando.
   O resultado deve ser:



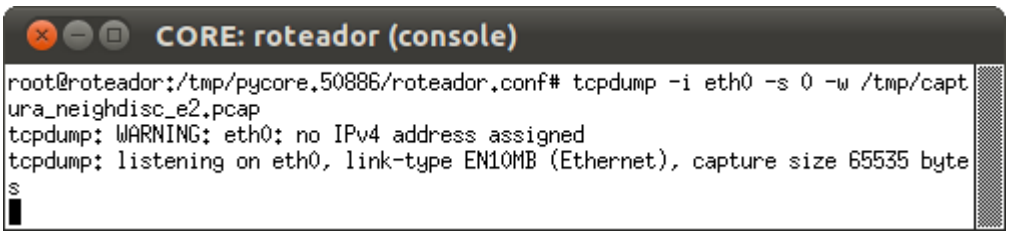
   ***Obs**: A partir desse comando é possível observar os endereços das interfaces.

4. Teste do envio da mensagem Router Solicitation:
    a. Abra o terminal do roteador, através do duplo-clique.

    b. Utilize o seguinte comando para iniciar a captura de pacotes do cliente:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_neighdisc_e2.pcap
```

    O resultado deve ser:



    ***Obs**: Não feche esse terminal até o final do experimento, uma vez que, isso ocasionará no término da execução do comando “tcpdump” e prejudicará o andamento da experiência.

    c. Abra o terminal do cliente, através do duplo-clique.

    d. No terminal utilize a seguinte sequencia de comandos, para forçar envio de router solicitation :

```
# ip link set eth0 down
# ip addr add 2001:db8::10/64 dev eth0
# ip link set eth0 up
```

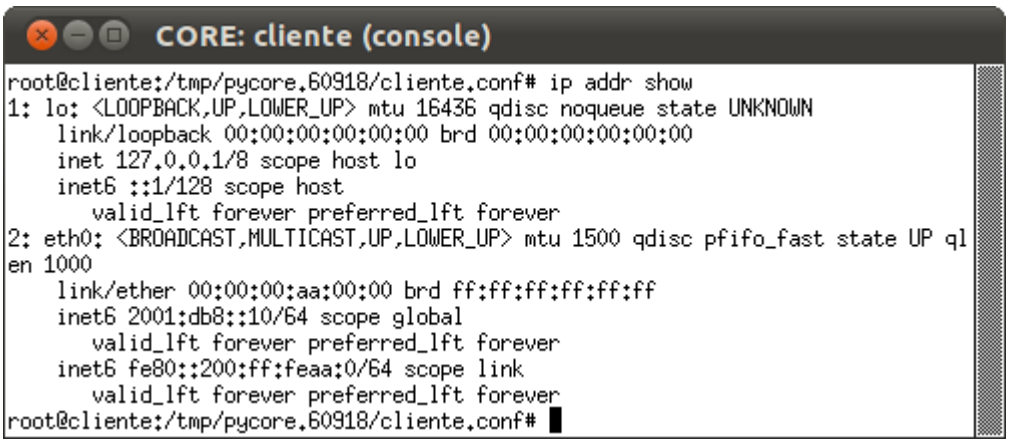
    O resultado deve ser:

    ***Obs**: Note que esses comandos estão reiniciando e restaurando a interface eth0, para ocasionar o envio da mensagem *Router Solicitation*.

e. Verifique a configuração da interface eth0 se houve alguma mudança. Utilize o comando:

```
# ip addr show
```

O resultado deve ser:

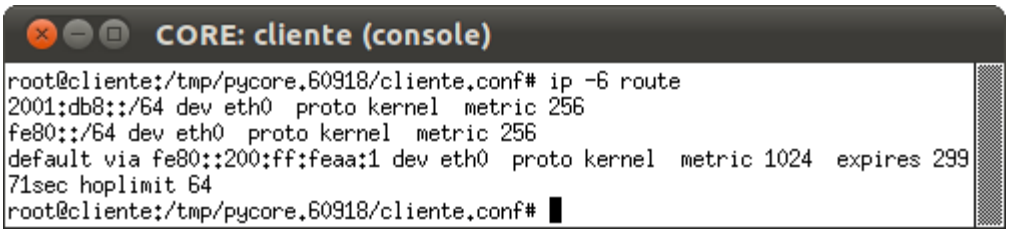


***Obs**: Não deve haver nenhuma alteração nas interfaces comparado com seu estado anterior, visto no passo 3.

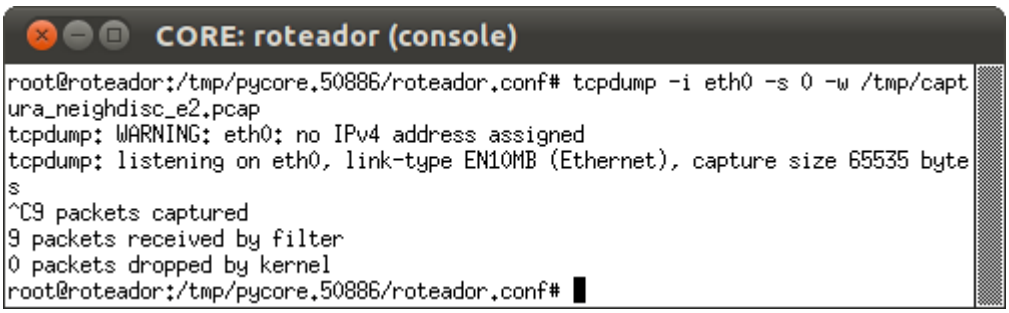
f. Verifique a rota aprendida pela funcionalidade de descoberta de roteadores. Utilize o comando:

```
# ip -6 route
```

O resultado deve ser:



e. No terminal do roteador, encerre a captura de pacotes através da sequência Ctrl+C.
O resultado deve ser:



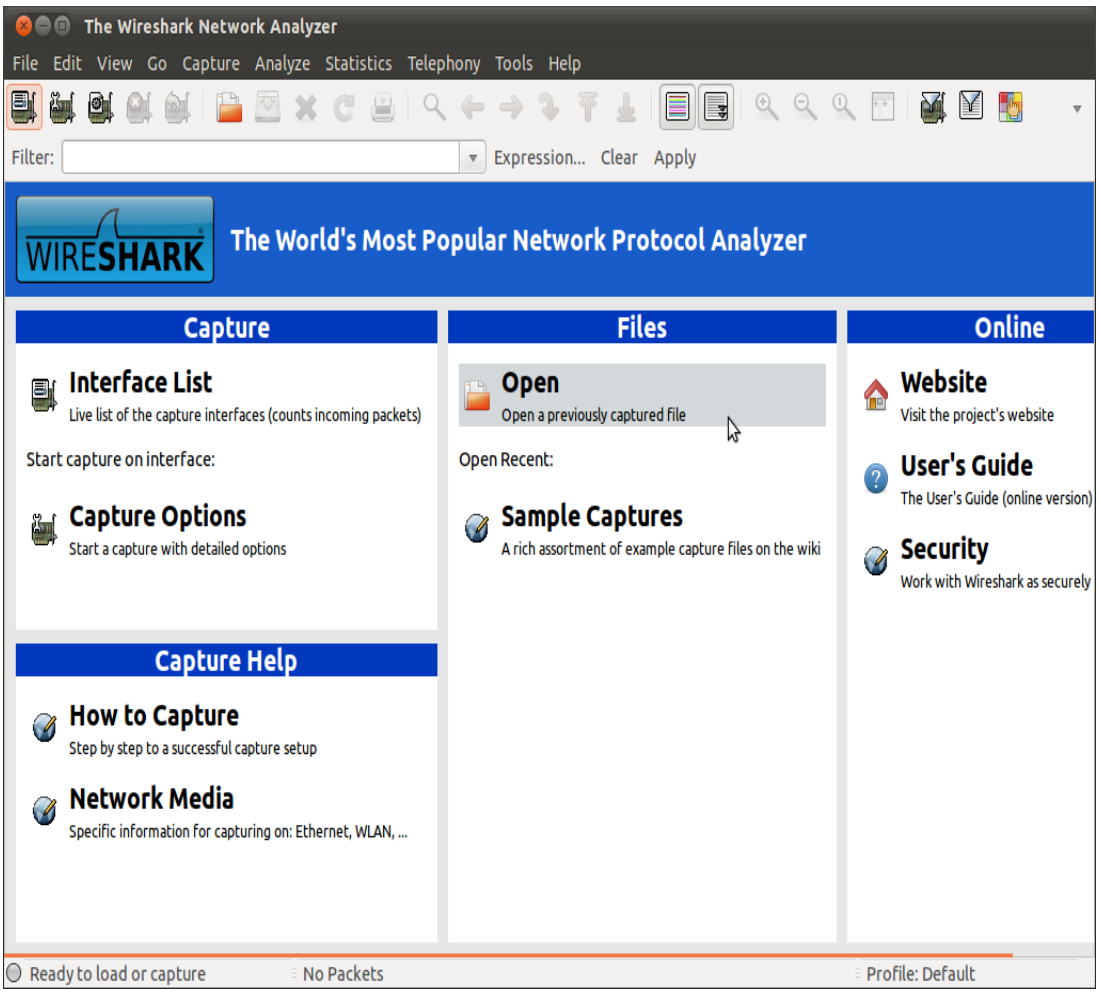
***Obs**: A quantidade de pacotes pode variar de acordo com o tempo esperado para dar o comando Ctrl+C.

5. Encerre a simulação com um dos seguintes comandos:

   i. aperte o botão  ;
   ii. utilize o menu Experiment > Stop.

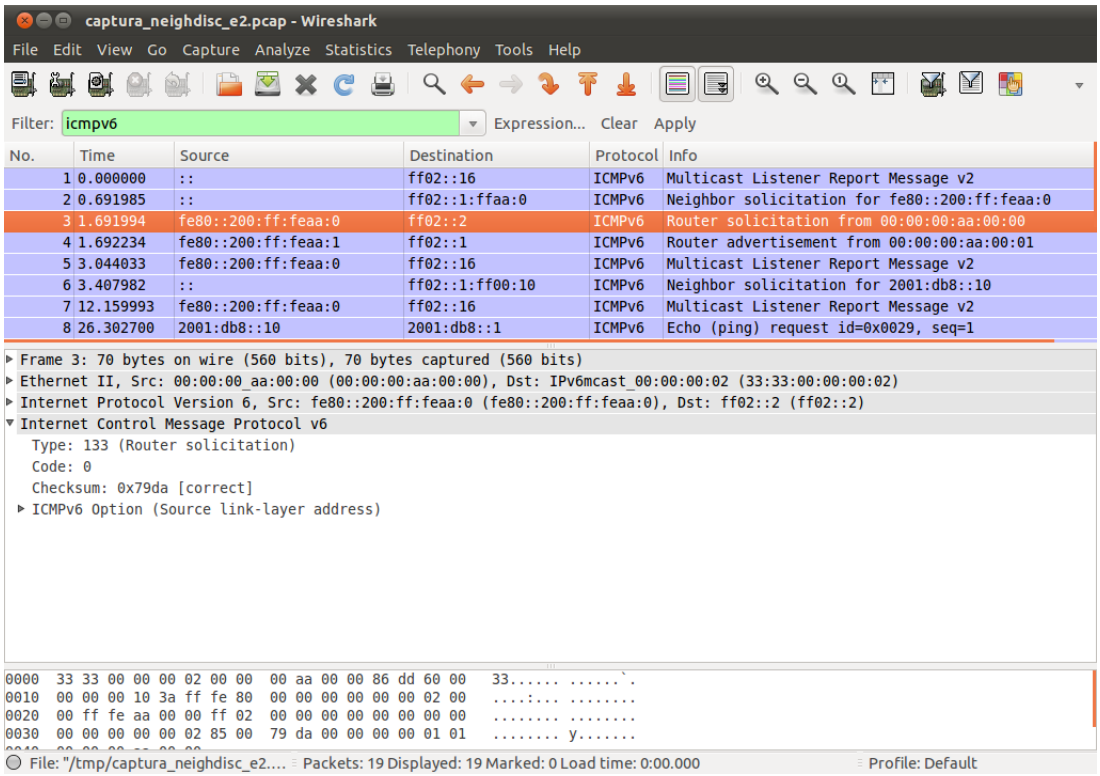
6. A verificação dos pacotes capturados será realizada através do programa Wireshark. Para iniciá-lo execute o seguinte comando em um terminal da máquina virtual:

```
$ wireshark
```



a. Abra o arquivo **/tmp/captura_neighdisc_e2.pcap** com o menu File>Open:
b. Procure pelos pacotes *Router Solicitation* e *Router Advertisement*. Analise-os e veja que os dados contidos no pacote confere com os passados na teoria.

Router Solicitation:

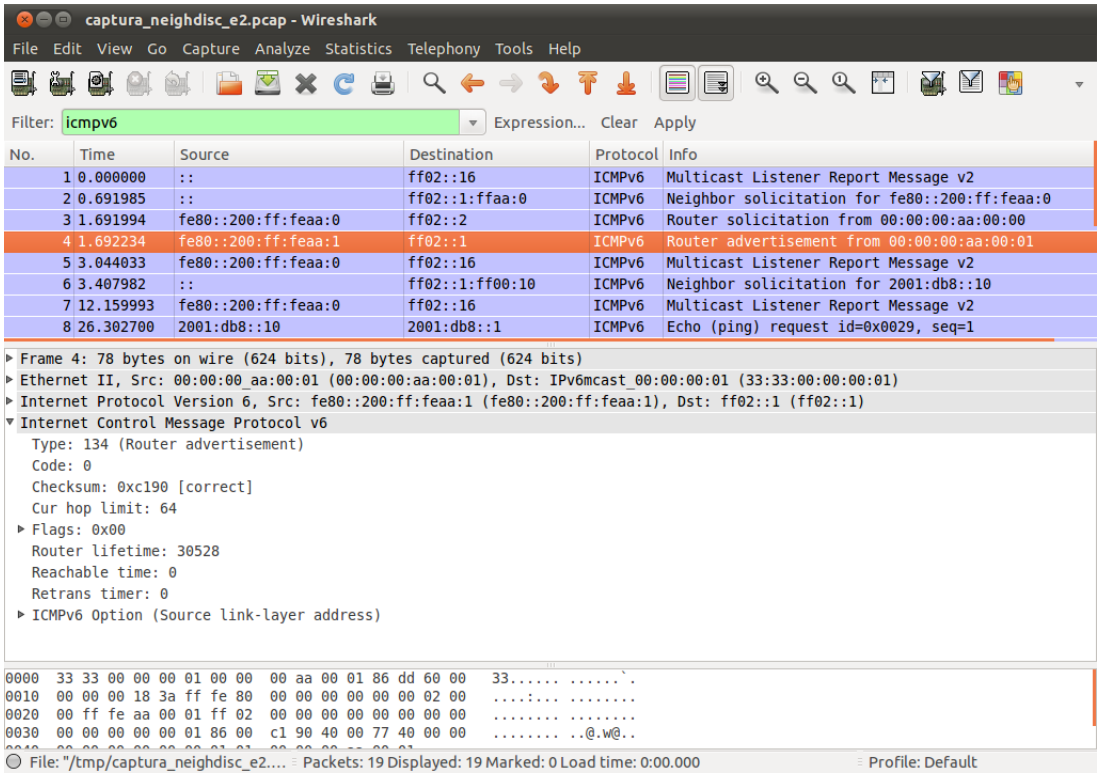


***Obs**: o filtro icmpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço (33:33:00:00:00:02). O prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet. O sufixo ff:00:00:02 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a origem é o MAC address da interface do dispositivo cliente que enviou a mensagem (00:00:00:aa:00:00).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho (de extensão do IPv6), no caso, o valor 58 (0x3a) refere-se à uma mensagem ICMPv6.
- **Source (IPv6)**: é o endereço ipv6 de link local da interface que originou a mensagem (fe80::200:ff:feaa:0).
- **Destination (IPv6)**: o destino é o endereço ipv6 Multicast All Routers (ff02::2).
- **Type (ICMPv6)**: indica que a mensagem é do tipo 133 (Router Solicitation).
- **ICMPv6 Option (ICMPv6)**: indica as opções do pacote ICMPv6:
    - Source Link Layer Address
        - *Type*: indica o tipo de dado da mensagem ICMPv6. Em nosso caso, ela é do tipo "*Source link-layer address*".
        - *Link-layer address*: indica o MAC address do endereço de origem da mensagem.

Router Advertisement:



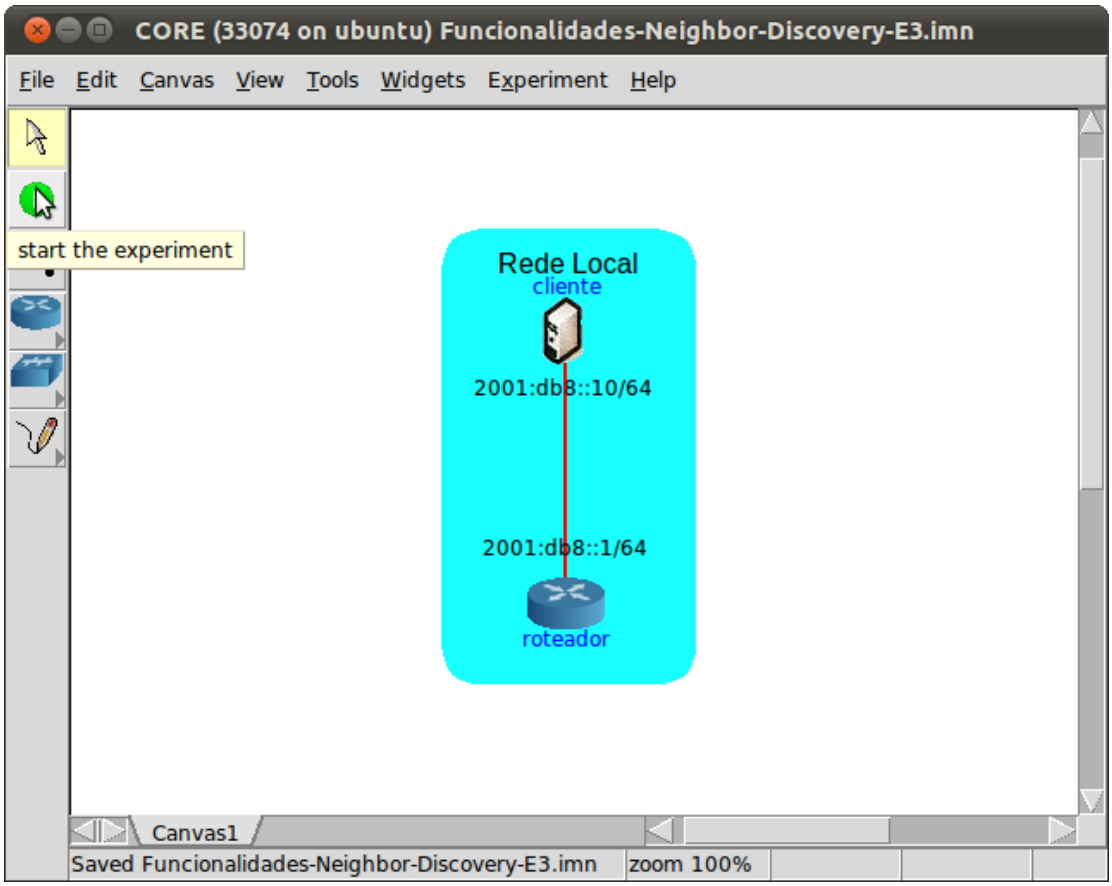
***Obs**: o filtro icmpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço (33:33:00:00:00:01). O prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet. O sufixo ff:00:00:01 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a origem é o MAC address da interface do roteador que enviou a mensagem (00:00:00:aa:00:01).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho (de extensão do IPv6), no caso, o valor 58 (0x3a) refere-se à uma mensagem ICMPv6.
- **Source (IPv6)**: é o ip de link local da interface que originou a resposta, que neste caso é o roteador (fe80::200:ff:feaa:1).
- **Destination (IPv6)**: o destino é o endereço Multicast All nodes (ff02::1).
- **Type (ICMPv6)**: indica que a mensagem é do tipo 134 (Router Advertisement).
- **ICMPv6 Option (ICMPv6)**: indica as opções do pacote ICMPv6:
    - Source Link Layer Address
        - *Type*: indica o tipo de dado da mensagem ICMPv6. Em nosso caso, ela é do tipo "*Source link-layer address*".
        - *Link-layer address*: indica o MAC address do endereço de origem da mensagem, que neste caso é o roteador.

Experiência 3 - Router Advertisement

1. Inicie o CORE e abra o arquivo "**Funcionalidade-Neighbor-Discovery-E3.imn**" localizado no diretório do desktop "Funcionalidades/NeighborDiscovery", da máquina virtual do NIC.br. A seguinte topologia inicial de rede deve aparecer deve aparecer:



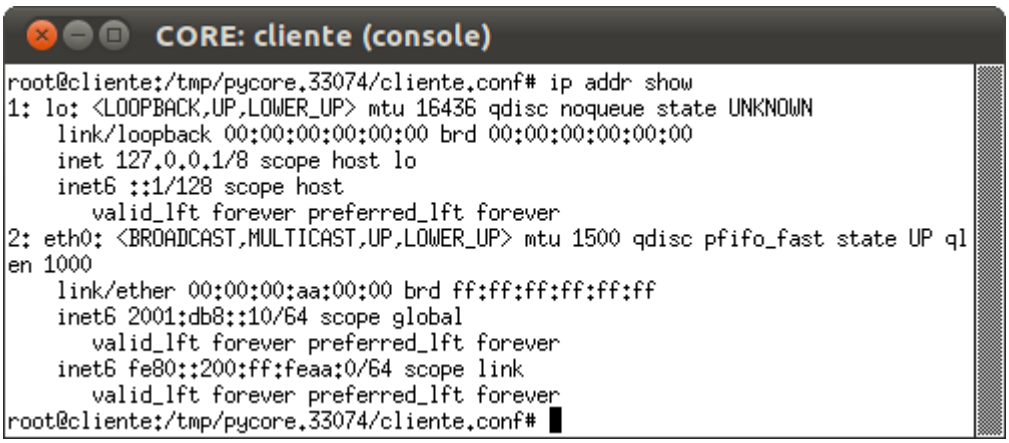
2. Verifique a configuração dos nós da topologia.
   a. Inicie a simulação com um dos seguintes comandos:
      i. aperte o botão  ;
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do cliente, através do duplo-clique.

   c. Verifique que a configuração do cliente através do seguinte comando:

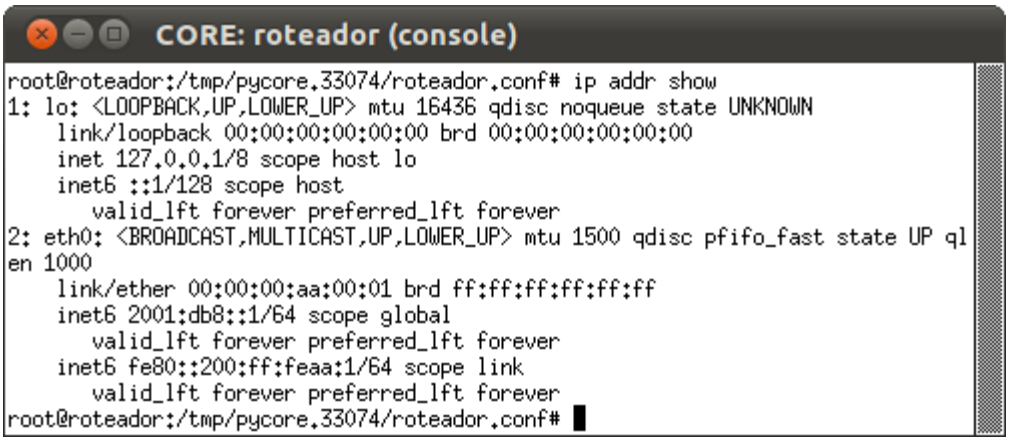
```
# ip addr show
```

O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces.

d. Verifique que a configuração do roteador através do mesmo comando.
O resultado deve ser:



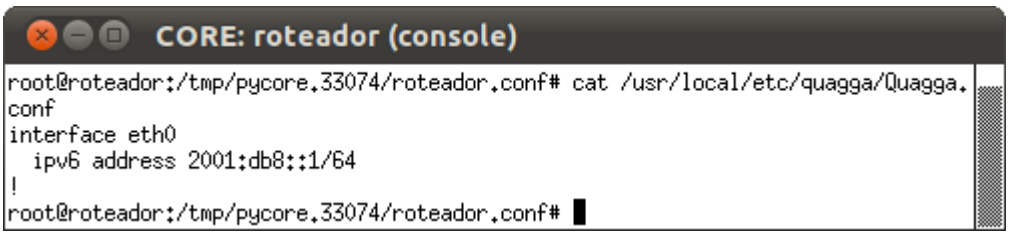
***Obs**: A partir desse comando é possível observar os endereços das interfaces.

3. Editando as configurações do Quagga, para enviar a mensagem Router Advertisement.
   a. Abra o terminal do roteador, através do duplo-clique.

   b. Utilize o seguinte comando para visualizar o arquivo de configuração do quagga, chamado “Quagga.conf”:

```
# cat /usr/local/etc/quagga/Quagga.conf
```

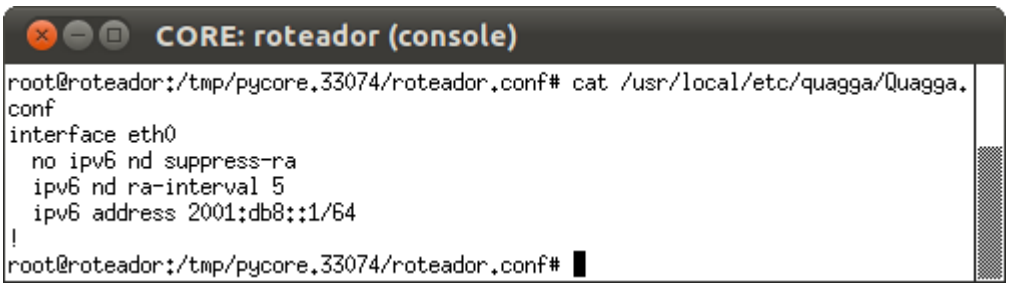
O resultado deve ser:



c. Edite esse arquivo de configuração (“Quagga.conf”), adicionando as seguintes linhas dentro do escopo da interface (ou seja, entre as linhas “interface eth0” e “!”:

```
no ipv6 nd suppress-ra
ipv6 nd ra-interval 5
```

O resultado deve ser:



***Obs**: um editor de texto presente na máquina virtual que pode ser utilizado é o **nano**. Para usá-lo digite no terminal:

```
# nano /usr/local/etc/quagga/Quagga.conf
```

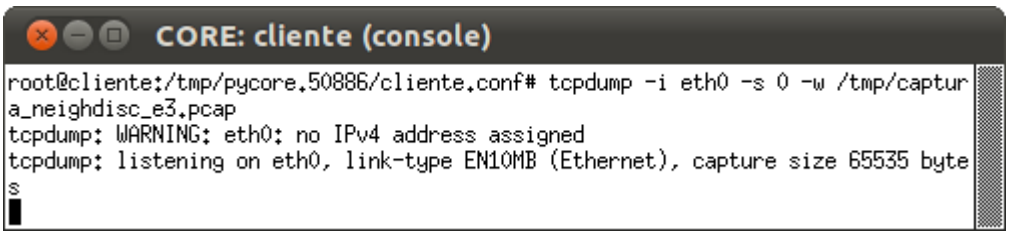
No nano, a sequência utilizada para salvar o arquivo é CTRL-O e para sair é CTRL-X.

4. Teste das novas configurações do Quagga.
   a. Abra o terminal do cliente, através do duplo-clique.

   b. Utilize o seguinte comando para iniciar a captura de enviados pacotes pelo roteador:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_neighdisc_e3.pcap
```

O resultado deve ser:



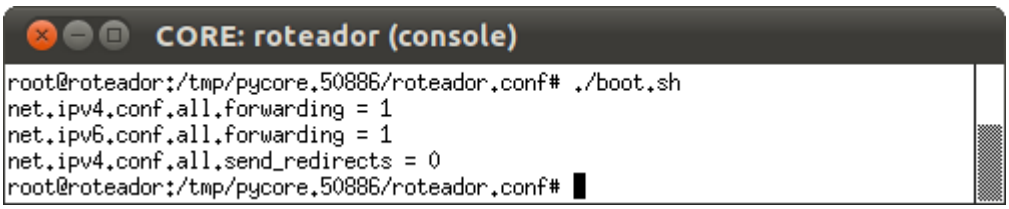
***Obs**: Não feche esse terminal até o final do experimento, uma vez que, isso ocasionará no término da execução do comando “tcpdump” e prejudicará o andamento da experiência.

c. Abra o terminal do roteador, através do duplo-clique.

d. Utilize o seguinte comando para iniciar o quagga com as novas configurações:

```
# ./boot.sh
```

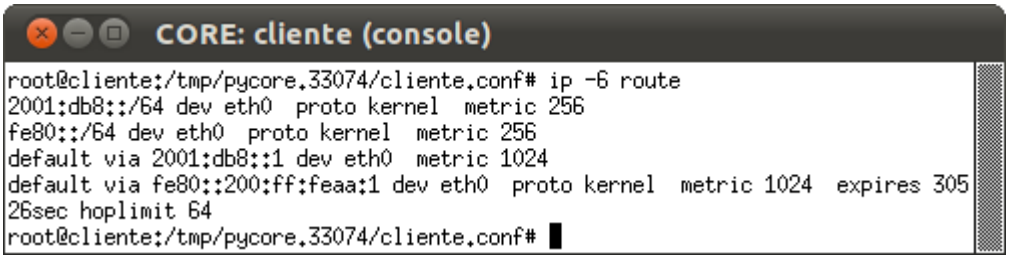
O resultado deve ser:



g. Espere alguns segundos e abra um novo terminal do cliente. Verifique a rota aprendida pela funcionalidade de descoberta de roteadores. Utilize o comando:

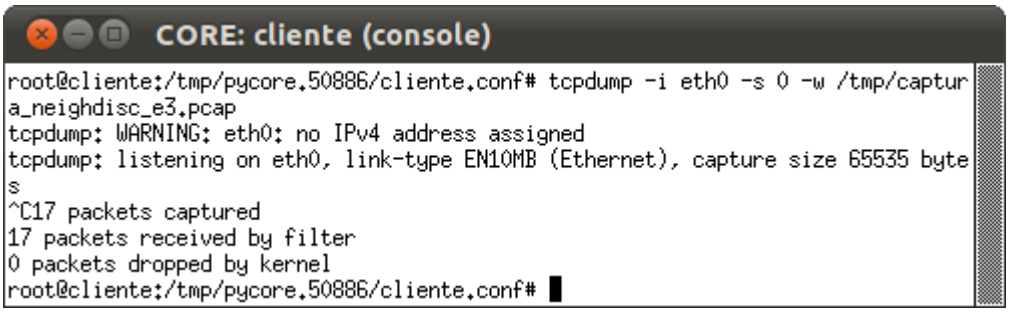
```
# ip -6 route
```

O resultado deve ser:



e. No terminal do cliente, encerre a captura de pacotes através da sequência Ctrl+C.

O resultado deve ser:



***Obs**: A quantidade de pacotes pode variar de acordo com o tempo esperado para dar o comando Ctrl+C.

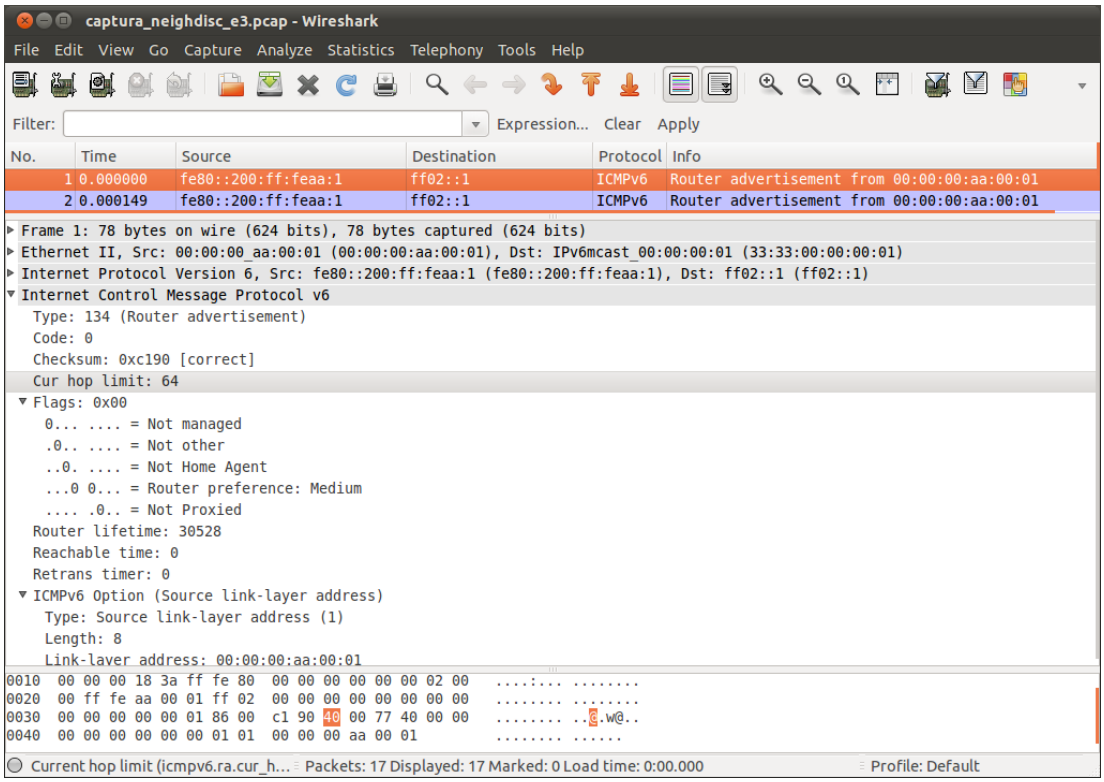
5. Encerre a simulação utilizando um dos seguintes comandos:
   a. aperte o botão  ;
   b. utilize o menu Experiment > Stop.

6. A verificação dos pacotes capturados será realizada através do programa Wireshark. Para iniciá-lo execute o seguinte comando em um terminal da máquina virtual:

```
$ wireshark
```

   a. Abra o arquivo **/tmp/captura_neighdisc_e3.pcap** com o menu File>Open:
   b. Procure pelo pacote *Router Advertisement*. Analise-o e veja se os dados contidos no pacote conferem com o que foi passado na teoria.

Router Advertisement:



***Obs** o filtro icmpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço MAC (33:33:00:00:00:01). O prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet. O sufixo ff:00:00:01 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a origem é o MAC address da interface do roteador que enviou a mensagem (00:00:00:aa:00:01).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho (de extensão do IPv6), no caso, o valor 58 (0x3a) refere-se à uma mensagem ICMPv6.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface que originou a mensagem, que neste caso é o roteador (fe80::200:ff:feaa:1).
- **Destination (IPv6)**: o destino é o endereço Multicast All Nodes (ff02::1).
- **Type (ICMPv6)**: indica que a mensagem é do tipo 134 (Router Advertisement).
- **ICMPv6 Option (ICMPv6)**: indica as opções do pacote ICMPv6:
    - Source Link Layer Address
        - *Type*: indica o tipo de dado da mensagem ICMPv6. Em nosso caso, ela é do tipo "*Source link-layer address*".
        - *Link-layer address*: indica o MAC address da interface a partir da qual a mensagem de Router Advertisement foi enviada, neste caso, 00:00:00:aa:00:01.

IPV6 - Neighbor Discovery Protocol

Experiência 4 - Detecção de endereços duplicados

Objetivo

Esta experiência possui o objetivo de apresentar o funcionamento do mecanismo de detecção de endereços duplicados, através do estudo das mensagens de *Neighbor Solicitation*, *Neighbor Advertisement* e *Reply*.

Para a realização do presente exercicio será utilizada a topologia descrita no arquivo: **Funcionalidade-Neighbor-Discovery-E4.imn**.

Introdução Teórica

A detecção de endereços duplicados é um procedimento realizado pelos nós para verificar a unicidade de seus endereços unicast no enlace, antes de se atribuir a uma interface. Independente da maneira que se foi obtido o endereço, seja manualmente, autoconfiguração stateless ou autoconfiguração stateful, endereços duplicados não devem ser aceitos em redes IPv6.

O mecanismo consiste no envio de uma mensagem *Neighbor Solicitation* pelo dispositivo que está tentando adicionar o endereço à interface. Essa mensagem é, então, transmitida a todos os nós do enlace (destino Multicast All Nodes) procurando pela existência de um nó utilizando o mesmo endereço. Para isso, ele carrega o campo *source* do IPv6 vazio e o *target* do ICMPv6 com o endereço requisitado.

Caso se receba uma mensagem *Neighbor Advertisement*, o processo de configuração será interrompido e o endereço não poderá ser utilizado. Nessa situação, o conflito só é solucionado manualmente, com adição de um novo endereço em um dos dois dispositivos.

Caso tempo de espera de 1 segundo (valor que pode ser alterado para diferentes links) seja ultrapassado e não seja recebida nenhuma mensagem, o dispositivo poderá então finalizar sua configuração de interface.

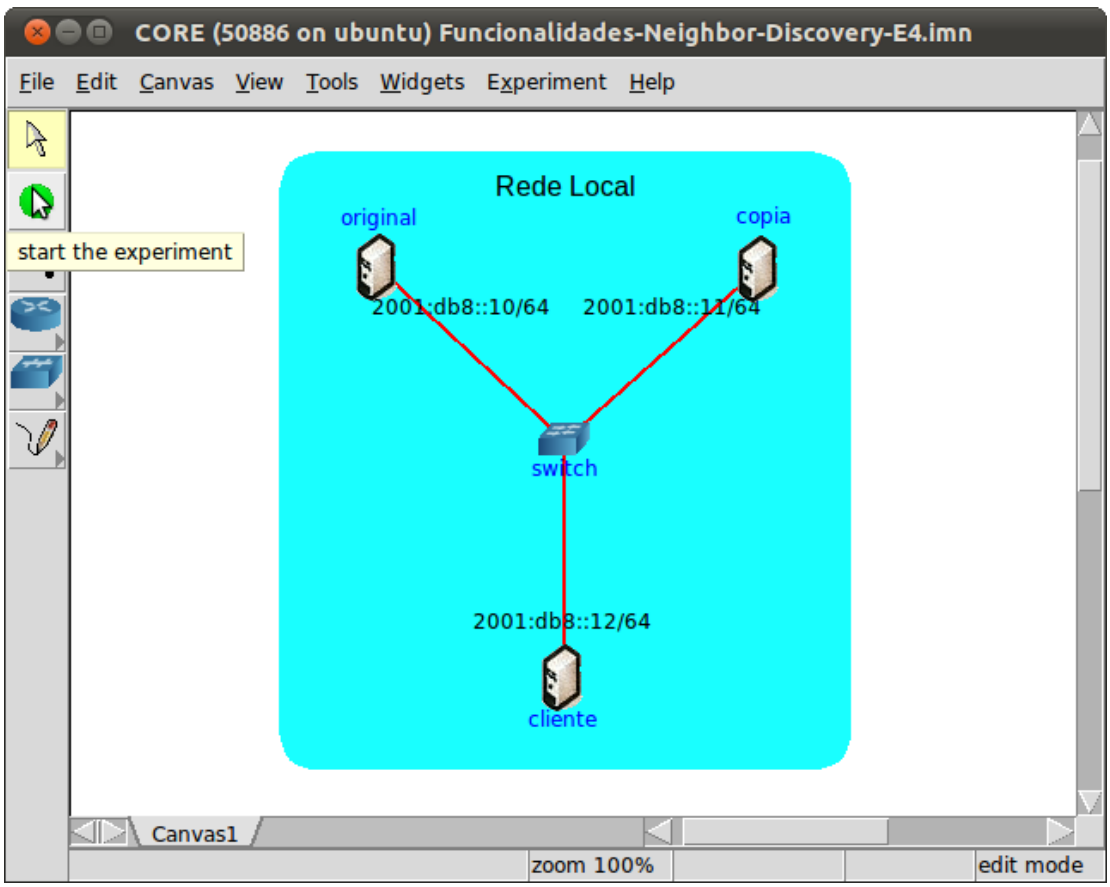
Roteiro Experimental

1. Caso não esteja utilizando a máquina virtual fornecida pelo NIC.br é preciso, antes de começar a experiência, instalar alguns softwares para auxiliar no aprendizado (caso contrário vá para o passo 2).
   Siga o passo a seguir para realizar a instalação:
   a. Para fazer algumas verificações durante o experimento será necessário a utilização do programa **Wireshark** que realiza a verificação dos pacotes que trafegam na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

   Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

2. Inicie o CORE e abra o arquivo “**Funcionalidade-Neighbor-Discovery-E4.imn**” localizado no diretório do desktop “Funcionalidades/NeighborDiscovery”, da máquina virtual do NIC.br. A seguinte topologia inicial deve aparecer:



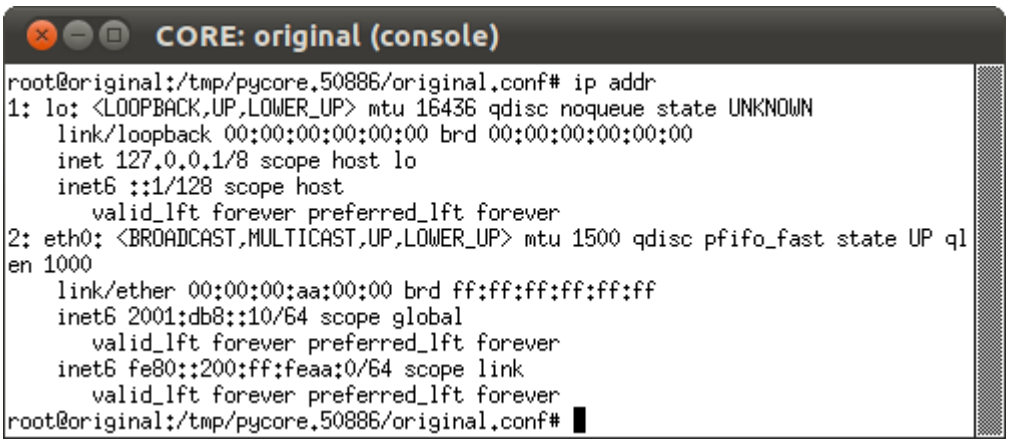
3. Verifique a configuração dos nós da topologia:
   a. Inicie a simulação realizando um dos seguintes passos:
      i. aperte o botão  ;
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do servidor *original*, com um duplo-clique.

   c. Verifique a configuração do *original* através do seguinte comando:

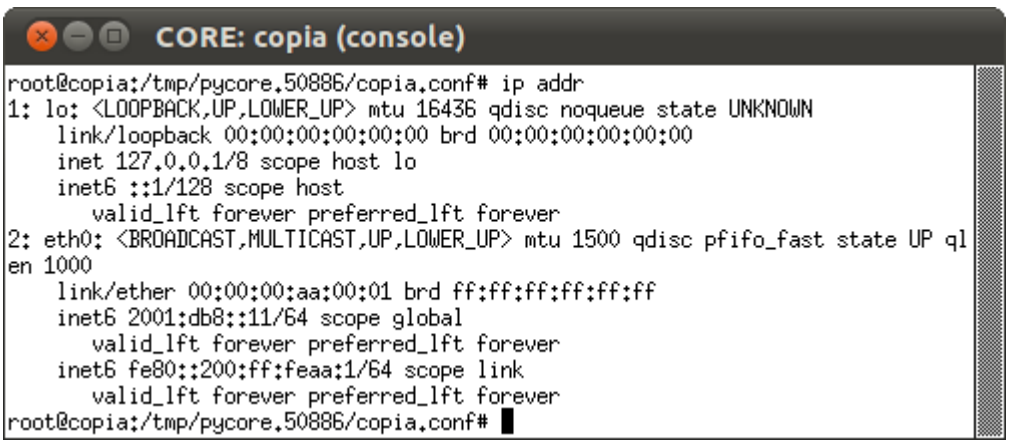
```
# ip addr
```

   O resultado deve ser:



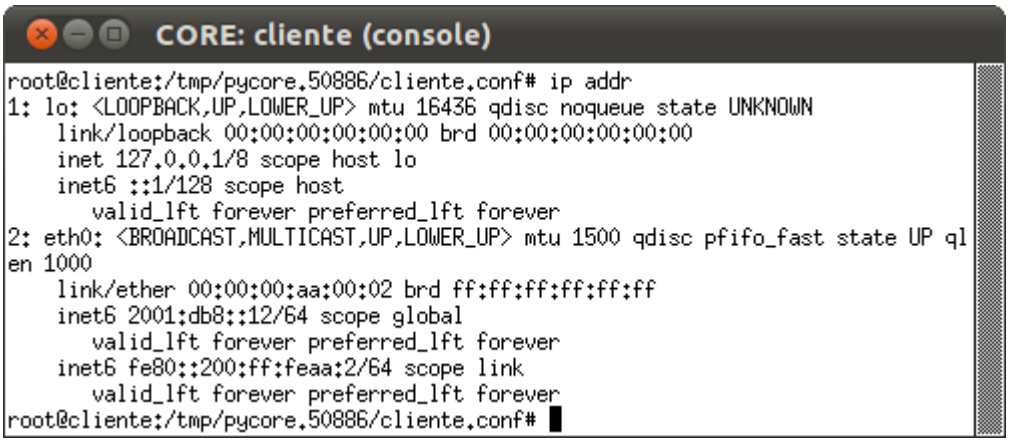
   ***Obs**: A partir desse comando é possível observar os endereços das interfaces.

   d. Verifique a configuração do servidor *copia* com o mesmo comando.
   O resultado deve ser:



   ***Obs**: A partir desse comando é possível observar os endereços das interfaces.

e. Verifique a configuração da máquina *cliente* com o mesmo comando.
O resultado deve ser:



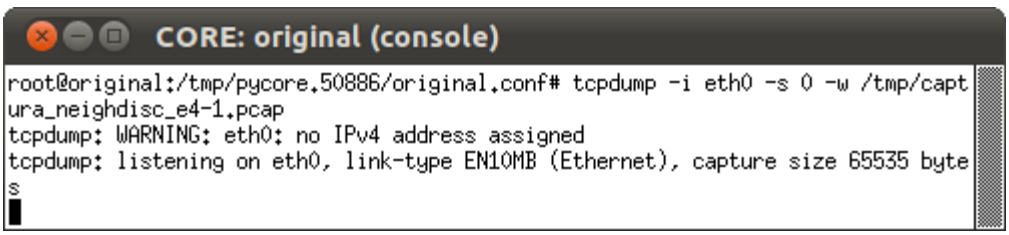
***Obs**: A partir desse comando é possível observar os endereços das interfaces.

4. Para analisar o comportamento do procolo com a duplição de IP's, o endereço da máquina *original* será copiado para a máquina *copia*:
   a. Abra o terminal da máquina *original*;

   b. Utilize o seguinte comando para iniciar a captura de pacotes enviados para esse dispositivo:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_neighdisc_e4-1.pcap
```

   O resultado deve ser:



   ***Obs**: Não feche esse terminal até o final do experimento, uma vez que, isso ocasionará no término da execução do comando "tcpdump" e prejudicará o andamento da experiência.

   c. Abra o terminal da máquina *copia*;

   d. E, em seguida, digite os seguintes comandos para trocar o endereço IPv6 pelo mesmo da máquina original:

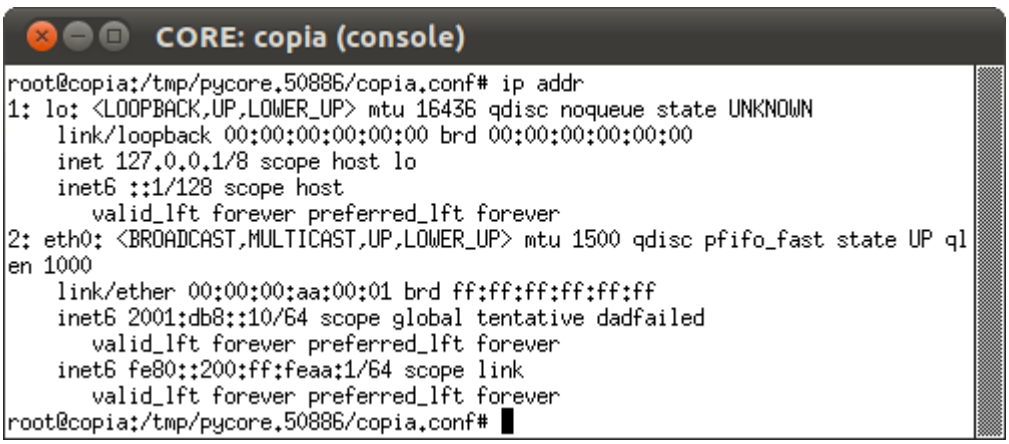
```
# ip addr del 2001:db8::11/64 dev eth0
# ip addr add 2001:db8::10/64 dev eth0
```

O resultado deve ser:

e. Verifique a nova configuração criada, através do comando:

```
# ip addr
```

O resultado deve ser:

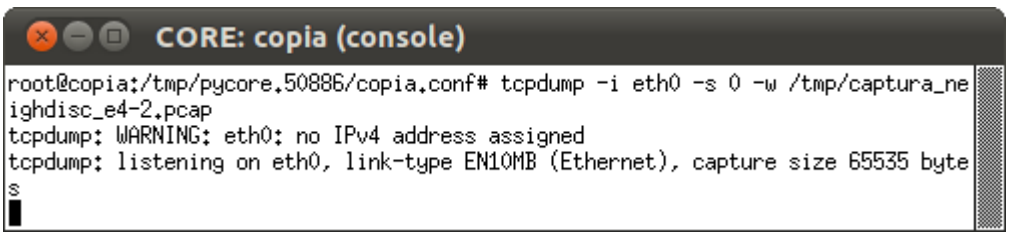


***Obs**: Observe que o endereço copiado já aparece nas configurações, contudo, isso não indica que o dispositivo o tenha aceitado ele na sua interface. Adiante veremos se ele foi ou não adicionado.

f. Em seguida utilize o seguinte comando para iniciar a captura de pacotes enviados para esse ip:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_neighdisc_e4-2.pcap
```

O resultado deve ser:



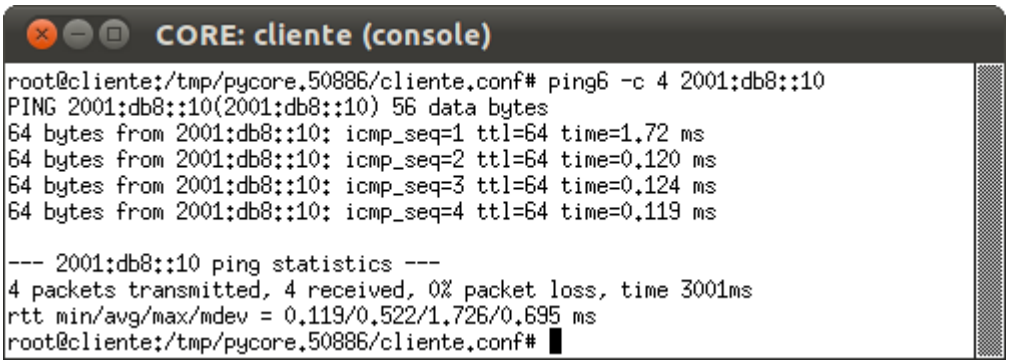
***Obs**: Não feche esse terminal até o final do experimento, uma vez que, isso ocasionará no término da execução do comando “tcpdump” e prejudicará o andamento da experiência.

5. Teste a conectividade ipv6 para o endereço duplicado.
   a. Abra um terminal da máquina *cliente*;

   b. Utilize o seguinte comando para enviar pacotes ao endereço IP duplicado:

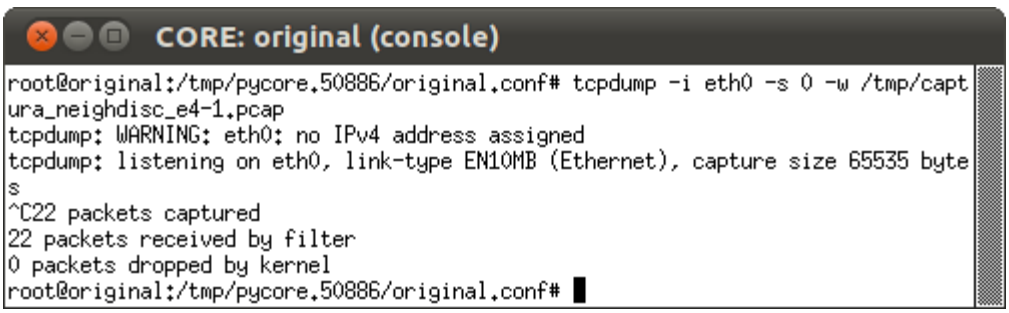
```
# ping6 -c 4 2001:db8::10
```

   O resultado deve ser:



   ***Obs**: Note que alguma máquina respondeu ao comando de ping. Adiante veremos qual das duas foi.

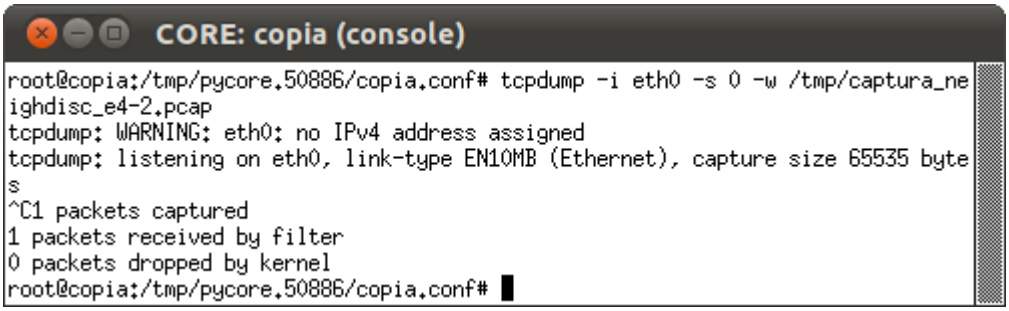
   c. No terminal do servidor *original*, encerre a captura de pacotes através da sequência Ctrl+C.
   O resultado deve ser:



   ***Obs**: A quantidade de pacotes pode variar de acordo com o tempo esperado para dar o comando Ctrl+C.

   d. No terminal do servidor *copia*, encerre a captura de pacotes utilizando a sequência Ctrl+C.

O resultado deve ser:

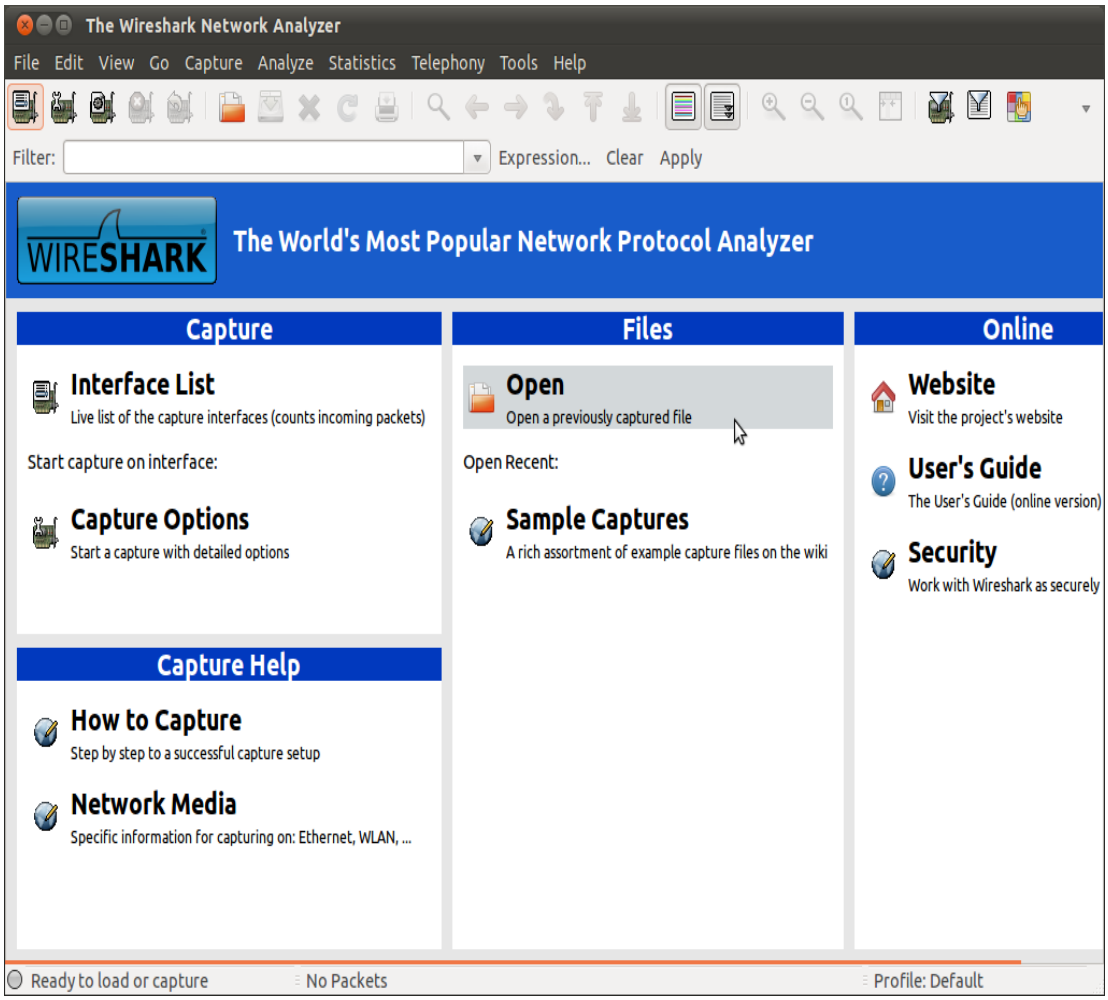


***Obs**: A quantidade de pacotes pode variar de acordo com o tempo esperado para dar o comando Ctrl+C.

6. Encerre a simulação com um dos seguintes comandos:
   a. aperte o botão  ;
   b. utilize o menu Experiment > Stop.

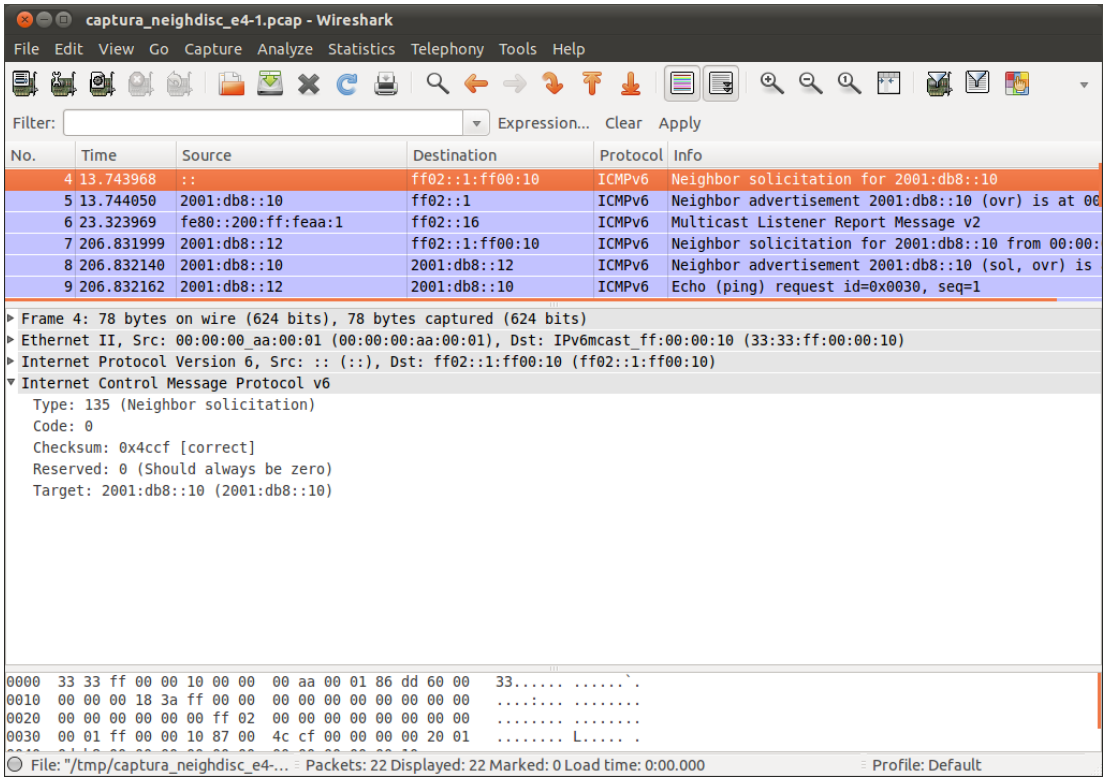
7. Para verificar os pacotes capturados, será utilizado o programa Wireshark, que pode ser aberto com a execução do seguinte comando na máquina virtual:

```
$ wireshark
```



a. Abra o arquivo da captura de pacotes da máquina *original* de nome **/tmp/captura_neighdisc_e4-1.pcap** com o menu File>Open;
b. Procure pelos pacotes de *Neighbor Solicitation* (com Source ::) , *Neighbor Advertisement* (resposta ao *Neighbor Solicitation* anterior) e *Ping Reply*. Analise-os e veja se os dados contidos nos pacotes conferem com o que foi passado na teoria.

Neighbor Solicitation:

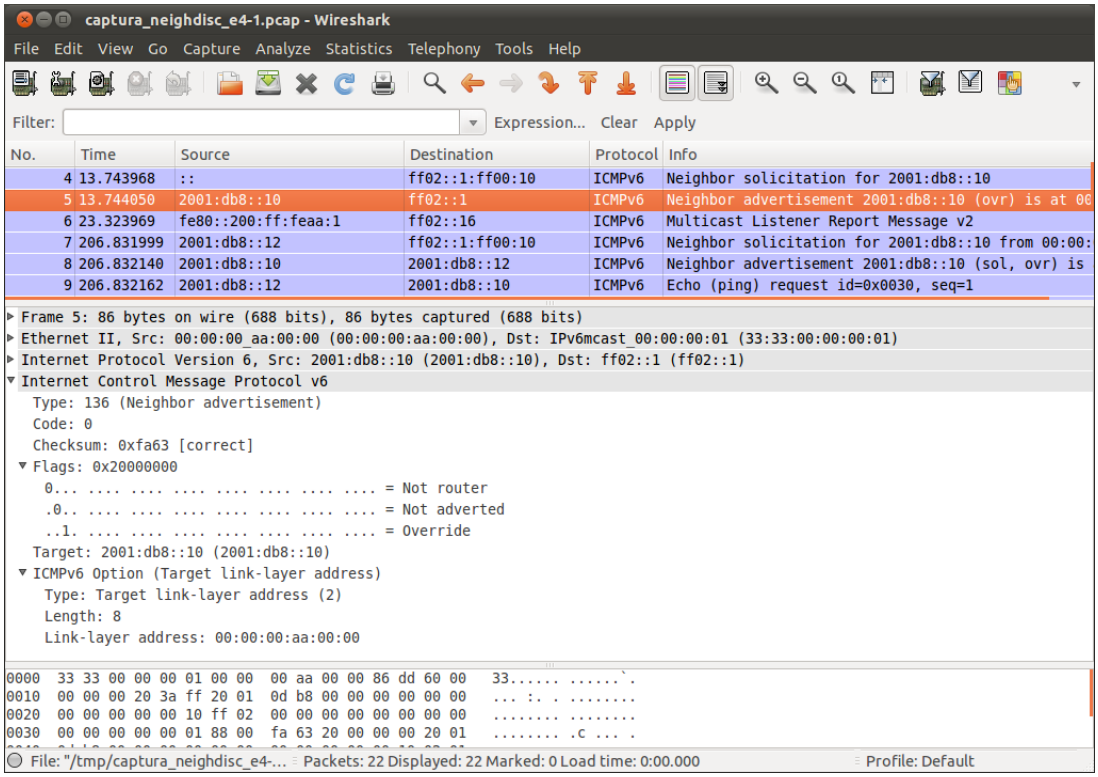


***Obs**: o filtro icmpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço (33:33:ff:00:00:10), sendo que o prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet e, o sufixo ff:00:00:10 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a origem é o MAC address da interface do dispositivo que enviou a mensagem (00:00:00:aa:00:01).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho (de extensão do IPv6), no caso, o valor 58(0x3a) refere-se à uma mensagem ICMPv6.
- **Source (IPv6)**: a origem não é especificada, endereço :: .
- **Destination (IPv6)**: o destino é o endereço multicast *Solicited-Node* (ff02::1:ff00:10).
- **Type (ICMPv6)**: indica que a mensagem é do tipo 135 (*Neighbor Solicitation*).
- **Target(ICMPv6)**: esse campo do contém o endereço IPv6 procurado (2001:db8::10).

Neighbor Advertisement:



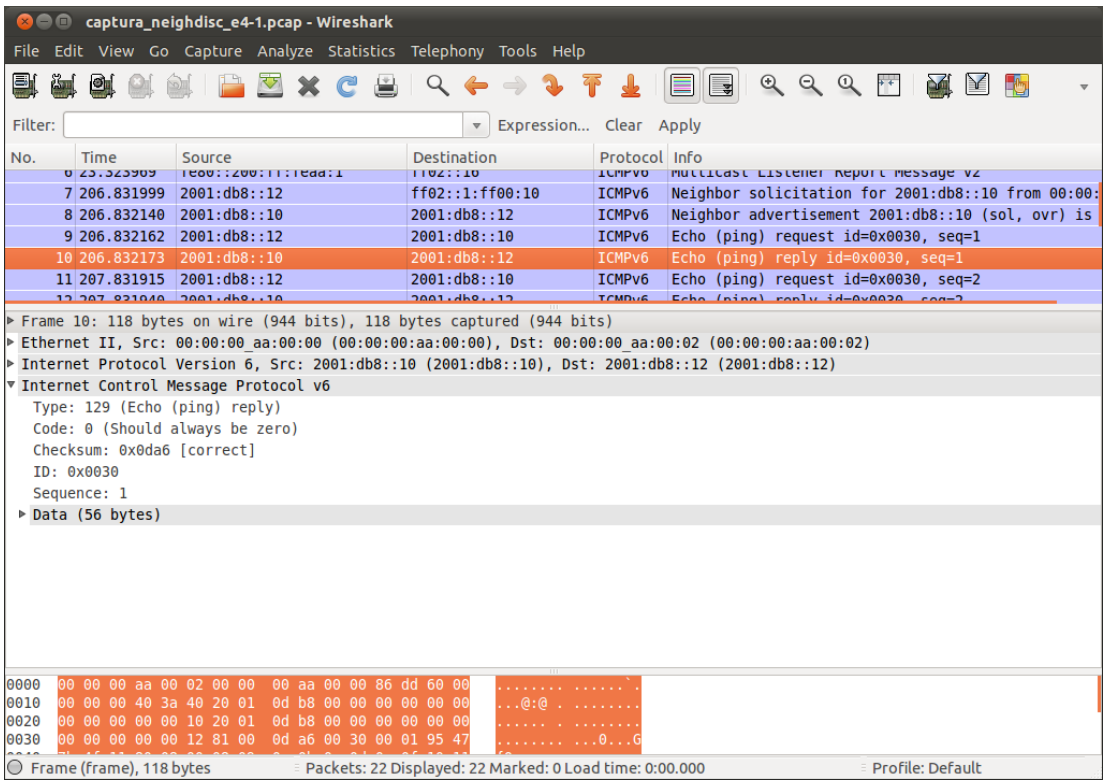
***Obs**: o filtro icmpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço MAC (33:33:00:00:00:01), sendo que o prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet e o sufixo 00:00:00:01 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a fonte é o MAC address da interface do dispositivo que enviou a resposta (00:00:00:aa:00:00).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho (de extensão do IPv6), no caso, o valor 58(0x3a) refere-se à uma mensagem ICMPv6.
- **Source (IPv6)**: a origem é o endereço IP da interface que enviou a mensagem de *Neighbor Advertisement*(2001:db8::10), em resposta ao *Neighbor Solicitation* .
- **Destination (IPv6)**: o destino é o endereço multicast *All Node* (ff02::1).
- **Type (ICMPv6)**: indica que a mensagem é do tipo 136 (*Neighbor Advertisement*).
- **Flags (ICMPv6)**: uma mensagem do tipo *Neighbor Advertisement* possui 3 flags:
    - A primeira indica se quem está enviando é um roteador. Em nosso caso, ela está marcada com 0.
    - A segunda indica se a mensagem é uma resposta a um *Neighbor Solicitation*. Contudo, ela não deve ser ativada em respostas multicast, como na presente situação.
    - A terceira indica se a informação carregada na mensagem é uma atualização de endereço de algum nó da rede. No nosso caso, está setada em 1.

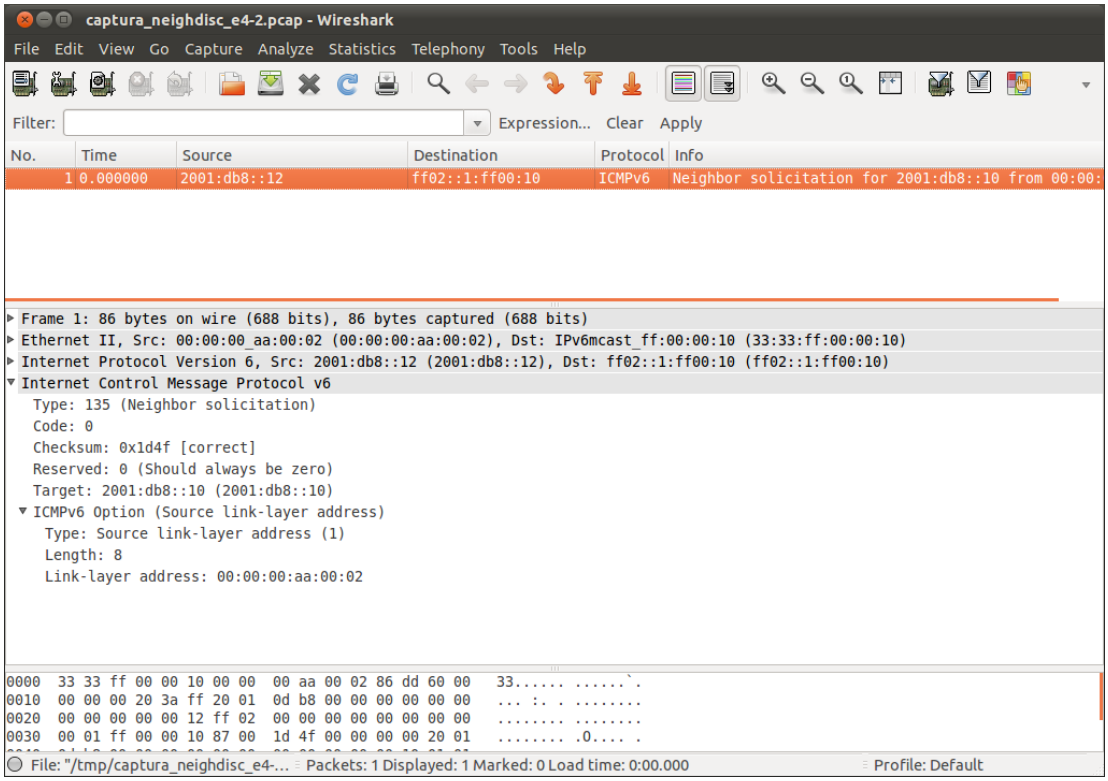
- **Target Address (ICMPv6)**: indica o endereço IP sobre o qual as flags informam. Em nosso caso, é o próprio endereço do dispositivo que gera a mensagem *Neighbor Advertisement* (máquina 'original')(2001:db8::10).
- **ICMPv6 Option (ICMPv6)**: indica as opções do pacote ICMPv6:
    - Target Link Layer Address
        - *Type*: indica o tipo de dado da mensagem ICMPv6. Em nosso caso, ela é do tipo "*Target link-layer address*".
        - *Link-layer address*: indica o MAC address da interface do dispositivo 'original' (00:00:00:aa:00:00).

Reply:



Observe que o dispositivo '*original*' respondeu todas as mensagens ping request.

c. Abra o arquivo da captura de pacotes da máquina *copia* de nome **/tmp/captura_neighdisc_e4-2.pcap** com o menu File>Open;
d. Procure por algum pacote enviado diretamente ao copia.



Observe que a interface do endereço copiado não capturou nenhum pacote direcionado ao ip copiado. As mensagens capturadas serão simplesmente direcionados ao link local ou multicast.

IPv6 - Autoconfiguração de Endereços Stateless

Objetivo

Esta experiência possui como objetivo apresentar o funcionamento da autoconfiguração stateless de endereços IPv6 através da configuração de mensagens *Router Advertisement* enviadas pelo pelo Quagga, uma plataforma de roteamento para servidores UNIX, e pelo RADVD, um daemon para sistema operacional Linux responsável por implementar o envio desse tipo de mensagem.

Para a realização do presente exercicio será utilizada a topologia descrita no arquivo: **Auto_conf-E1.imn** e **Auto_conf-E2.imn**.

Introdução Teórica

Autoconfiguração Stateless, através de mensagem *Router Advertisement,* é um procedimento utilizado por roteadores para transmitir as informações necessárias à autoconfiguração de outros nós da rede.

Essas características podem tanto ser requisitadas pelos dispositivos, com a mensagem *Router Solicitation,* quanto serem enviadas periodicamente pelos roteadores. Independente do modo, após o recebimento da mensagem *Router Advertisement*, inicia-se o procedimento *Duplicate Address Detection* para evitar a criação de endereços repetidos no enlace.

Este protocolo é denominado *stateless,* pois, o dispositivo que fornece informações de configuração não mantém o registro do estado e caracteristicas do nó destinatário, enquanto o nó destino se encarrega de sua própria configuração.

Roteiro Experimental

Experiência1 - Quagga - Router Advertisement

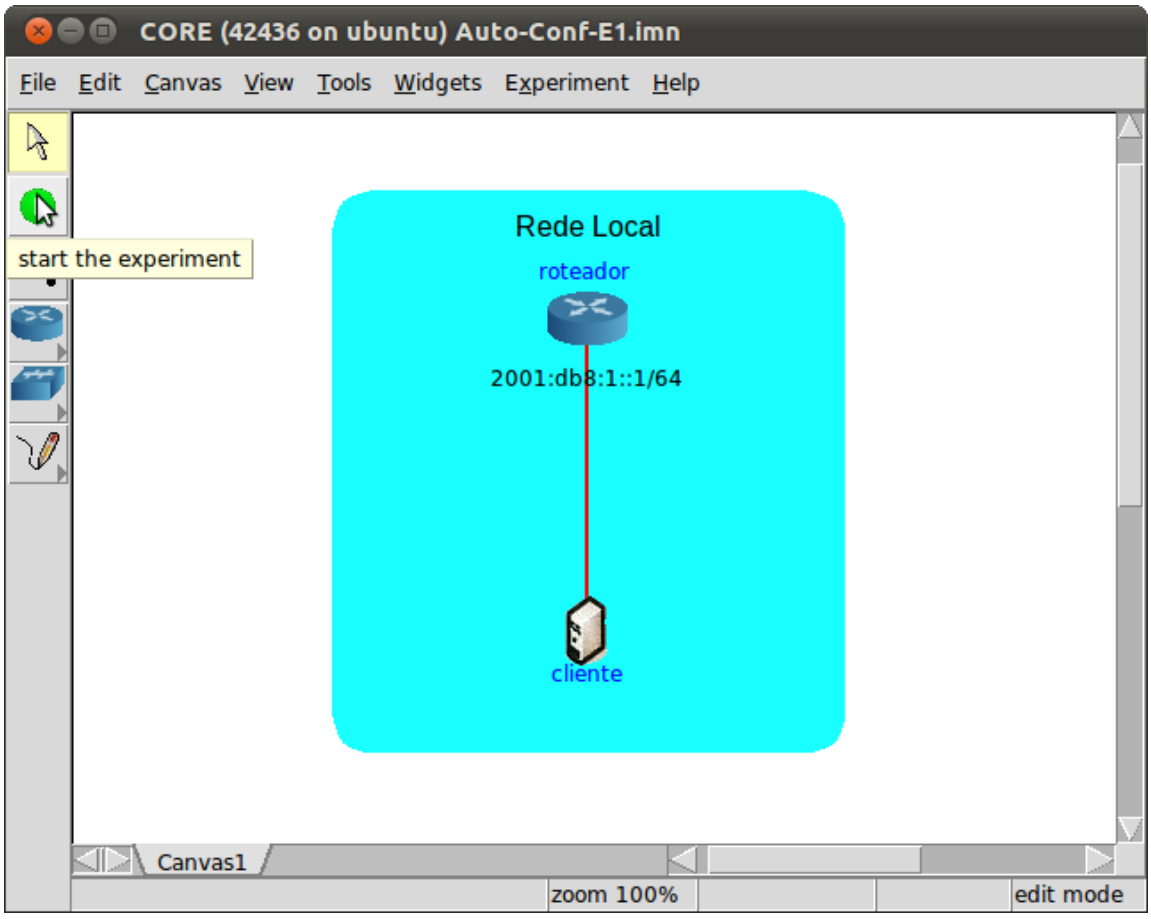
1. Caso não esteja utilizando a máquina virtual fornecida pelo NIC.br é preciso, antes de começar a experiência, instalar alguns softwares para auxiliar no aprendizado (caso contrário vá para o passo 2). Siga o passo a seguir para realizar a instalação:

a. Para fazer algumas verificações durante o experimento será necessário a utilização do programa **Wireshark**, que melhora a visualização de pacotes transmitidos na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

2. Inicie o CORE e abra o arquivo “**Auto_Conf-E1.imn**” localizado no diretório do desktop “Funcionalidades/AutoStateless”, da máquina virtual do NIC.br. A seguinte topologia inicial de rede deve aparecer deve aparecer:



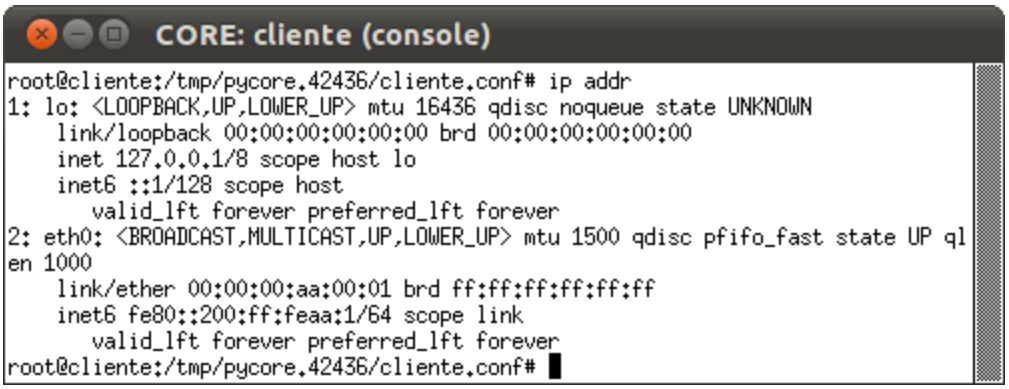
3. Verifique a configuração dos nós da topologia.
   a. Inicie a simulação realizando um dos seguintes passos:
      i. aperte o botão  ;
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do ‘cliente’ através do duplo-clique.

   c. Observe a configuração do ‘cliente’ com o seguinte comando:

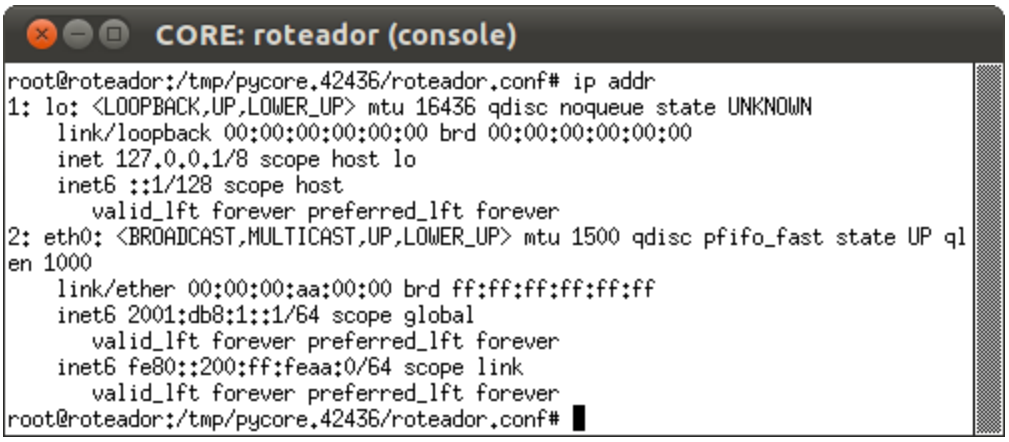
```
# ip addr
```

   O resultado deve ser:



   ***Obs**: A partir desse comando é possível observar os endereços das interfaces.

   d. Observe a configuração do ‘roteador’ com o mesmo comando.
   O resultado deve ser:



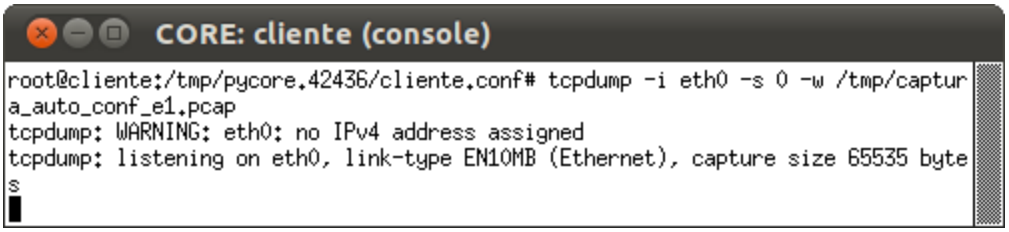
   ***Obs**: A partir desse comando é possível observar os endereços das interfaces.

4. Edite as configurações do Quagga, para enviar mensagens *Router Advertisement* contendo informações de configuração para o cliente.
   a. Abra o terminal do cliente com um duplo-clique.

   b. Utilize o seguinte comando para iniciar a captura de pacotes do roteador:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_auto_conf_e1.pcap
```

   O resultado deve ser:



   ***Obs**: Não feche esse terminal até o final do experimento, uma vez que, isso ocasionará no término da execução do comando "tcpdump" e prejudicará o andamento da experiência.

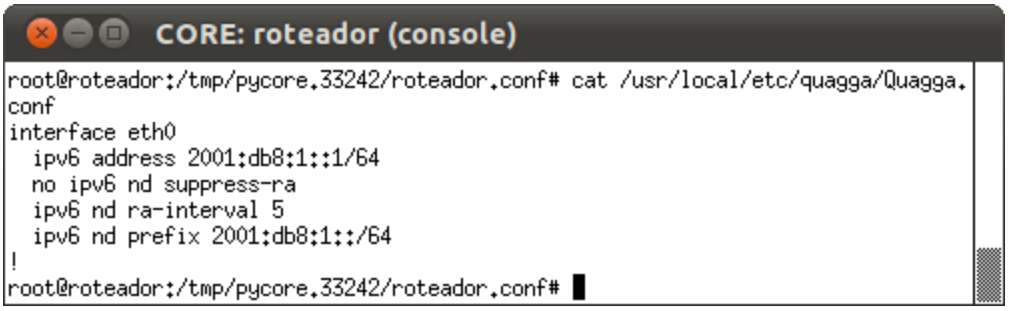
   c. Abra o terminal do roteador com um duplo-clique.

   d. Substitua o conteúdo do arquivo Quagga.conf localizado na pasta /usr/local/etc/quagga pelo seguinte:

```
interface eth0
  ipv6 address 2001:db8:1::1/64
  no ipv6 nd suppress-ra
  ipv6 nd ra-interval 5
  ipv6 nd prefix 2001:db8:1::/64
!
```

   ***Obs**: Na versão atual do quagga instalado no CORE, não é possivel enviar o DNS via *Router Advertisement*.
   Mais informações sobre essa configuração podem ser encontradas em:
   http://www.nongnu.org/quagga/docs/docs-info.html#SEC140

O resultado deve ser:



***Obs**: um editor de texto presente na máquina virtual que pode ser utilizado é o **nano**. Para usá-lo digite no terminal:

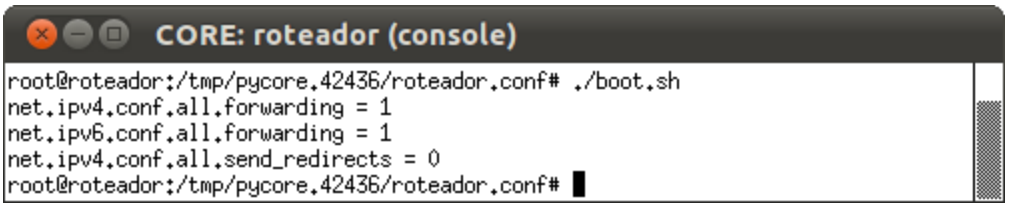
```
# nano /usr/local/etc/quagga/Quagga.conf
```

No nano, a sequência utilizada para salvar o arquivo é CTRL-O e para sair é CTRL-X.

e. Em seguida, execute o script que re-inicia as configurações do quagga. Ele se encontra na pasta base do 'roteador' (pasta inicial quando se abre o terminal):

```
# ./boot.sh
```

O resultado deve ser:

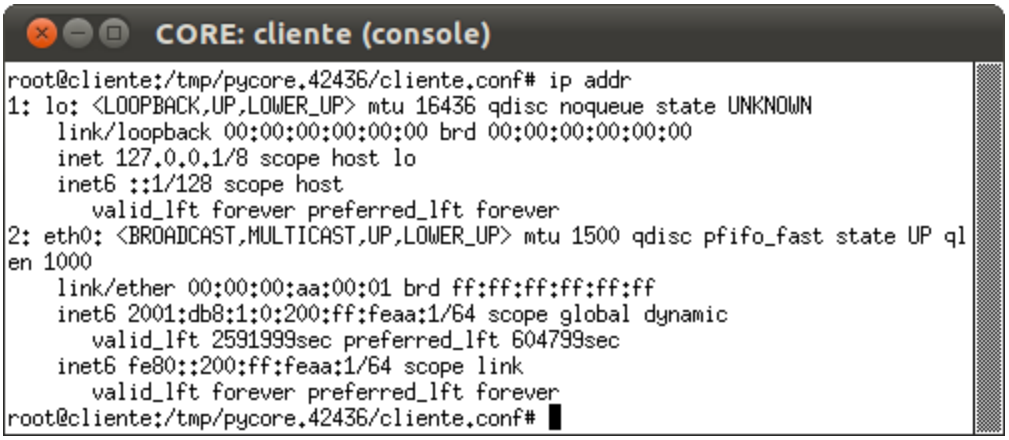


5. Teste a conectividade IPv6 de um nó ao outro.
   a. Abra o terminal do 'cliente' com um de duplo-clique.

   b. Espere alguns segundos e digite o seguinte comando para observar o endereço adquirido:

```
# ip addr
```

O resultado deve ser:



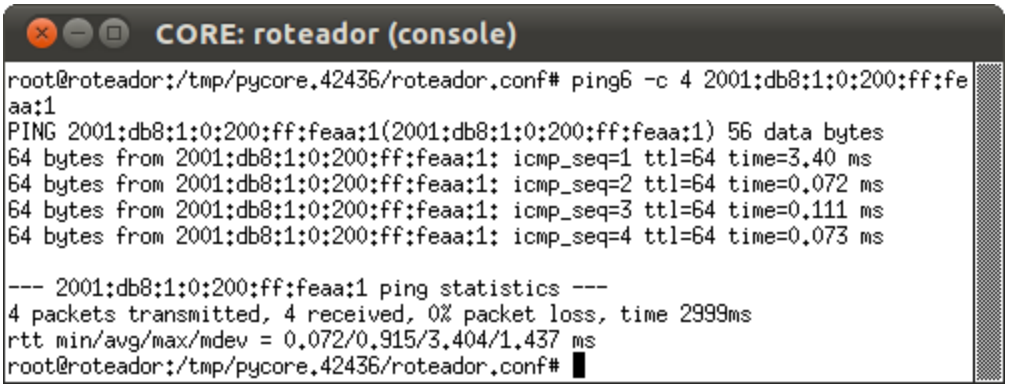
***Obs**: Note a existência do endereço de escopo global na interface eth0.

c. Em seguida, abra o terminal do roteador com um duplo-clique.

d. Inicie o envio de pacotes para o novo ip, para testar a conectividade através do seguinte comando:

```
# ping6 -c 4 2001:db8:1:0:200:ff:feaa:1
```

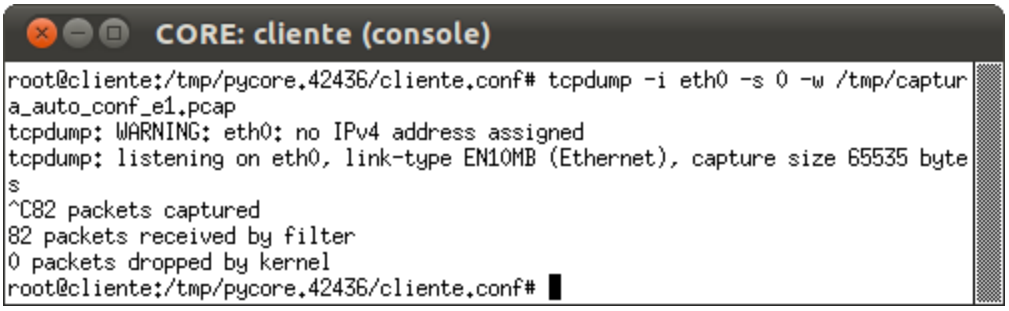
O resultado deve ser:



***Obs**: o ip deve ser o do cliente, obtido pelo comando anterior.

e. No terminal do cliente, encerre a captura de pacotes através da sequência Ctrl+C.

O resultado deve ser:



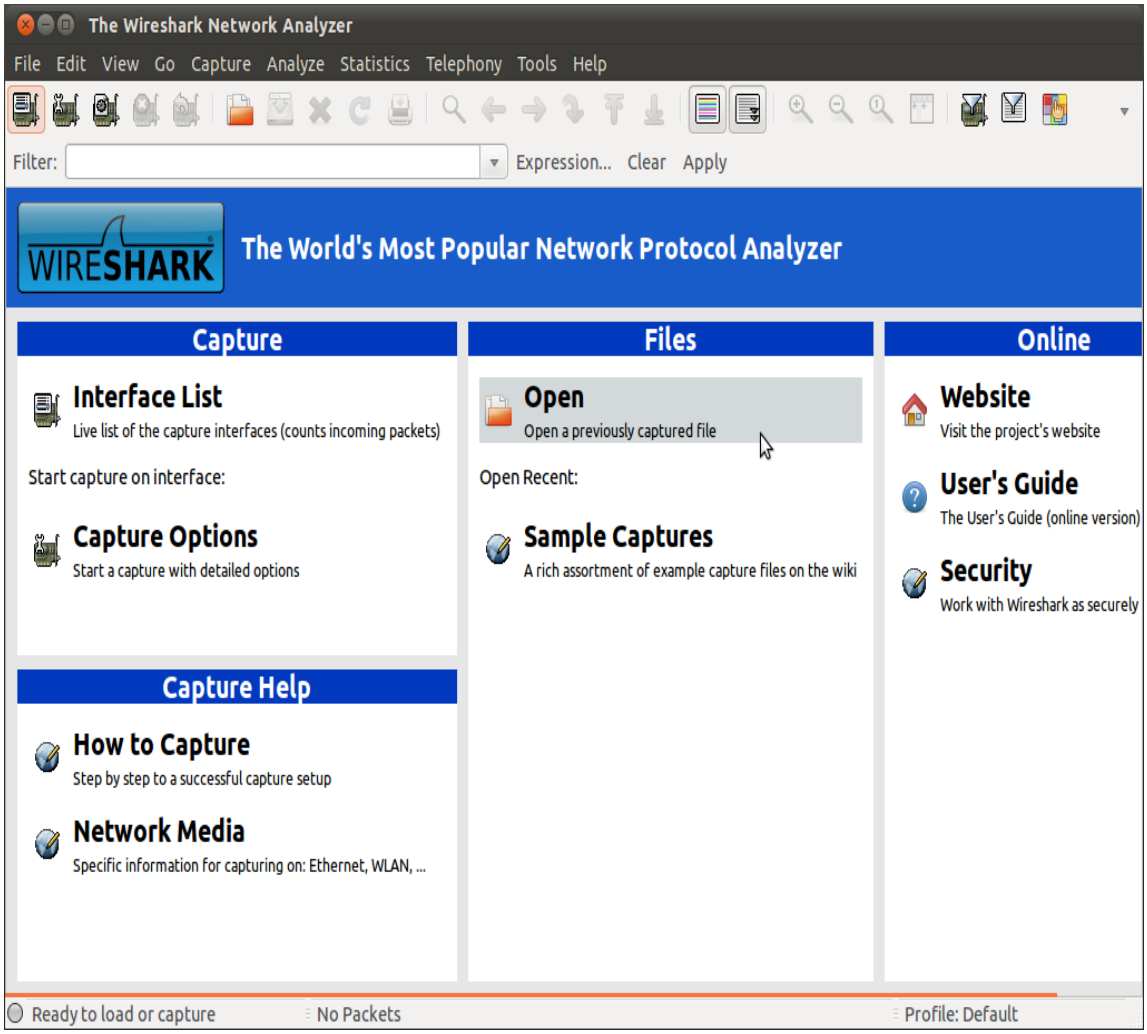
***Obs**: A quantidade de pacotes pode variar de acordo com o tempo esperado para dar o comando Ctrl+C.

6. Encerre a simulação com uma das seguintes ações:

   a. aperte o botão  ;
   b. utilize o menu Experiment > Stop

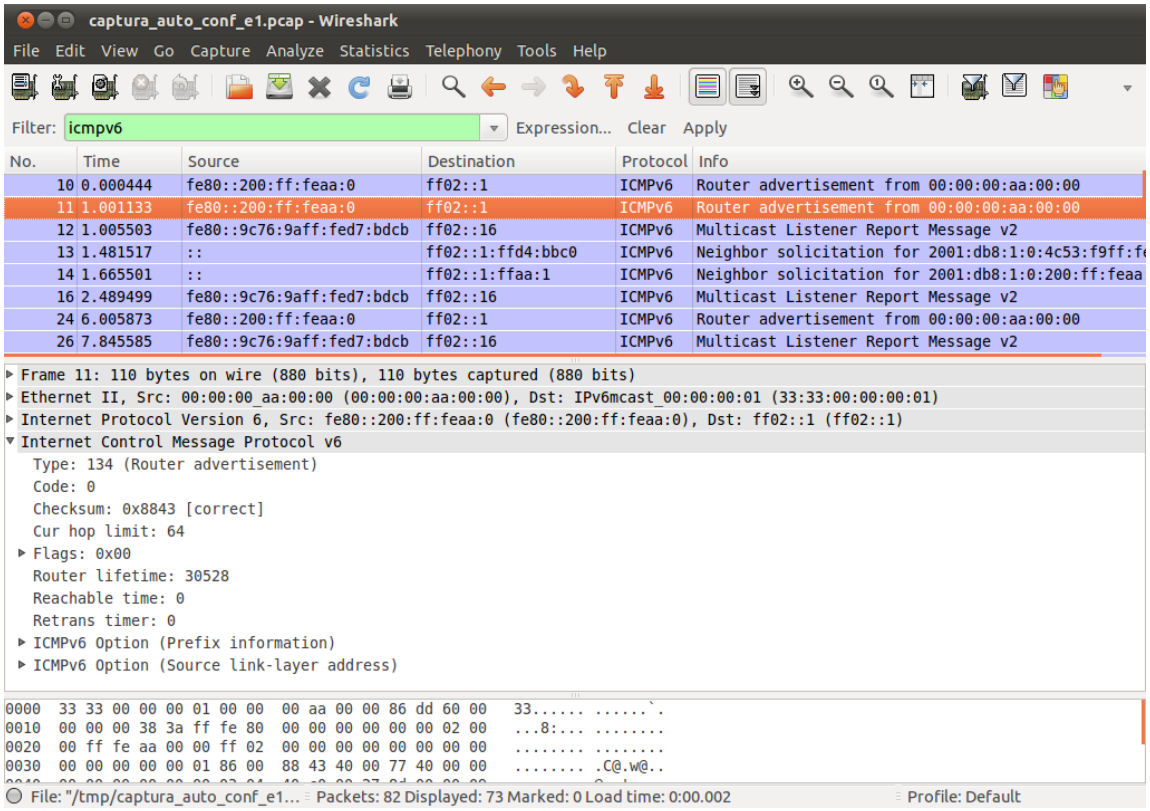
7. A verificação dos pacotes capturados será realizada através do programa Wireshark. Para iniciá-lo execute o seguinte comando em um terminal da máquina virtual:

```
$ wireshark
```



a. Abra o arquivo **/tmp/captura_auto_conf_e1.pcap** com o menu File>Open:
b. Procure por um pacote *Router Advertisement* que contenha o protocolo opcional *Prefix Information*. Analise-o e veja que os dados contidos no pacote conferem com o que foi passado na teoria.

Router Advertisement:



***Obs**: o filtro icmpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço MAC (33:33:00:00:00:01) sendo que o prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet e, que o sufixo 00:00:00:01 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a origem é o MAC Address da interface do roteador que enviou a mensagem (00:00:00:aa:00:00).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho (de extensão do IPv6), no caso, o valor 58(0x3a) refere-se à uma mensagem ICMPv6.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface que envia a mensagem (fe80::200:ff:feaa:0).
- **Destination (IPv6)**: o destino é o endereço *Multicast All Nodes* (ff02::1).
- **Type (ICMPv6)**: indica que a mensagem é do tipo 134 (Router Advertisement).
- **ICMPv6 Option (ICMPv6)**: indica as opções do pacote ICMPv6:
    - Prefix Information
        - *Type*: contém o valor 3 que identifica o "*Prefix Information*".
        - *Autonomous Address-Configuration Flag (A)*: indica se o prefixo deve ser

utilizado para autoconfiguração stateless (1) .

- *Preferred Lifetime*: marca o tempo, em segundos, que o endereço é preferencial, ou seja, um endereço que pode ser utilizado indistintamente. O valor (0xffffffff) indica infinito.
- *Valid Lifetime*: marca o tempo, em segundos, de expiração do endereço gerado. O valor (0xffffffff) indica infinito.
- *Prefix*: contém o prefixo de rede a ser utilizado (2001:db8:1::).
- *Prefix length*: contém o tamanho do prefixo da rede.

- Source Link Layer Address
  - *Type*: indica o tipo de dado da mensagem ICMPv6. Em nosso caso, ela é do tipo "*Source link-layer address*";
  - *Link-layer address*: indica o MAC address do endereço de origem da mensagem.

Roteiro Experimental

Experiência2 - Radvd - Router Advertisement

Experiência

1. Caso não esteja utilizando a máquina virtual fornecida pelo NIC.br é preciso, antes de começar a experiência, instalar alguns softwares para auxiliar no aprendizado (caso contrário vá para o passo 2). Siga o passo a seguir para realizar a instalação:
    a. Para fazer algumas verificações durante o experimento será necessário a utilização do programa **Wireshark**, que melhora a visualização de pacotes transmitidos na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

b. O **radvd**, que realiza o envio do tamanho do valor de MTU a ser configurado. Para instalá-lo use o comando:

```
$ sudo apt-get install radvd
```

Caso haja algum problema no pacote oferecido, pode-se baixa-lo no endereço: http://packages.ubuntu.com/hardy/radvd (acessado em 2012).

c. O **ndisc6**, que permite a descoberta do valor MTU configurado no link. Para instalá-lo use o comando:

```
$ sudo apt-get install ndisc6
```

Caso haja algum problema no pacote oferecido, pode-se baixa-lo no endereço: https://launchpad.net/ubuntu/maverick/+package/ndisc6 (acessado em 2012).

d. O **rdnssd**, que permite a captura de dados sobre o dns em um nó cliente. Para instalá-lo use o comando:

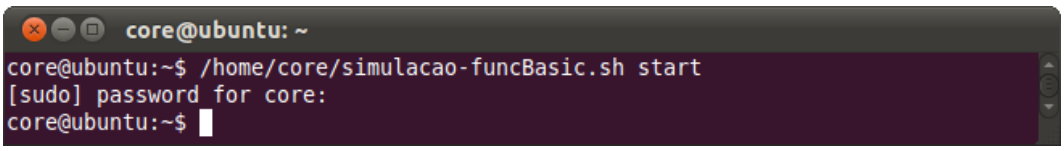
```
$ sudo apt-get install rdnssd
```

Caso haja algum problema no pacote oferecido, pode-se baixa-lo no endereço: http://packages.debian.org/squeeze/rdnssd (acessado em 2012).

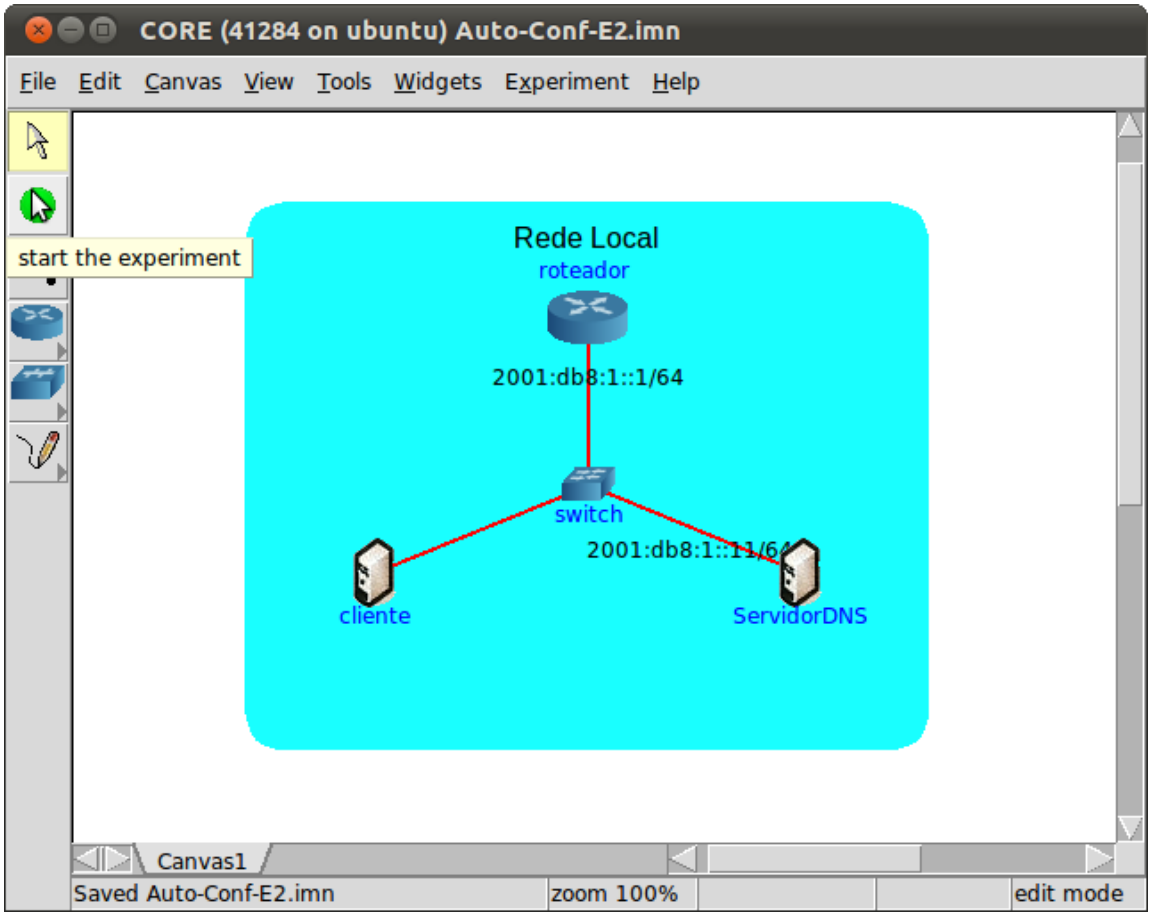
2. Inicie a configuração da simulação.
    a. Num terminal da máquina virtual digite o seguinte comando:

```
$ /home/core/simulacao-funcBasic.sh start
```

Caso necessário, digite “core” como senha para prosseguir com a configuração. O resultado deve ser:



3. Inicie o CORE e abra o arquivo “**Auto-Conf-E2.imn**” localizado no diretório do desktop “Funcionalidades/AutoStateless”, da máquina virtual do NIC.br. A seguinte topologia inicial de rede deve aparecer:



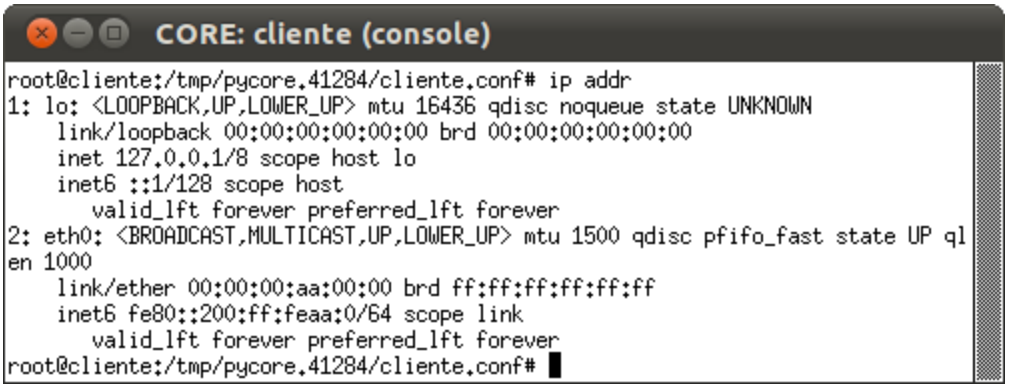
4. Verifique a configuração dos nós da topologia.
    a. Inicie a simulação realizando um dos seguintes passos:
        i. aperte o botão  ;
        ii. utilize o menu Experiment > Start.

    b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do cliente com um duplo-clique.

    c. Observe a configuração do 'cliente' com o seguinte comando:

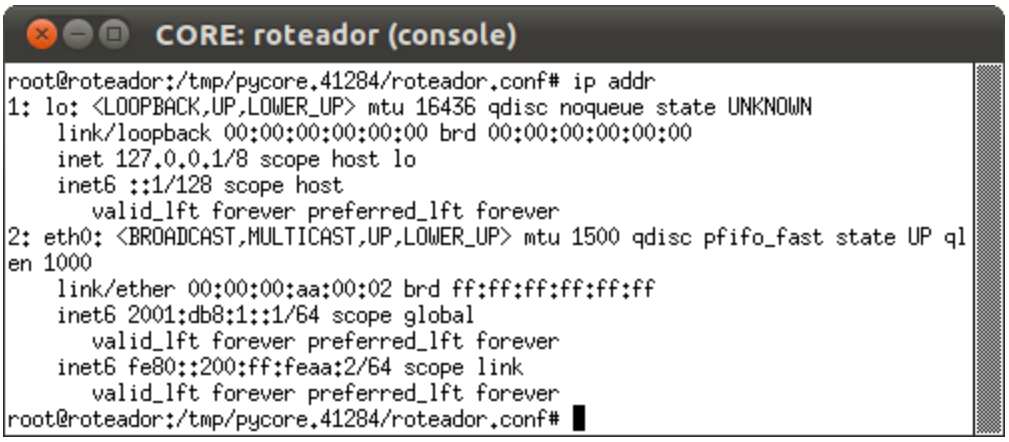
```
# ip addr
```

    O resultado deve ser:



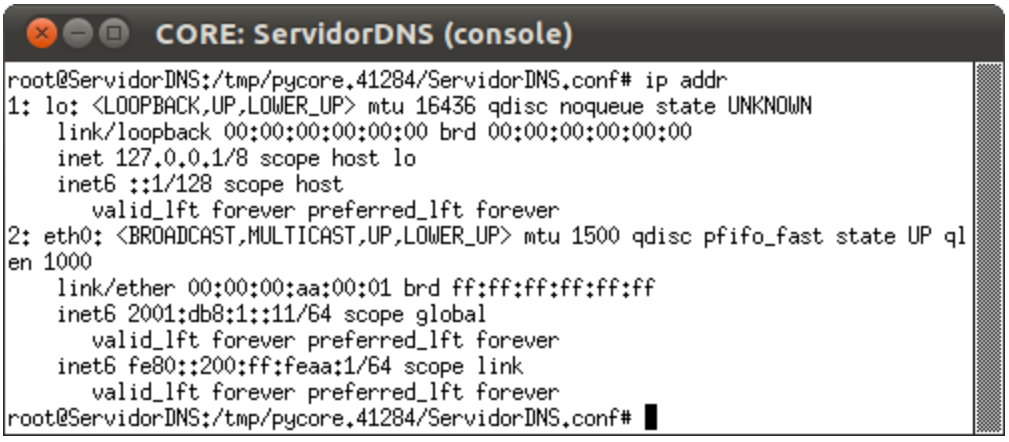
***Obs**: A partir desse comando é possível observar os endereços das interfaces.

d. Observe a configuração do 'roteador' utilizando o mesmo comando. O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces.

e. Observe a configuração do 'ServidorDNS' utilizando o mesmo comando. O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces.

5. Inicie o serviço DNS no 'ServidorDNS'.
   a. Abra o terminal do 'ServidorDNS' com um duplo-clique.

   b. Utilize o seguinte comando para iniciar o serviço DNS:

```
# dnsmasq -i eth0
```

   O resultado deve ser:

6. Configure o cliente para capturar as configurações do DNS.
   a. Abra o terminal do 'cliente' através do duplo-clique.

   b. Utilize o seguinte comando para iniciar o programa que coleta informações de DNS.

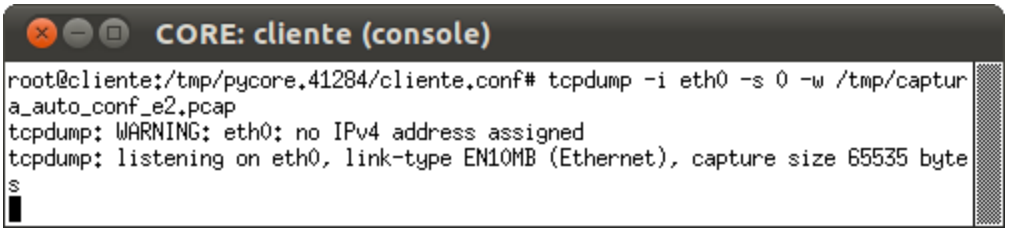
```
# /etc/init.d/rdnssd start
```

   O resultado deve ser:

7. Edite as configurações do RADVD, para enviar a mensagem Router Advertisement contendo informções de configuração para o cliente.
   a. No terminal do 'cliente', utilize o seguinte comando para iniciar a captura de pacotes do roteador:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_auto_conf_e2.pcap
```

O resultado deve ser:



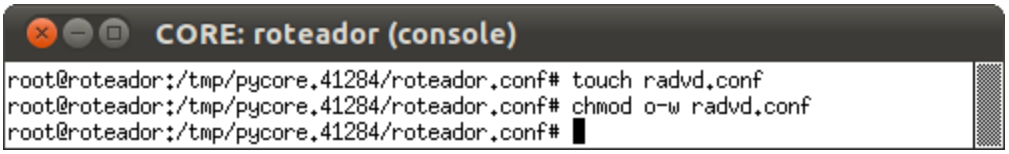
***Obs**: Não feche esse terminal até o final do experimento, uma vez que, isso ocasionará no término da execução do comando "tcpdump" e prejudicará o andamento da experiência.

b. Abra o terminal do 'roteador' com um duplo-clique.

c. Crie um arquivo dentro da pasta base do roteador sem premissão de escrita para outros usuários com os seguintes comandos:

```
# touch radvd.conf
# chmod o-w radvd.conf
```

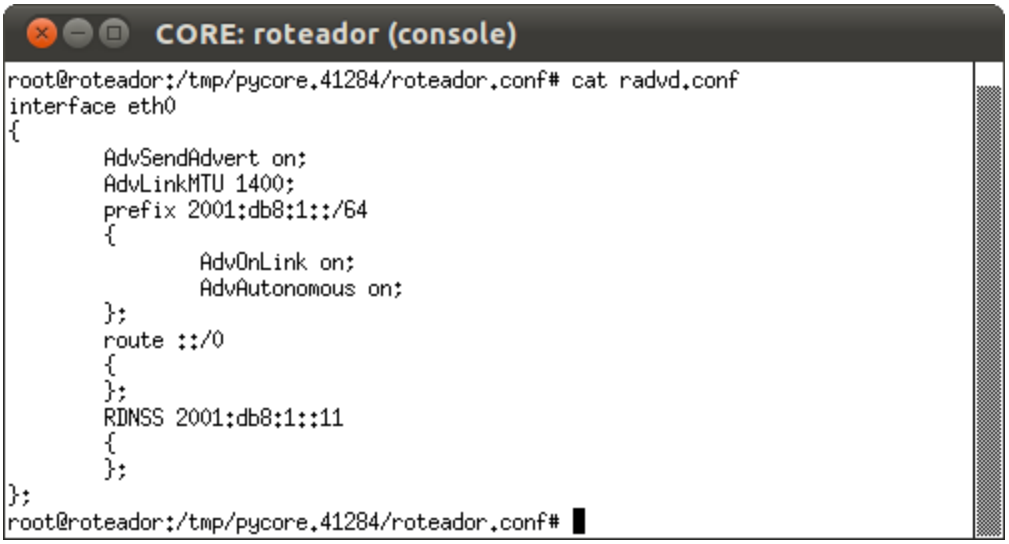
O resultado deve ser:



d. Edite o arquivo criado para que ele contenha as seguintes linhas:

```
interface eth0
{
      AdvSendAdvert on;
      AdvLinkMTU 1400;
      prefix 2001:db8:1::/64
      {
            AdvOnLink on;
            AdvAutonomous on;
      };
      route ::/0
      {
      };
      RDNSS 2001:db8:1::11
      {
      };
};
```

*Obs: Mais informações sobre essa configuração podem ser encontradas em: http://linux.die.net/man/5/radvd.conf (Acessado em 26/04/2012)
O resultado deve ser:



*Obs: um editor de texto presente na máquina virtual que pode ser utilizado é o **nano**. Para usá-lo digite no terminal:

```
# nano radvd.conf
```

No nano, a sequência utilizada para salvar o arquivo é CTRL-O e para sair é CTRL-X.

e. Depois, inicie o programa radvd com essas configurações através do comando:

```
# radvd -C radvd.conf
```

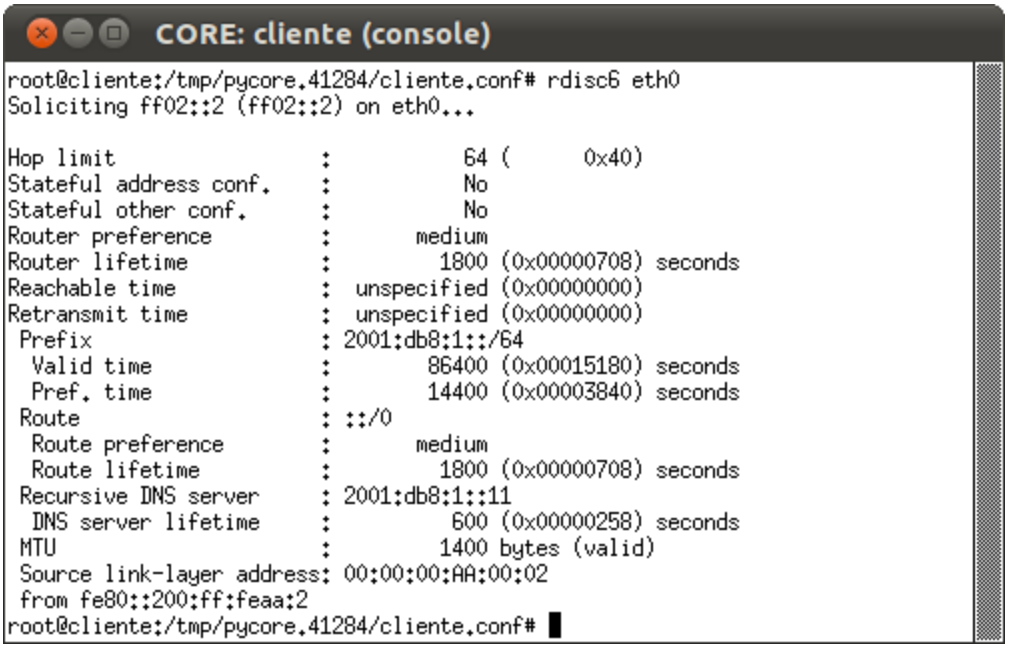
O resultado deve ser:

8. Verifique as configurações enviadas pelo 'roteador' aos outros nós da rede:
   a. Abra outro terminal do 'cliente' através de duplo-clique.

   b. Digite o seguinte comando para observar as características do enlace:

```
# rdisc6 eth0
```

O resultado deve ser:



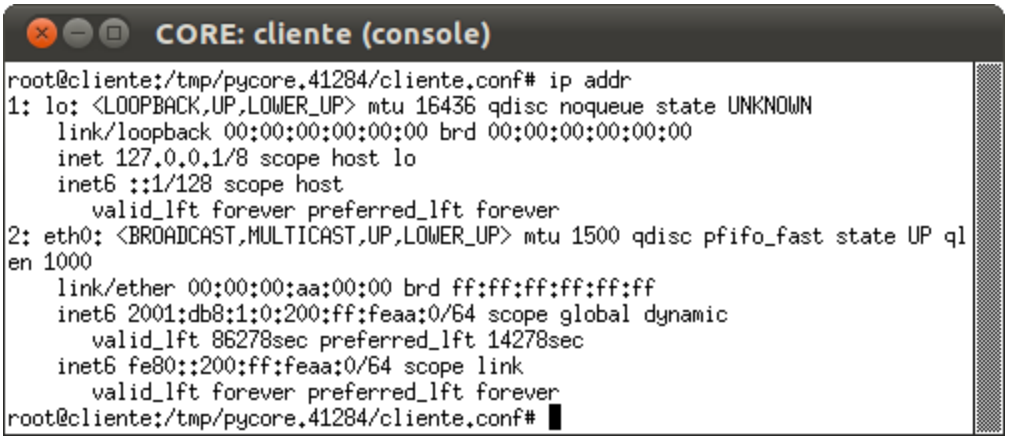
***Obs**: Note os dados enviados pelo roteador ao cliente.

9. Teste a conectividade IPv6 entre o 'cliente' e o 'ServidorDNS'.
   a. Abra o terminal do 'cliente' através do duplo-clique.

   b. Verifique sua configuração através do comando:

```
# ip addr
```

O resultado deve ser:

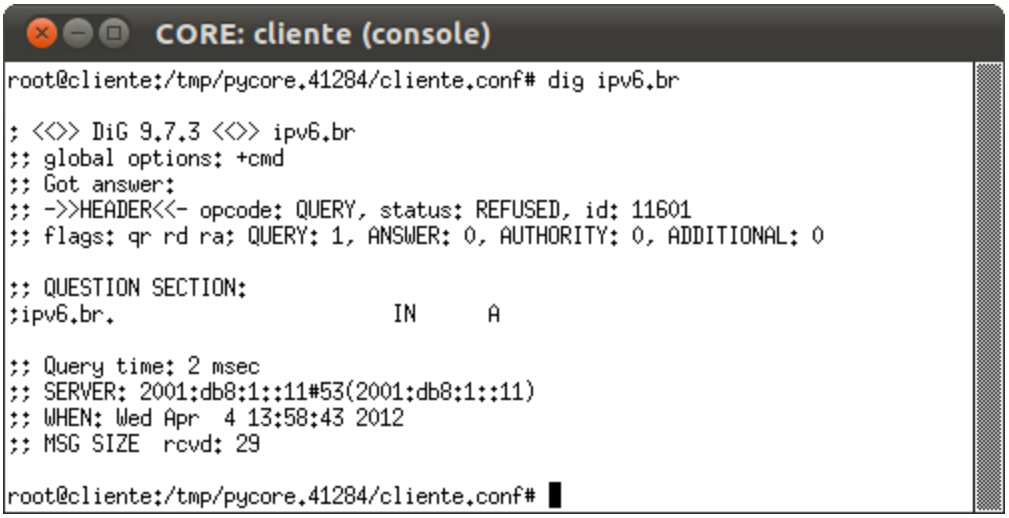


***Obs**: Note o endereço obtido via autoconfiguração Stateless.

c. Teste o serviço DNS através do comando:

```
# dig ipv6.br
```

O resultado deve ser:

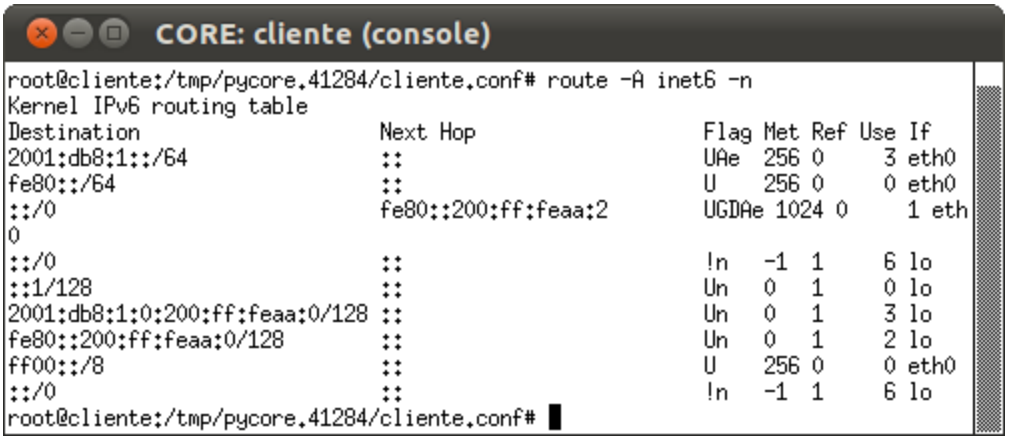


***Obs**: Note que o servidor DNS respondeu a requisição do cliente.

d. Verifique as rotas utilizadas através do comando:

```
# route -A inet6 -n
```

O resultado deve ser:

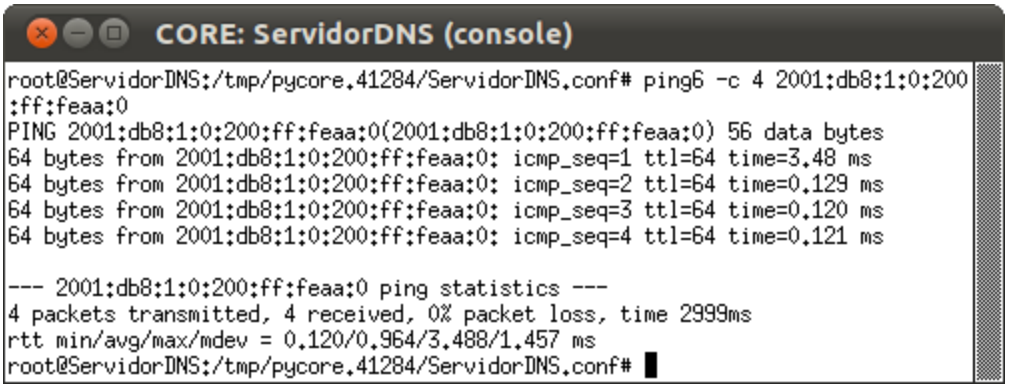


e. Abra o terminal do 'ServidorDNS' com um duplo-clique.

f. Teste a conectividade com o cliente do servidor DNS utilizando o comando:

```
# ping6 -c 4 2001:db8:1:0:200:ff:feaa:0
```

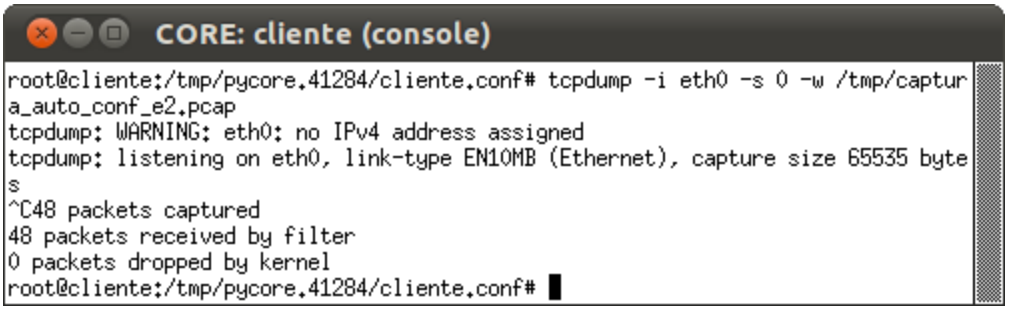
O resultado deve ser:



***Obs**: Note que o endereço ip obtido é comunicável na rede.

g. No terminal do cliente, encerre a captura de pacotes através da sequência Ctrl+C.

O resultado deve ser:



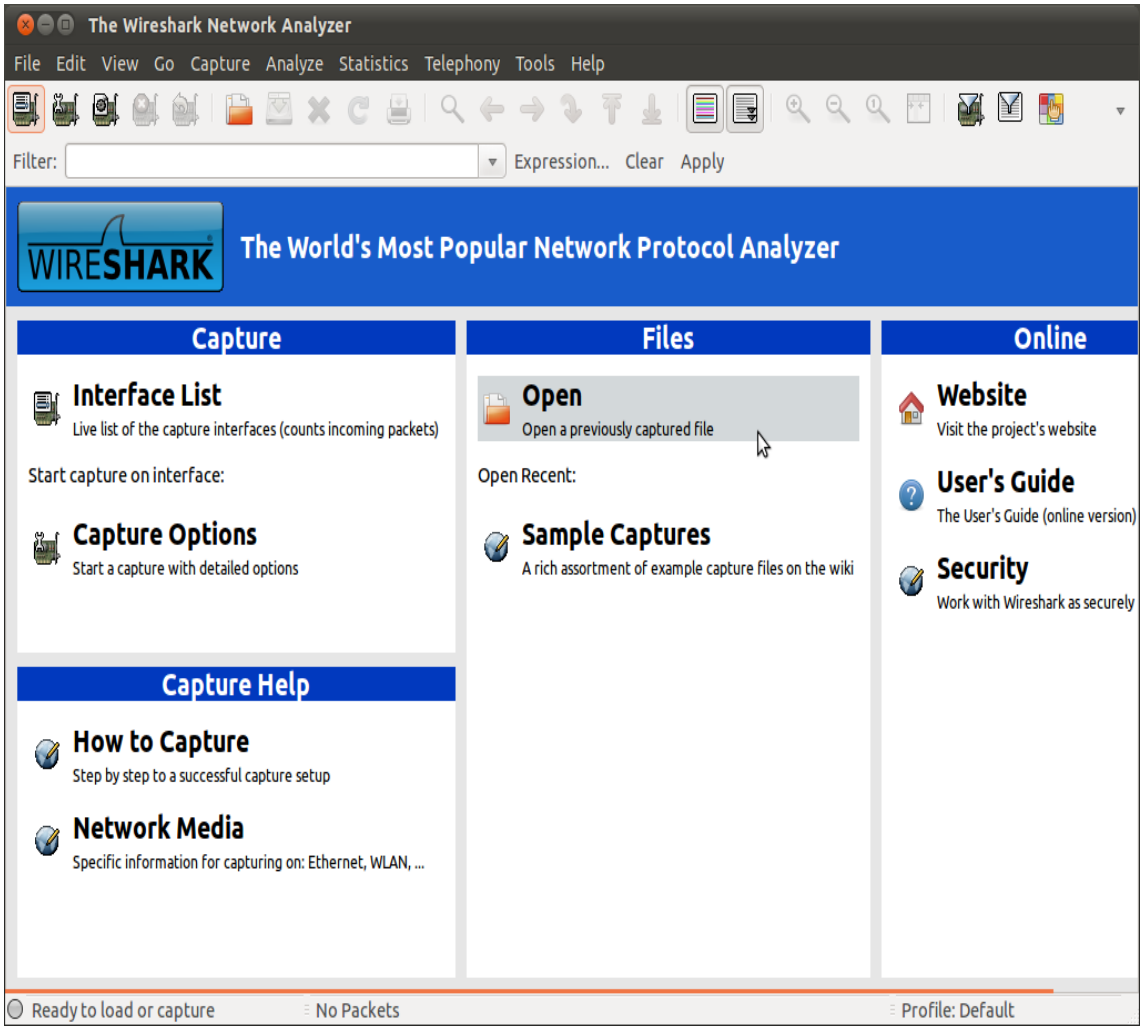
***Obs**: A quantidade de pacotes pode variar de acordo com o tempo esperado para dar o comando Ctrl+C.

10. Encerre a simulação com uma das seguintes ações:

    a. aperte o botão  ;
    b. utilize o menu Experiment > Stop.

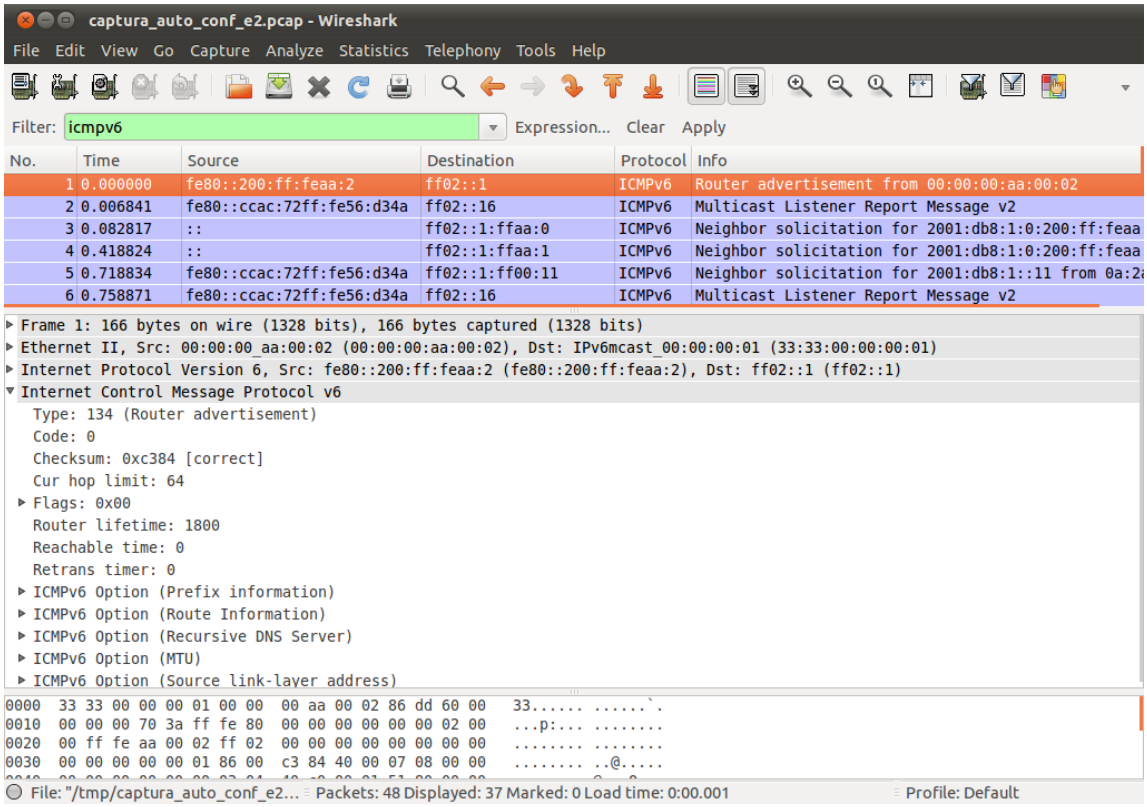
11. A verificação dos pacotes capturados será realizada através do programa Wireshark. Para iniciá-lo execute o seguinte comando em um terminal da máquina virtual:

```
$ wireshark
```



a. Abra o arquivo **/tmp/captura_auto_conf_e2.pcap** com o menu File>Open:
b. Procure um pacote do tipo *Router Advertisement* que contenha o protocolo opcional *Prefix Information*. Analise-o e veja que os dados contidos nos pacotes conferem com o que foi passado na teoria.

Router Advertisement:



***Obs**: o filtro icmpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço MAC (33:33:00:00:00:01), sendo que o prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet e o sufixo 00:00:00:01 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a origem é o endereço MAC da interface do roteador (00:00:00:aa:00:02).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho (de extensão do IPv6), no caso, o valor 58(0x3a) refere-se à uma mensagem ICMPv6.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface que enviou a mensagem (fe80::200:ff:feaa:2).
- **Destination (IPv6)**: o destino é o endereço *Multicast All nodes* (ff02::1).
- **Type (ICMPv6)**: indica que a mensagem é do tipo 134 (Router Advertisement).
- **ICMPv6 Option (ICMPv6)**: indica as opções do pacote ICMPv6:
    - Prefix Information
        - *Type: indica o tipo de dado da mensagem ICMPv6. No nosso caso, ela é do tipo "Prefix information", 3;*
        - *Autonomous Address-Configuration Flag (A):* indica se o prefixo deve ser

utilizado para autoconfiguração stateless (1) .

- *Preferred Lifetime*: marca o tempo, em segundos, que o endereço é preferencial, ou seja, um endereço que pode ser utilizado indistintamente. O valor (0xffffffff) indica infinito.
- *Valid Lifetime*: marca o tempo, em segundos, de expiração do endereço gerado. O valor (0xffffffff) indica infinito.
- *Prefix*: contém o prefixo de rede a ser utilizado (2001:db8:1::).
- *Prefix Length*: contém o tamanho bits do Prefixo informado

- Source Link Layer Address
  - *Type*: indica o tipo de dado da mensagem ICMPv6. No nosso caso, ela é do tipo “Source link-layer address”, 1;
  - *Link-layer address*: indica o MAC address da interface que originou a mensagem.
- Route Information
  - *Type*: indica o tipo de dado da mensagem ICMPv6. No nosso caso, ela é do tipo “Route Information”, 24.
  - *Prefix*: indica um prefixo que pode ser alcançavel por esse roteador (::0).
  - *Length*: indica o tamanho desse prefixo (24).
- Recursive DNS Server
  - *Type*: indica o tipo de dado da mensagem ICMPv6. No nosso caso, ela é do tipo “Recursive DNS Server”, 25;
  - *Recursive DNS Sever*: indica o endereço IPv6 do servidor DNS configurado (2001:db8:1::11);
- MTU
  - *Type*: indica o tipo de dado da mensagem ICMPv6. No nosso caso, ela é do tipo “MTU”, 5;
  - *MTU*: indica o tamanho máximo a ser utilizado naquele enlace (1400).

IPV6 - DHCPv6

Experiência1 - DHCPv6 Full - Solicit, Advertise, Request e Reply

Objetivo

Esta experiência tem como objetivo apresentar o funcionamento do DHCPv6 stateful, ou seja, que envia dados de configurações opcionais, DNS, e o endereço IPv6.

Para a realização do presente exercicio será utilizada a topologia descrita no arquivo: **DHCPv6-E1.imn**.

Introdução Teórica

O Dynamic Host Configuration Protocol (DHCP) é um protocolo de autoconfiguração stateful, utilizado para distribuir endereços IP e informações de rede dinamicamente. Contudo, suas implementações IPv6 possuem significativas diferenças e particularidades com relação ao IPv4, o que torna estas implementações incompatíveis entre si. Nessa experiência, apenas o DHCPv6 operando como stateful será observado.

A arquitetura cliente-servidor é utilizada como base do funcionamento desse protocolo. Em cada rede, deve haver um servidor capaz de decidir sobre a configuração de cada uma das interfaces de rede presentes. Na pratica, a comunicação entre o servidor DHCP e as máquinas cliente se dá com a troca de quatro mensagens:

- **Solicit** é enviada pelo cliente, com endereço *Multicast Agent DHCP* (ff02::1:2), para a rede com o intuito de encontrar o Servidor DHCP;
- **Advertise** é enviada pelo Servidor DHCP, diretamente ao endereço de link local do cliente, para indicar que ele pode fornecer as informações de configuração necessárias;
- **Request** é enviada pelo cliente diretamente ao Servidor DHCP para requisitar os dados de configuração;
- **Reply** é enviada pelo Servidor DHCP ao endereço de link local do cliente como resposta à mensagem Request.

Existe uma configuração para o cliente, chamada *rapid commit*, que permite a troca de informações com apenas duas mensagens. Contudo, ela só é aconselhavel quando a rede possui apenas um servidor ou, quando existem muitos endereços a serem resolvidos.

Caso existam roteadores na rede, algumas particularidades no mecanismo DHCPv6

acontecem, a começar pelo envio, a partir dos roteadores, de informações sobre as caracteristicas do serviço através das mensagens do tipo *Router Advertisement*. Nelas, são ativadas duas opções: *AdvManagedFlag*, que define a permissão do recebimento do endereço IPv6, por meio do servidor DHCPv6, e o *AdvOtherConfigFlag*, que habilita o recebimento de outras configurações vindas servidor DHCPv6.

Há duas maneiras para se retirar a influência dos roteadores sobre o funcionamento do DHCP, uma realizada no lado do cliente e ou outra no roteador. No lado do cliente, é preciso selecionar a opção de não aceitar *Router Advertisement.* Já o roteador precisa ser configurado para não enviar a mensagem *Router Advertisement*.

A outra particularidade, que ocorre com presença de roteadores na rede, acontece caso algum roteador esteja localizado entre o cliente e o servidor. Nela, o roteador fica também encarregado de traduzir as mensagens *multicast* enviadas pelo cliente para encontrar o servidor DHCP.

Roteiro Experimental

1. Caso não esteja utilizando a máquina virtual fornecida pelo NIC.br é preciso, antes de começar a experiência, instalar alguns softwares para auxiliar no aprendizado (caso contrário vá para o passo 2). Siga o passo a seguir para realizar as instalações:
    a. Para fazer algumas verificações durante o experimento será necessário a instalação do programa **Wireshark** que facilita a visualização dos pacotes enviados na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

    Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

    b. O **dhcpd**, que é o serviço reponsável pelas tarefas de servidor do DHCP. Para instalá-lo, baixe a última versão do pacote ‘*tar.gz*’ no site http://www.isc.org/software/dhcp (acessado em 10/04/2012) e utilize os seguintes comandos em um Terminal:

```
$ cd <pasta onde o arquivo foi baixado>
$ tar xf dhcp-4.2.3-P2.tar.gz
$ cd dhcp-4.2.3-P2/
$ ./configure
$ make
$ sudo make install
```

    ***Obs**: Lembre-se de utilizar os numeros corretos de versão para extrair o pacote

e acessar a pasta com os arquivos de instalação. Depois do comando ‘`sudo`‘ será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

c. O **dibbler-client**, que realiza as funções de cliente DHCP. Para instalá-lo, baixe a última versão (Acima da 0.8.2) do código fonte no site http://klub.com.pl/dhcpv6/ (acessado em 10/04/2012) e utilize os seguintes comandos em um Terminal:

```
$ cd <pasta onde o arquivo foi baixado>
c.    $ tar xf dibbler-0.8.2.tar.gz
d.    $ cd dibbler-0.8.2/
e.    $ ./configure
f.    $ make
g.    $ sudo make install
```

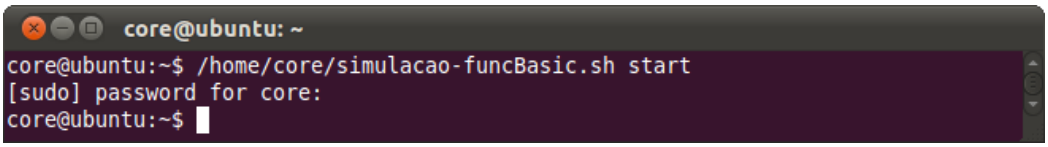
***Obs**: Lembre-se de utilizar os números corretos de versão para extrair o pacote e acessar a pasta com os arquivos de instalação. Depois do comando ‘`sudo`‘ será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

2. Inicie a configuração de simulação:
   a. Num terminal da máquina virtual digite o seguinte comando:

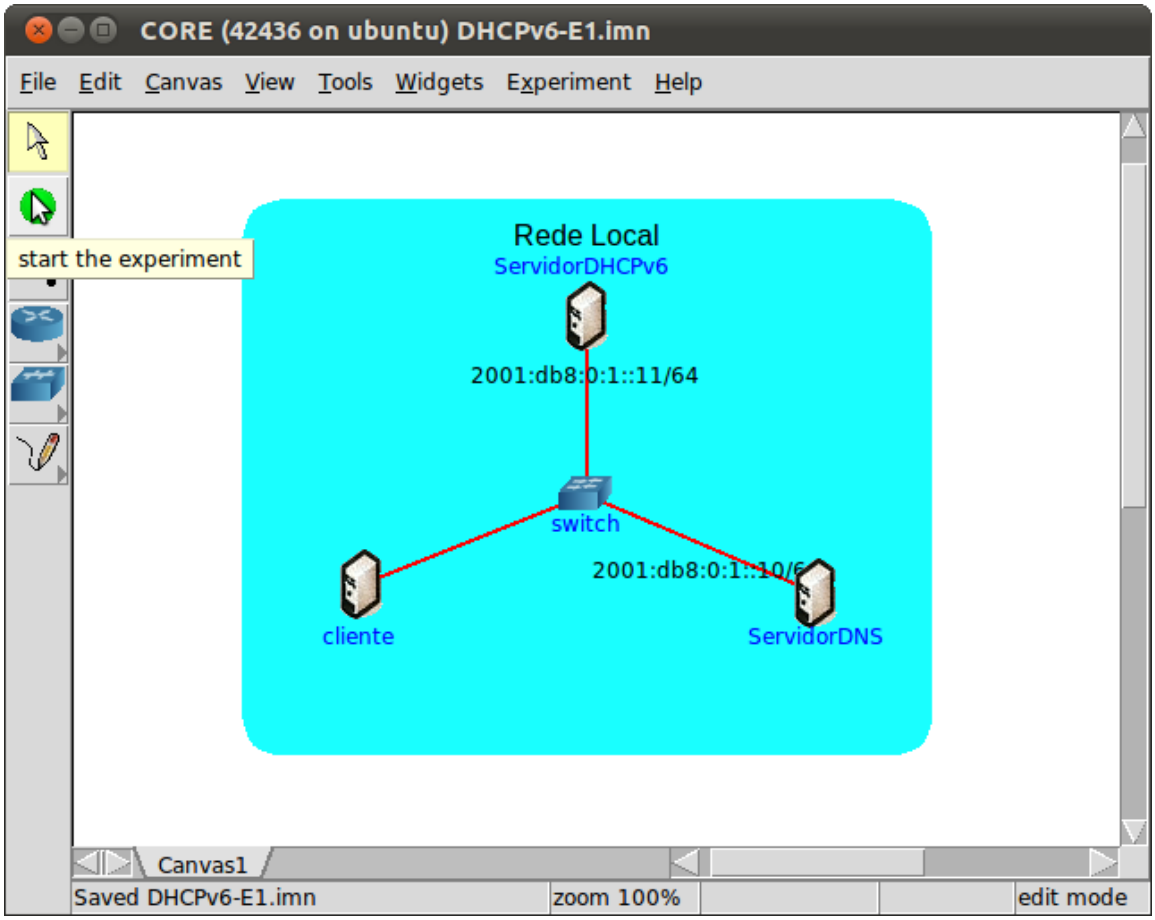
```
$ /home/core/simulacao-funcBasic.sh start
```

Caso necessário, digite “core” para prosseguir com a configuração.

O resultado deve ser:



3. Inicie o CORE e abra o arquivo “**DHCPv6-E1.imn**” localizado no diretório do desktop “Funcionalidades/DHCPv6”, da máquina virtual do NIC.br. A seguinte topologia deve aparecer:



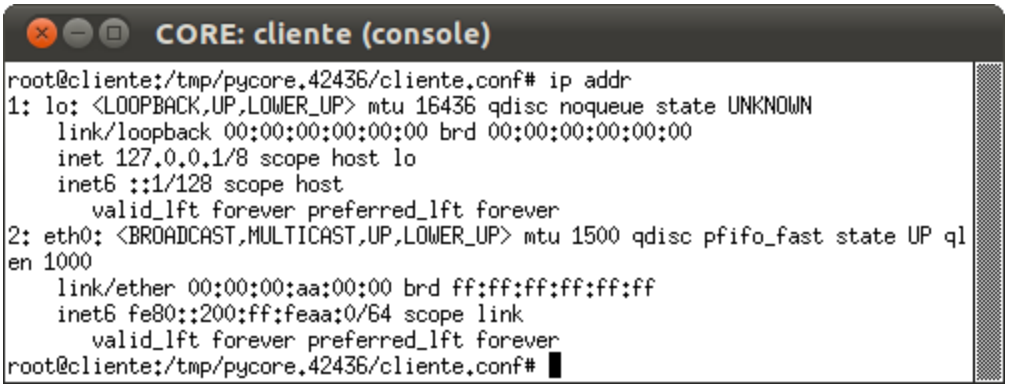
4. Verifique a configuração dos nós da topologia.
   a. Inicie a simulação realizando um dos seguintes passos:
      i. clique no botão  ;
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine de iniciar a simulação e abra o terminal da máquina cliente, com um duplo-clique sobre ela.

   c. Observe a configuração de rede do cliente através do seguinte comando:

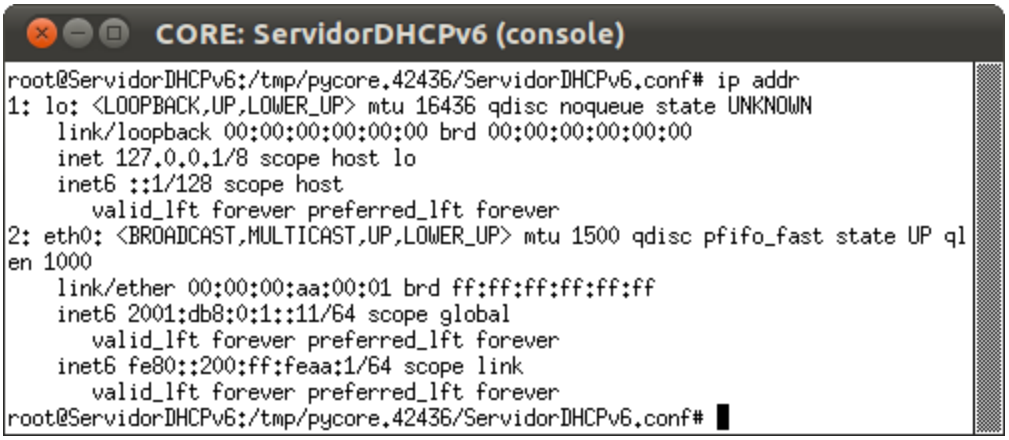
```
# ip addr
```

O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces.

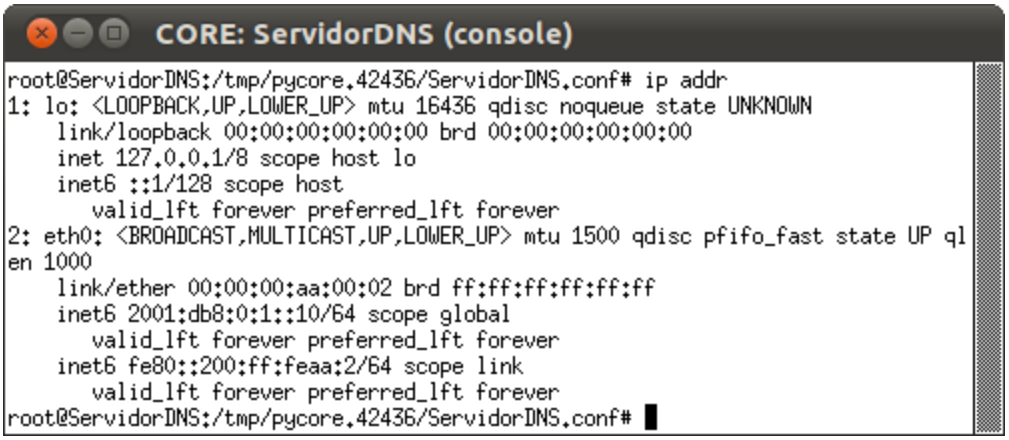
d. Observe a configuração do ServidorDHCPv6 com o mesmo comando.
O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces.

e. Observe a configuração do ServidorDNS com o mesmo comando.

O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces.

5. Inicie o serviço DNS no ServidorDNS.
   a. Abra o terminal do ServidorDNS, com um duplo-clique.

   b. Utilize o seguinte comando para iniciar o serviço DNS:

```
# dnsmasq -i eth0
```

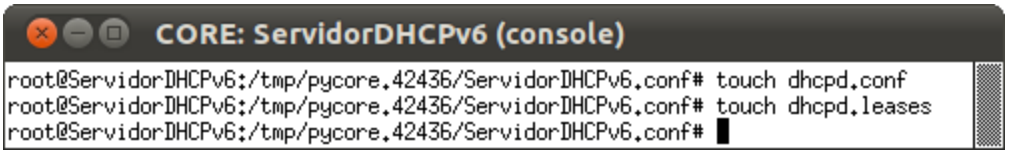
O resultado deve ser:

6. Configure o dhcp no Servidor para enviar as configurações ao cliente.
   a. Abra o terminal do ServidorDHCPv6;

   b. Crie dois arquivos com os seguintes nomes: dhcpd.conf e dhcp.leases. Para fazer isso digite os comandos:

```
# touch dhcpd.conf
# touch dhcpd.leases
```

O resultado deve ser:



c. Edite o arquivo dhcpd.conf. Ele deverá conter as linhas:

```
default-lease-time 600;
max-lease-time 7200;
subnet6 2001:db8:0:1::/64
{
      range6 2001:db8:0:1::129 2001:db8:0:1::254;
      option dhcp6.name-servers 2001:db8:0:1::10;
}
```

***Obs**: O campo name-servers contém o endereço da máquina que funcionará como servidor DNS. E, o campo range6 contém a faixa de endereços dentro do prefixo de sub-rede configurado que será distribuida entre os dispositivos clientes.
O resultado deve ser:

***Obs**: um editor de texto presente na máquina virtual que pode ser utilizado é o **nano**. Para usá-lo digite no terminal:

```
# nano dhcpd.conf
```

No nano, a sequência utilizada para salvar o arquivo é CTRL-O e para sair é CTRL-X.

d. Inicie o programa do dhcp através do comando:

```
# dhcpd -6 -cf dhcpd.conf -lf dhcpd.leases
```

O resultado deve ser:

7. Configure o dibbler-client no cliente para receber as configurações do servidor.
   e. Abra o terminal da máquina cliente.

   f. Dentro da pasta '*/etc/dibbler*', crie um arquivo chamado client.conf com o comando:

```
# touch /etc/dibbler/client.conf
```

   O resultado deve ser:

   g. Inclua nesse arquivo o seguinte texto:

```
iface eth0 {
      ia
      option dns-server
}
```

O resultado deve ser:

***Obs**: Para essa simulação, a pasta de configurações do *dibbler* foi virtualizada para a máquina cliente, porém, normalmente, esse programa é instalado com uma configuração padrão também localizada em *'/etc/dibbler/client.conf'*. A única ressalva necessária é que o modo stateless não pode ser acionado quando são feitas requisições de endereço para servidores DHCP. Logo, a seguinte linha do arquivo deve continuar comentada (adição do caracter #) ou, deve ser apagada:

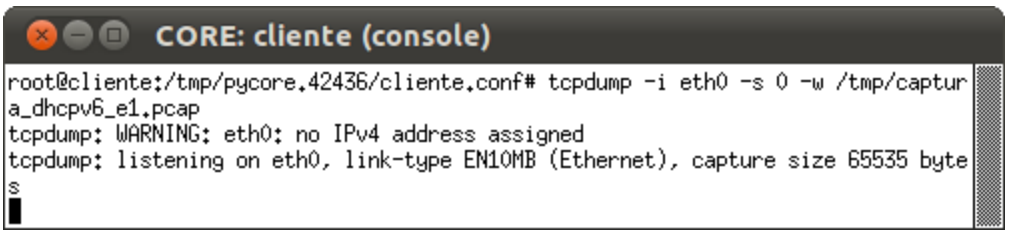
```
# stateless
```

8. Efetue a troca de mensagens DHCP no ServidorDHCP.
   a. Abra o terminal do cliente;

   b. Utilize o seguinte comando para iniciar a captura de pacotes em sua interface interface:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_dhcpv6_e1.pcap
```

O resultado deve ser:



***Obs**: Não feche esse terminal até o final do experimento, uma vez que, isso ocasionará no término da execução do comando "tcpdump" e prejudicará o andamento da experiência.

c. Abra outro terminal da máquina cliente;

d. Inicie o programa dibbler-client:

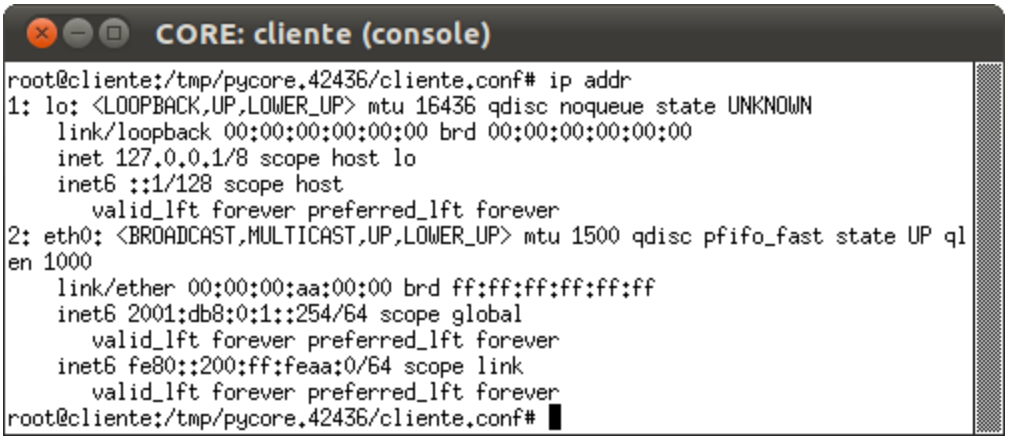
```
# dibbler-client start
```

O resultado deve ser:

e. Espere alguns segundos e utilize o seguinte comando para verificar que endereço IPv6 de escopo global foi adquirido:

```
# ip addr
```

O resultado deve ser:



***Obs**: Note o endereço ipv6 adquirido via DHCPv6.

f. Para visualizar o dns obtido via dhcp, digite o comando:

```
# cat /etc/resolv.conf
```

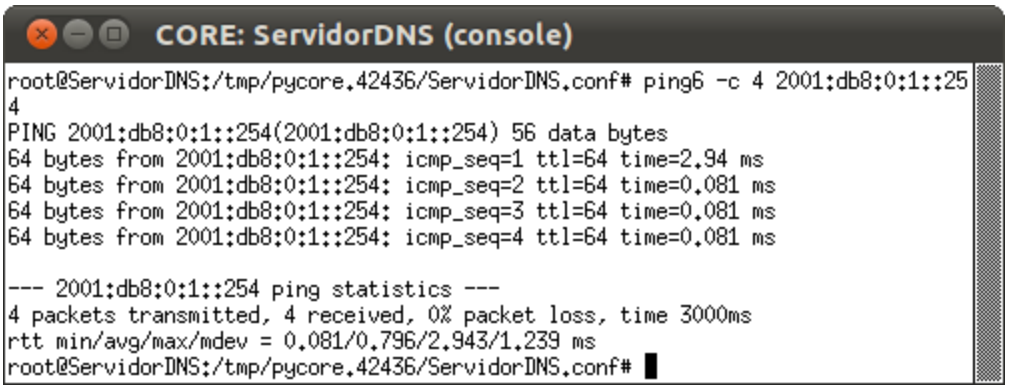
O resultado deve ser:

***Obs**: Observe o endereço IPv6 do DNS recebido via DHCPv6.

9. Teste a conectividade IPv6 entre os nós da rede:
   a. No terminal do ServidorDNS, utilize o seguinte comando :

```
# ping6 -c 4 2001:db8:0:1::254
```

O resultado deve ser:

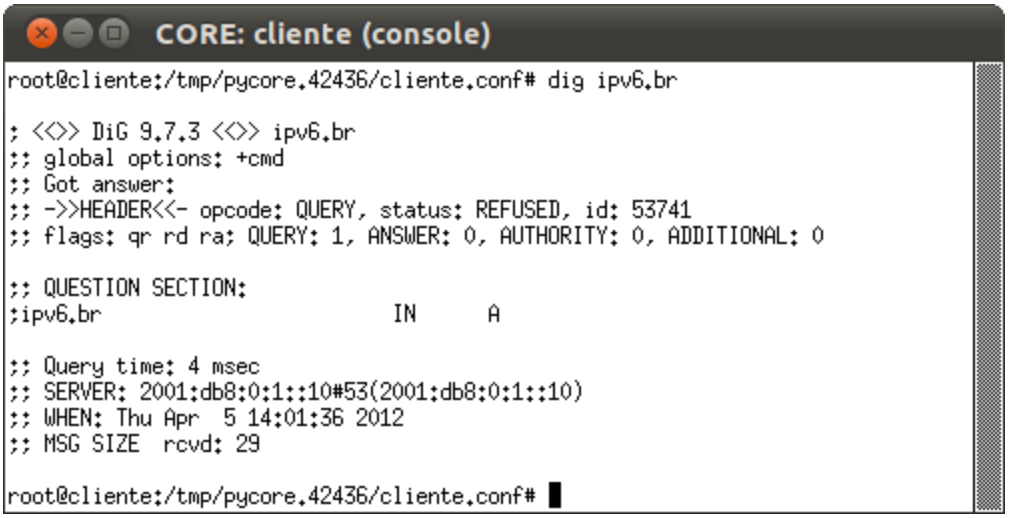


***Obs**: O endereço IPv6 deve ser o mesmo que o adquirido via DHCPv6, mostrado no passo 8.

   b. No terminal do cliente, teste o serviço DNS com o comando:

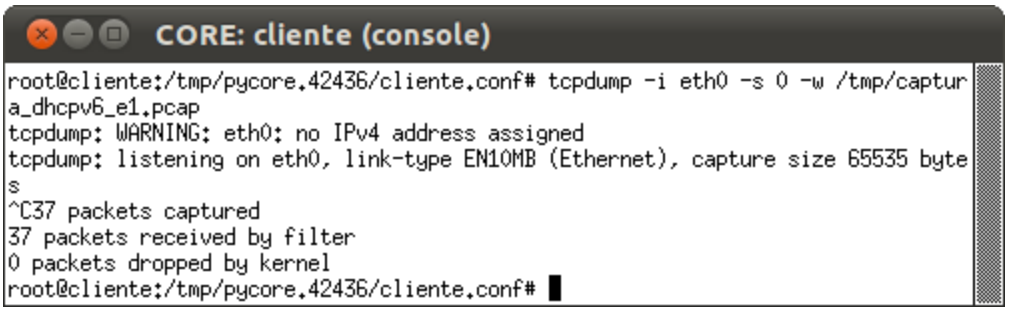
```
# dig ipv6.br
```

O resultado deve ser:



***Obs**: Observe que o servidor DNS respondeu a requisição do cliente.

c. No terminal do cliente, encerre a captura de pacotes através da sequência Ctrl+C.
O resultado deve ser:



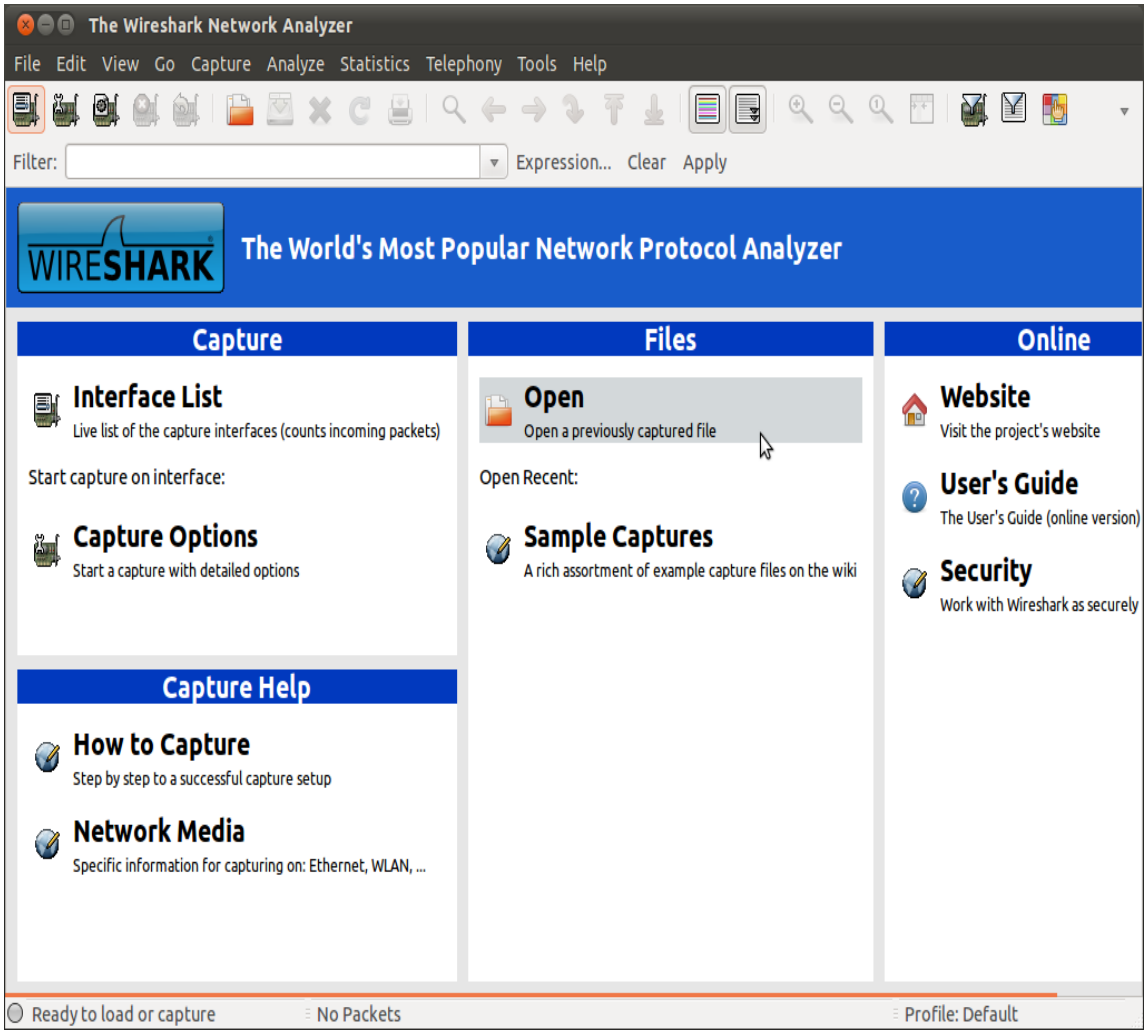
***Obs**: A quantidade de pacotes pode variar de acordo com o tempo esperado para dar o comando Ctrl+C.

10. Encerre a simulação com uma das seguintes ações:

a. aperte o botão  ;
b. utilize o menu Experiment > Stop.

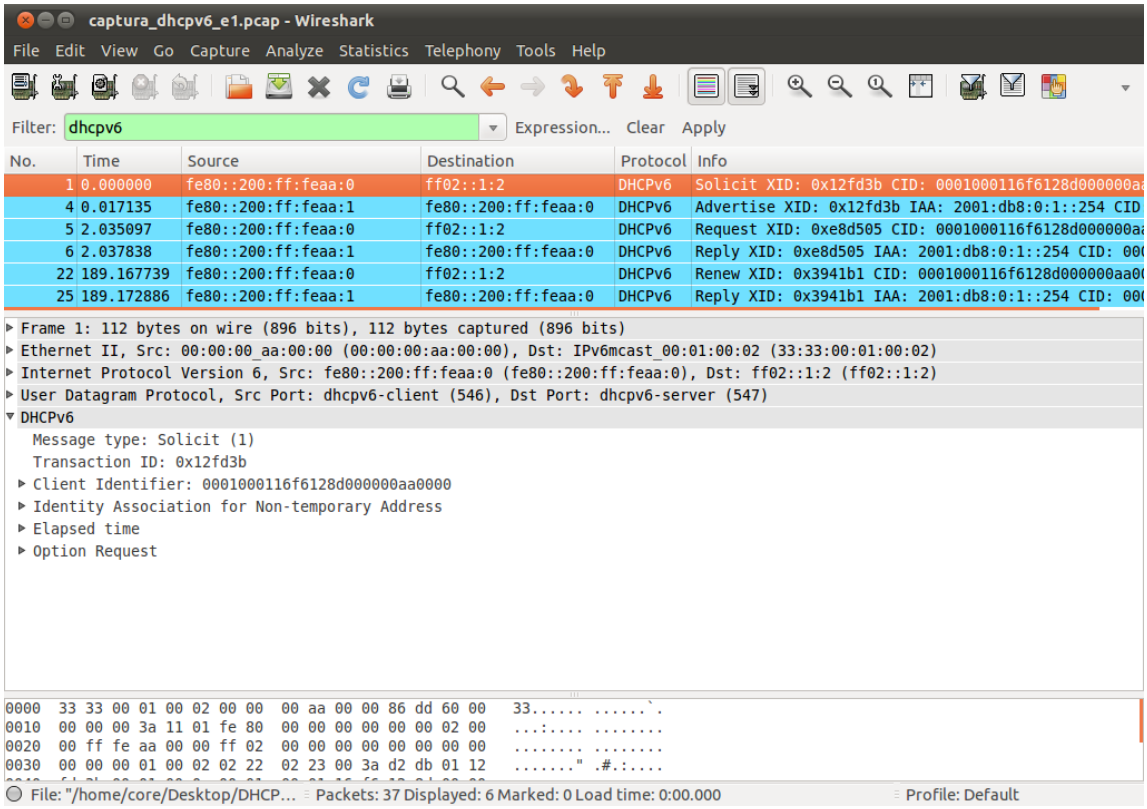
11. A verificação dos pacotes capturados será realizada através do programa Wireshark. Para iniciá-lo execute o seguinte comando em um terminal da máquina virtual:

```
$ wireshark
```



a. Abra o arquivo **/tmp/captura_dhcpv6_e1.pcap** com o menu File>Open;
b. Procure pelos pacotes *Solicit*, *Advertise*, *Request* e *Reply*. Analise-os e veja que os dados contidos nos pacotes conferem com o que foi passado na teoria.

Solicit:



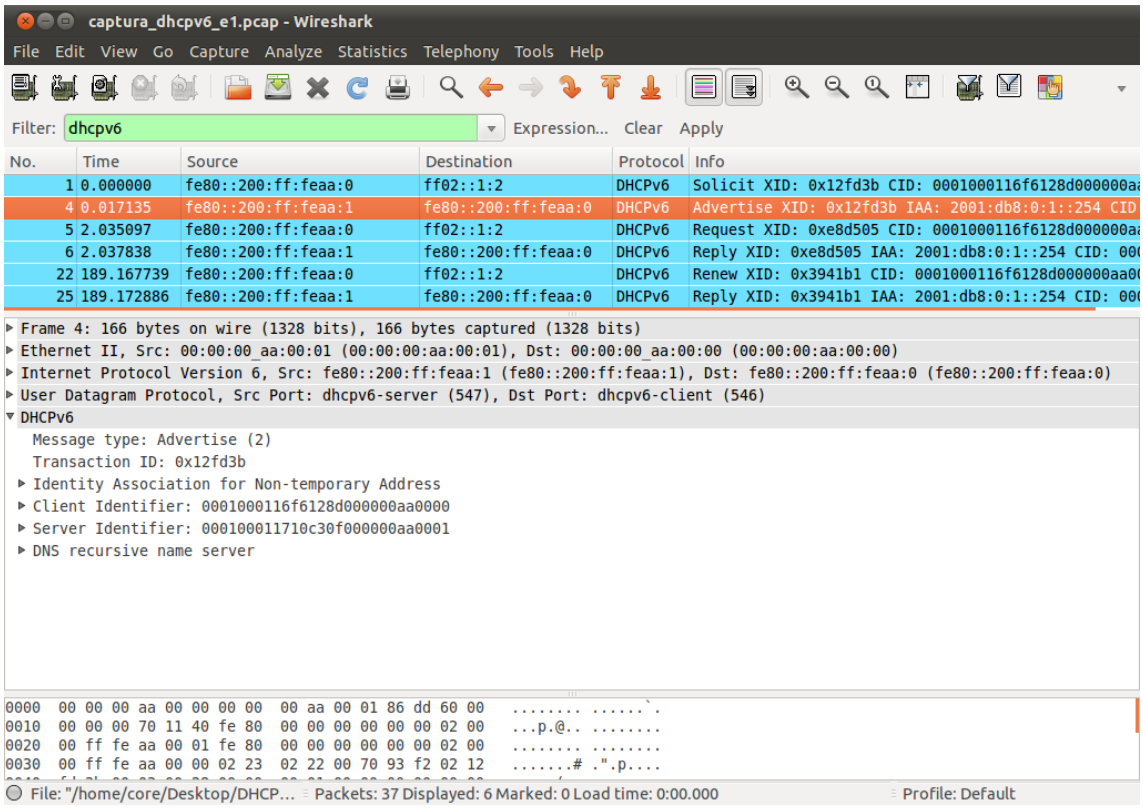
***Obs**: o filtro dhcpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço (33:33:00:01:00:02) sendo que o prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet e o sufixo 00:01:00:02 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a origem é o MAC address da interface da máquina que enviou a solicitação (00:00:00:aa:00:00).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho, no caso, o valor 0x11 refere-se à uma mensagem UDP.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface diretamente conectada ao enlace ao qual se fez a solicitação (fe80::200:ff:feaa:0).
- **Destination (IPv6)**: o destino é o endereço *Multicast Agent DHCP* (ff02::1:2).
- **Source port (UDP)**: indica a porta que se encontra o serviço dhcpv6-client cujo o valor é 546.
- **Destination port (UDP)**: indica a porta que se encontra o serviço dhcpv6-server no servidor. Seu valor é 547.
- **Message type (DHCPv6)**: indica através do valor 1 que o tipo da mensagem é Solicit;

- **Client Identifier (DHCPv6)**: contém dados da identificação única do cliente baseada no endereço fisico.
- **Identity Association for Non-temporary Address (DHCPv6)**: serve para requisitar o endereço IPv6 para o servidor.
- **Option Request (DHCPv6)**:
    - Requested Option Code: indica a informação que o dispositivo está solicitando ao servidor DHCP, no caso, *DNS Recursive Name Server* com o valor 23;

Advertise:



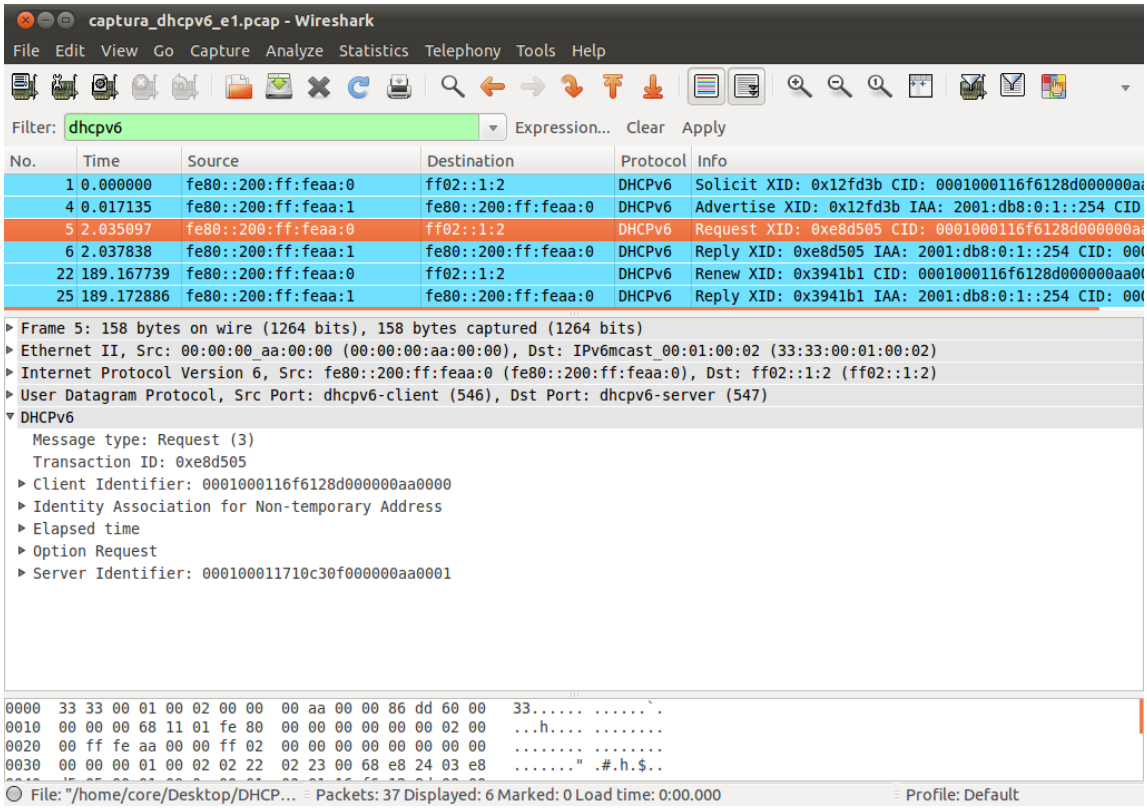
***Obs**: o filtro dhcpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço MAC da interface da máquina solicitante (00:00:00:aa:00:00).
- **Source (Ethernet)**: a origem é o MAC address da interface da máquina que enviou a resposta (00:00:00:aa:00:01).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho, no caso, o valor 0x11 refere-se à uma mensagem UDP.

- **Source (IPv6)**: a origem é o endereço IP de link local da interface do dispositivo que enviou a mensagem, ou seja, do Servidor DHCP6 (fe80::200:ff:feaa:1).
- **Destination (IPv6)**: o destino é o endereço unicast de link local da máquina solicitante (fe80::200:ff:feaa:0).
- **Source port (UDP)**: indica a porta que se encontra o serviço dhcpv6-server cujo o valor é 547.
- **Destination port (UDP)**: indica a porta que se encontra o serviço dhcpv6-client cujo o valor é 546.
- **Message type (DHCPv6)**: indica através do valor 2 que o tipo da mensagem é Advertise;
- **Identity Association for non-temporary address (DHCPv6)**: serve para carregar o endereço IPv6 para o cliente.
    - IA Address: contém o endereço e as caracteristicas que o cliente deve utilizar (2001:db8:0:1::254).
- **Client Identifier (DHCPv6)**: contém dados da identificação única do cliente baseada em seu endereço fisico.
- **Server Identifier (DHCPv6)**: contém dados da identificação única do servidor baseada em seu endereço fisico.
- **DNS recursive name server (DHCPv6)**:
    - DNS servers address: indica o endereço ipv6 do servidor DNS requisitado (2001:db8:0:1::10);

Request:



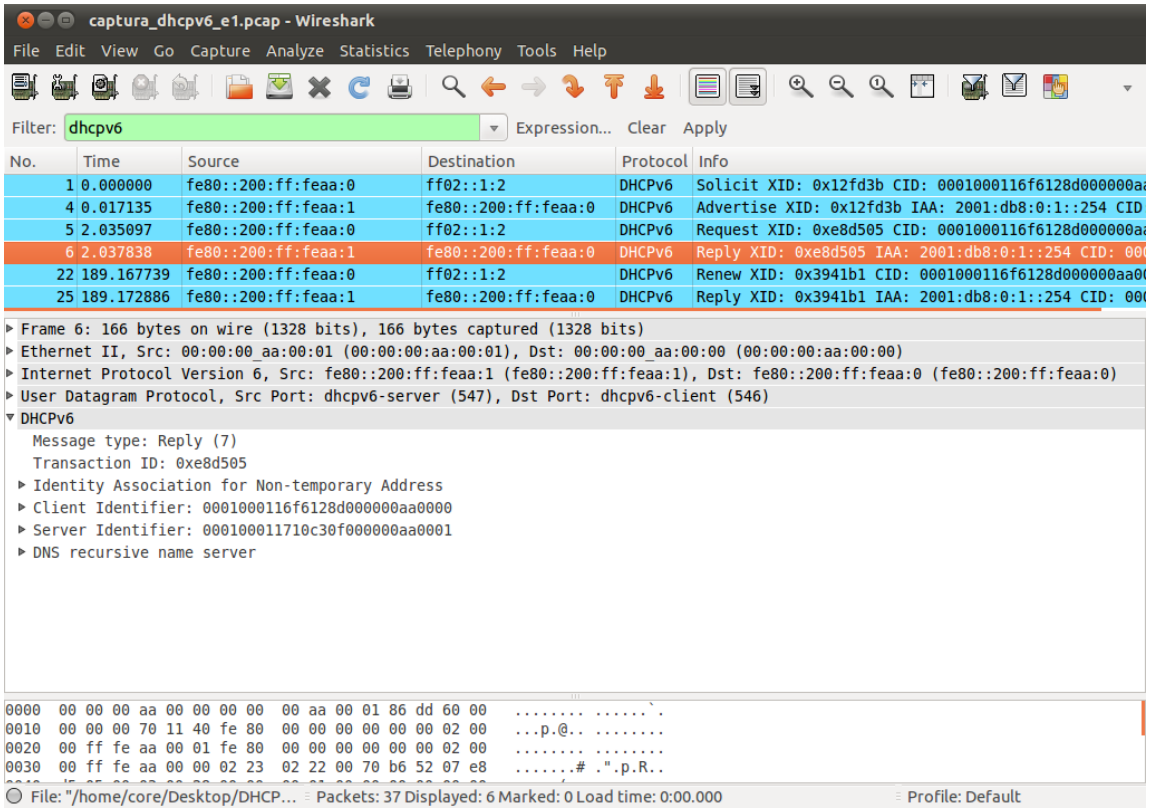
***Obs**: o filtro dhcpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço (33:33:00:01:00:02) sendo que o prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet e o sufixo 00:01:00:02 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a origem é o MAC address da interface da máquina cliente (00:00:00:aa:00:00).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho, no caso, o valor 0x11 refere-se à uma mensagem UDP.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface do dispositivo que enviou a mensagem, ou seja, do cliente (fe80::200:ff:feaa:0).
- **Destination (IPv6)**: o destino é o endereço *Multicast Agent DHCP* (ff02::1:2).
- **Source port (UDP)**: indica a porta que se encontra o serviço dhcpv6-client cujo o valor é 546.
- **Destination port (UDP)**: indica a porta que se encontra o serviço dhcpv6-server cujo o valor é 547.
- **Message type (DHCPv6)**: indica através do valor 3 que o tipo da mensagem é Request;

- **Client Identifier (DHCPv6)**: contém dados da identificação única do cliente baseada em seu endereço fisico.
- **Identity Association for non-temporary address (DHCPv6)**: serve para confirmar o endereço IPv6 recebido.
    - IA Address: contém o endereço e as caracteristicas que o cliente irá utilizar (2001:db8:0:1::254).
- **Option Request (DHCPv6)**:
    - Requested Option Code: indica quais informações estão sendo solicitadas ao servidor DHCP. No caso, o *DNS recursive name server* com o valor 23;
- **Server Identifier (DHCPv6)**: contém dados da identificação única do servidor baseada em seu endereço fisico.

Reply:



***Obs**: o filtro dhcpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço MAC da interface da máquina cliente (00:00:00:aa:00:00).
- **Source (Ethernet)**: a origem é o MAC address da máquina que está enviando a resposta (00:00:00:aa:00:01).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).

- **Next Header (IPv6)**: indica qual é o próximo cabeçalho, no caso, o valor 0x11 refere-se à uma mensagem UDP.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface do dispositivo que enviou a mensagem, ou seja, do Servidor DHCPv6 (fe80::200:ff:feaa:1).
- **Destination (IPv6)**: o destino é o endereço IPv6 unicast de link local do cliente (fe80::200:ff:feaa:0).
- **Source port (UDP)**: indica a porta que se encontra o serviço dhcpv6-server cujo o valor é 547.
- **Destination port (UDP)**: indica a porta que se encontra o serviço dhcpv6-client cujo o valor é 546.
- **Message type (DHCPv6)**: indica através do valor 7 que o tipo da mensagem é Reply;
- **Identity Association for non-temporary address (DHCPv6)**: serve para confirmar o endereço IPv6 fornecido.
    - IA Address: contém o endereço e as caracteristicas que o cliente irá utilizar (2001:db8:0:1::254).
- **Client Identifier (DHCPv6)**: contém dados da identificação única do cliente baseada em seu endereço fisico.
- **Server Identifier (DHCPv6)**: contém dados da identificação única do servidor baseada em seu endereço fisico.
- **DNS recursive name server (DHCPv6)**:
    - DNS servers address: indica o endereço IPv6 do servidor DNS requisitado (2001:db8:0:1::10);

IPV6 - DHCPv6

Experiência2 - DHCPv6 stateless - Information-Request e Reply

Objetivo

Esta experiência possui como objetivo apresentar o funcionamento do DHCPv6 em modo stateless, ou seja, o modo em que o servidor DHCPv6 envia apenas dados de configurações opcionais, como as referentes ao DNS, sendo o endereço IPv6 fornecido via *Router Advertisement*.

Para a realização do presente exercicio será utilizada a topologia descrita no arquivo: **DHCPv6-E2.imn**.

Introdução Teórica

O DHCPv6 possui dois modos de operação, um *stateless* em que o servidor DHCP fornece apenas informações de DNS (*Domain Name Server*) para seus clientes e, outro, *stateful* no qual fica encarregado também da distribuição de endereços IPv6 na rede. Este segundo modo foi abordado na primeira experiência sobre DHCPv6. A presente experiência foca unicamente no modo de configuração *stateless*.

Duas etapas são necessárias para a realização do experimento, sendo a primeira constituida pela configuração do roteador da rede, uma vez que este influencia diretamente o comportamento do sevidor DHCP.
Tal configuração tem como intuito fazer com que as mensagem *Router Advertisement*, que são enviadas pelo roteador, tanto periodicamente como em resposta à requisições do tipo *Router Solicitation*, contenham informações sobre o prefixo de rede, bem como, sobre a necessidade da captura de outras características da rede com servidor DHCP.

Já a segunda, consiste na alterção do comportamento do servidor DHCP com o acionamento de duas opções: *AdvManagedFlag*, que configura se o cliente possui ou não permissão para receber endereço IPv6 do servidor DHCPv6; e, o AdvOtherConfigFlag, que configura se o cliente pode receber outras configurações a partir do servidor DHCPv6.

Com a conclusão dessas etapas, será observada a tranferência das mensagens entre o servidor DHCPv6 e uma máquina cliente que precise configurar sua interface de rede. A comunicação é iniciada com o envio, pelo cliente, da mensagem *Information-Request,* que

utiliza o endereço de multicast de link local, para requisitar a qualquer servidor DHCP as caracteristicas da rede. Em resposta, os servidores enviam diretamente ao cliente uma mensagem *Reply* com tais características.

Roteiro Experimental

1. Caso não esteja utilizando a máquina virtual fornecida pelo NIC.br é preciso, antes de começar a experiência, instalar alguns softwares para auxiliar no aprendizado (caso contrário vá para o passo 2). Siga o passo a seguir para realizar as instalações:
    a. Para fazer algumas verificações durante o experimento será necessário a utilização do programa **Wireshark** que melhora a visualização de pacotes que transmitidos na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

b. O **dhcpd** que é o serviço reponsável pelas tarefas de servidor DHCP. Para instalá-lo, baixe a última versão do pacote ‘*tar.gz*’ no site http://www.isc.org/software/dhcp (acessado em 10/04/2012) e utilize os seguintes comandos em um Terminal:

```
$ cd <pasta onde o arquivo foi baixado>
$ tar xf dhcp-4.2.3-P2.tar.gz
$ cd dhcp-4.2.3-P2/
$ ./configure
$ make
$ sudo make install
```

***Obs**: Lembre-se de utilizar os numeros corretos de versão para extrair o pacote e acessar a pasta com os arquivos de instalação.
Depois do comando ‘`sudo`‘ será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

c. O **dibbler-client**, que realiza as funções de cliente DHCP. Para instalá-lo, baixe a última versão (Acima da 0.8.2) do código fonte no site

http://klub.com.pl/dhcpv6/ (acessado em 10/04/2012) e utilize os seguintes comandos em um Terminal:

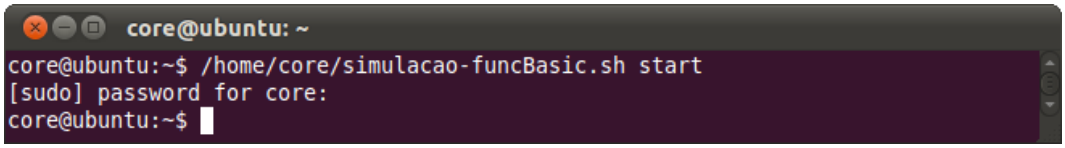
```
$ cd <pasta onde o arquivo foi baixado>
c.    $ tar xf dibbler-0.8.2.tar.gz
d.    $ cd dibbler-0.8.2/
e.    $ ./configure
f.    $ make
g.    $ sudo make install
```

***Obs**: Lembre-se de utilizar os números corretos de versão para extrair o pacote e acessar a pasta com os arquivos de instalação. Depois do comando '`sudo`' será solicitada a senha do usuário core. Digite "core" para prosseguir com a instalação.

2. Agora, utilize o seguinte comando para iniciar a configuração da simulação:

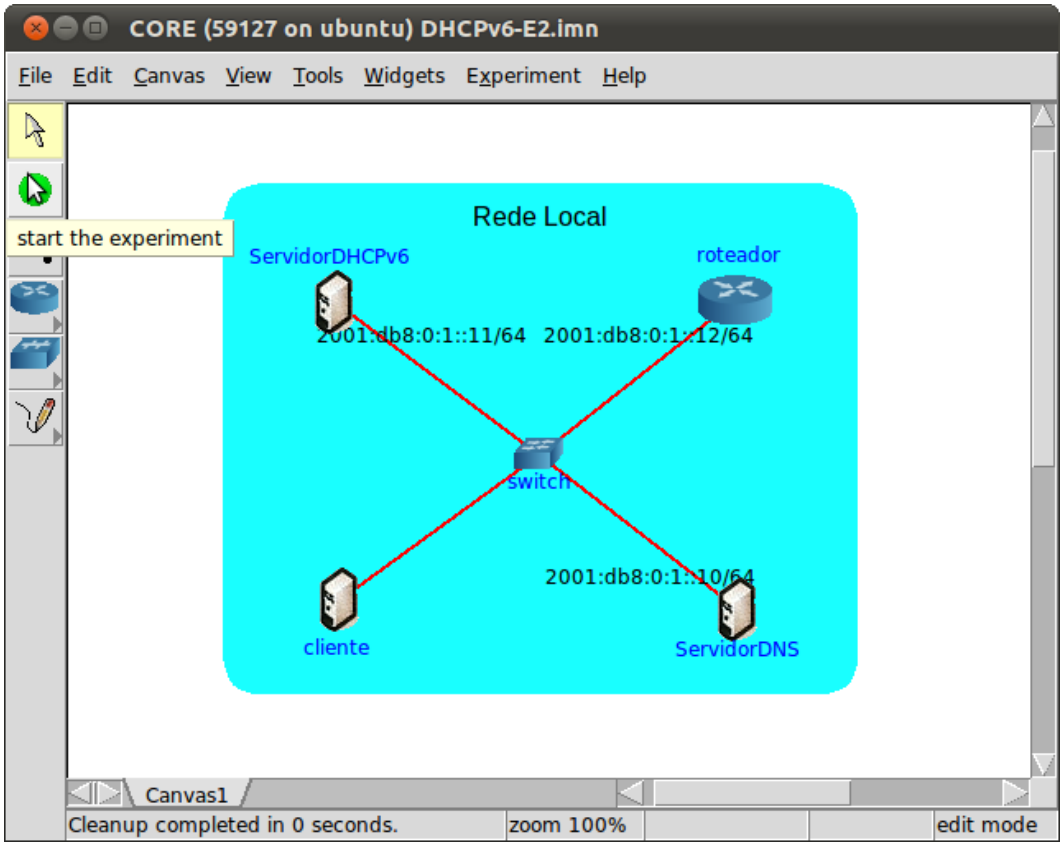
```
$ /home/core/simulacao-funcBasic.sh start
```

Caso necessário, digite "core" para prosseguir com a configuração.
O resultado deve ser:



3. Inicie o CORE e abra o arquivo "**DHCPv6-E2.imn**" localizado no diretório do desktop "Funcionalidades/DHCPv6", da máquina virtual do NIC.br. A seguinte topologia deve

aparecer:



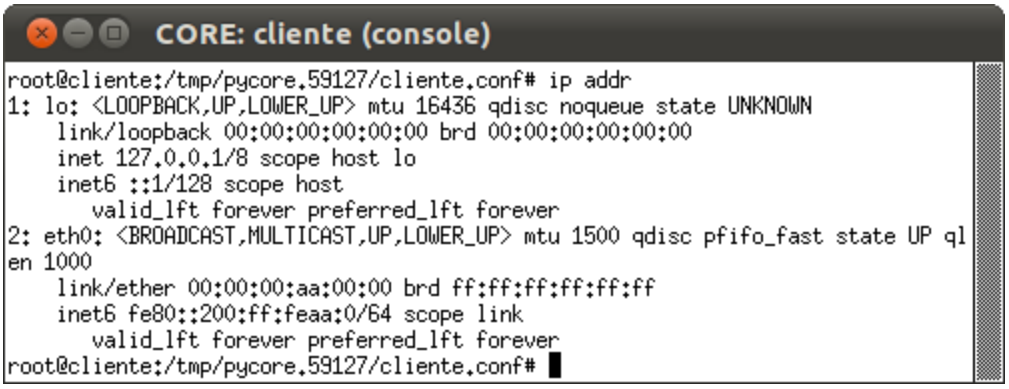
5. Verifique a configuração dos nós da topologia:
   a. Inicie a simulação realizando um dos seguintes passos:
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do ‘cliente’ com um duplo-clique.

   c. Observe a configuração do ‘cliente’ com o seguinte comando:

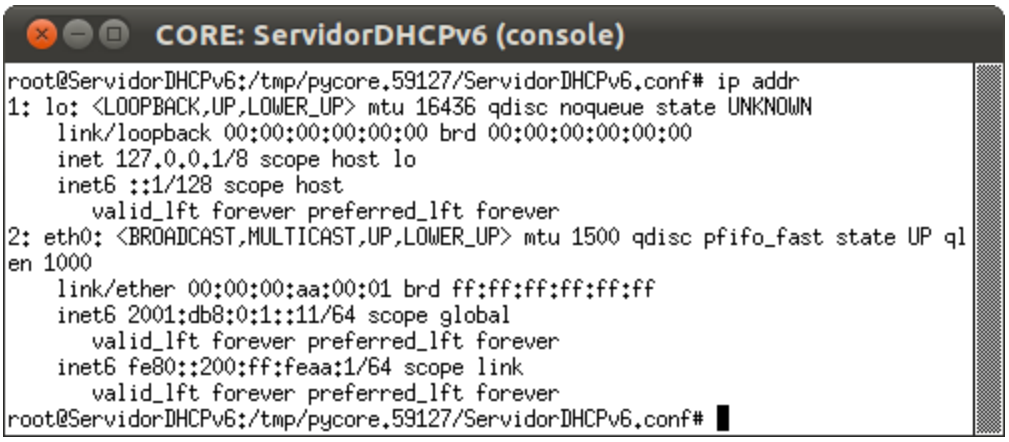
```
# ip addr
```

O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces.

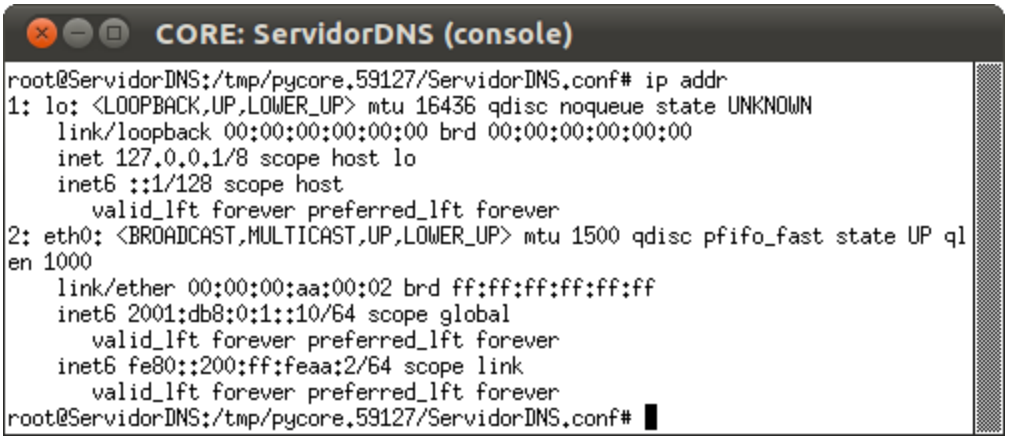
d. Observe a configuração do 'ServidorDHCPv6' utilizando o mesmo comando. O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces.

e. Verifique a configuração do 'ServidorDNS' utilizando o comando.

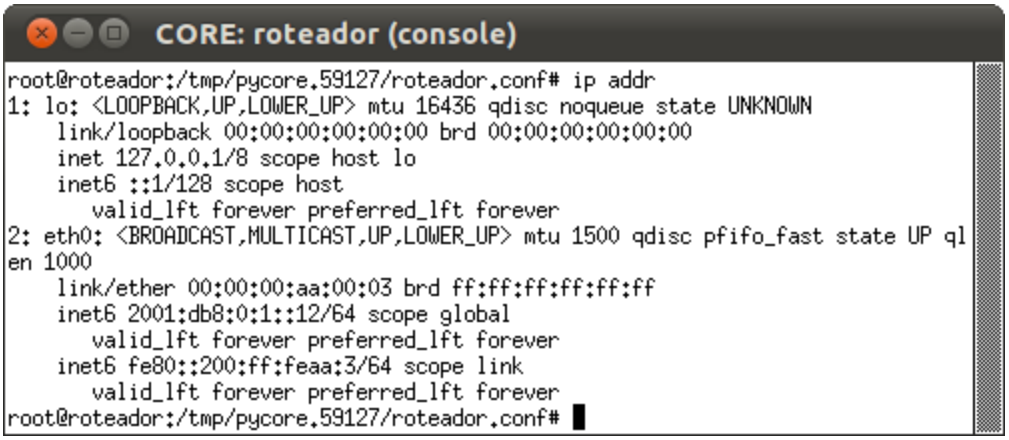
O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces.

f. Por fim, utilize o mesmo comando para verificar a configuração de rede do 'roteador'.

O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces.

6. Inicie o serviço DNS no 'ServidorDNS':
   a. Abra o terminal do 'ServidorDNS' com um duplo-clique.

   b. Utilize o seguinte comando para iniciar o serviço DNS:

```
# dnsmasq -i eth0
```

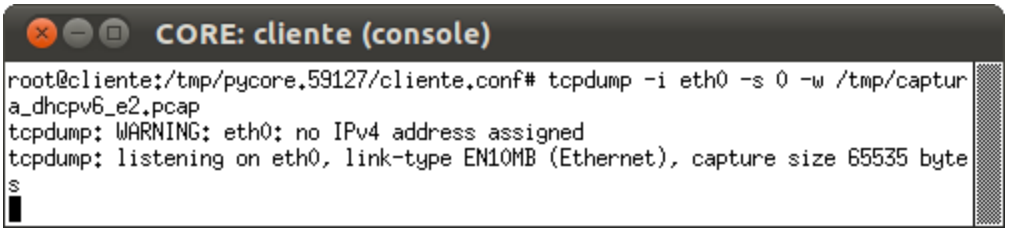
O resultado deve ser:

7. Edite as configurações do Quagga no 'roteador', para que ele envie a mensagem Router Advertisement contendo as características da rede para o 'cliente':
   a. Abra o terminal do 'cliente' com um duplo-clique.

   b. Utilize o seguinte comando para iniciar a captura de pacotes:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_dhcpv6_e2.pcap
```

   O resultado deve ser:



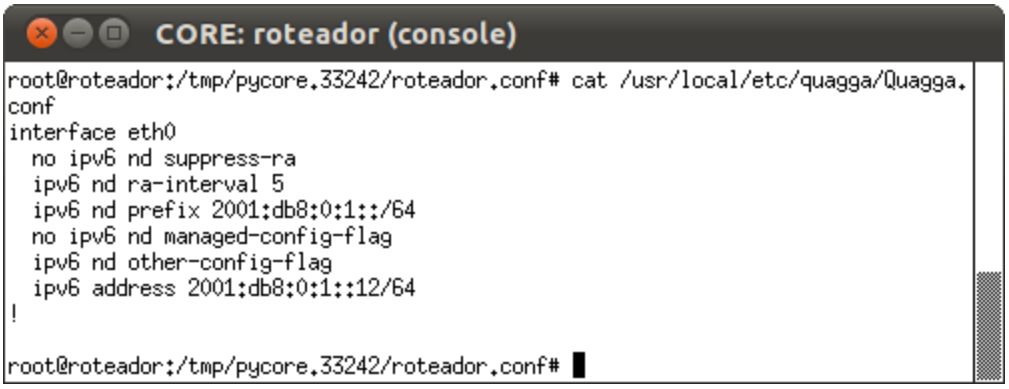
   ***Obs**: Não feche esse terminal até o final do experimento, uma vez que, isso ocasionará no término da execução do comando "tcpdump" e prejudicará o andamento da experiência.

   c. Abra o terminal do 'roteador' com um duplo-clique.

   d. Substitua pelo seguinte o conteúdo do arquivo Quagga.conf que está localizado dentro da pasta /usr/local/etc/quagga:

```
interface eth0
  no ipv6 nd suppress-ra
  ipv6 nd ra-interval 5
  ipv6 nd prefix 2001:db8:0:1::/64
  no ipv6 nd managed-config-flag
  ipv6 nd other-config-flag
  ipv6 address 2001:db8:0:1::12/64
!
```

O resultado deve ser:



***Obs**: um editor de texto presente na máquina virtual que pode ser utilizado é o **nano**. Para usá-lo digite no terminal:

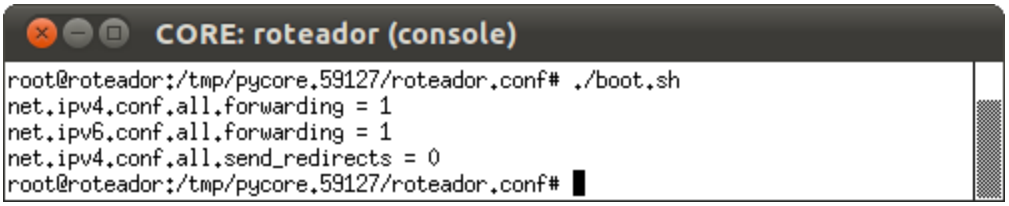
```
# nano /usr/local/etc/quagga/Quagga.conf
```

No nano, a sequência utilizada para salvar o arquivo é CTRL-O e para sair é CTRL-X.

e. Em seguida, acione o script que re-inicia as configurações do quagga. Ele que se encontra na pasta base do roteador:

```
# ./boot.sh
```

O resultado deve ser:

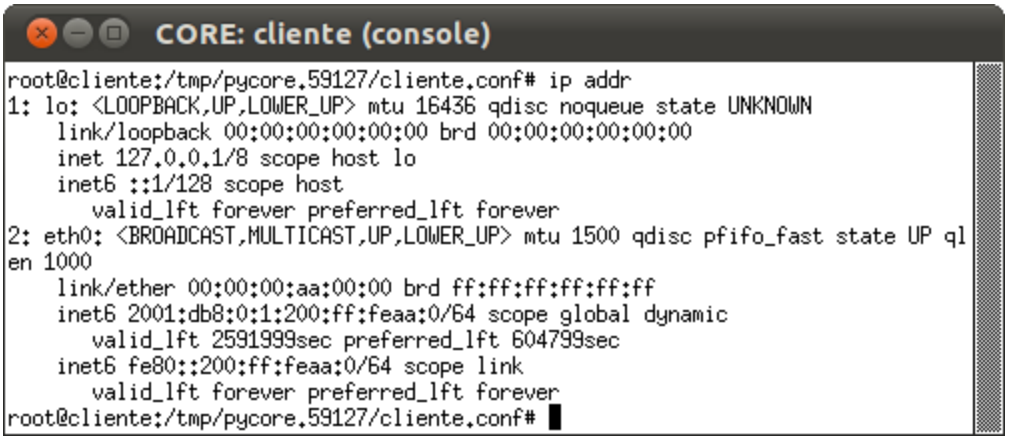


8. Verifique o endereço IPv6 adquirido e teste a conectividade.
   a. Abra outro terminal do 'cliente' com um duplo-clique.

   b. Espere alguns segundos e verifique o endereço ipv6 adquirido, com o comando:

```
# ip addr
```

O resultado deve ser:

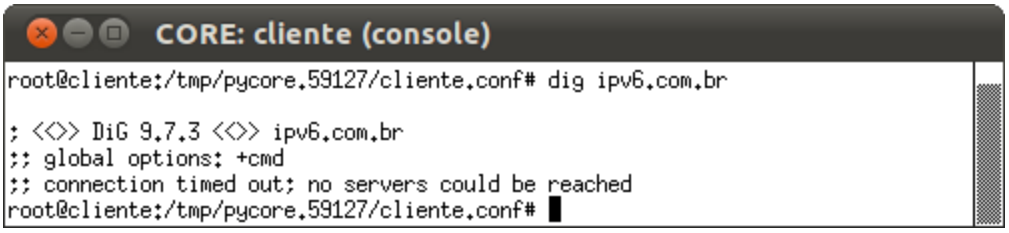


***Obs**: Note o endereço IPv6 adquirido via Router Advertisement.

c. Teste o serviço DNS através do comando:

```
# dig ipv6.br
```

O resultado deve ser:



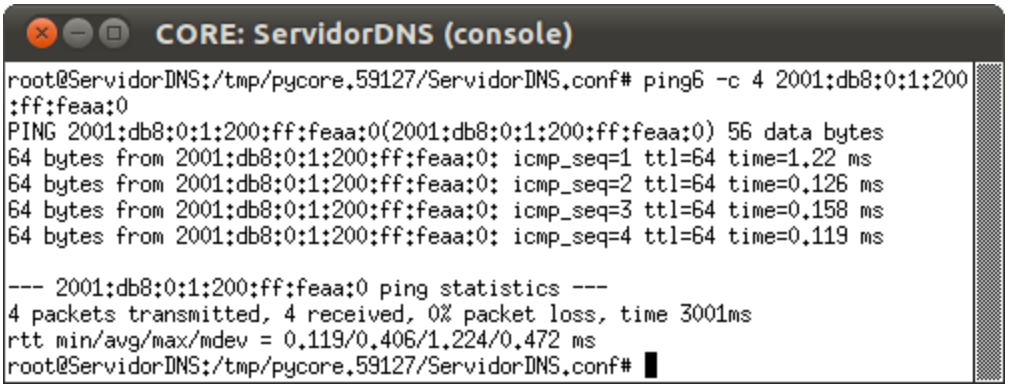
***Obs**: Note que não foi adquirido endereço do servidor DNS.

d. Abra o terminal do 'ServidorDNS' com um duplo-clique.

e. O utilize o seguinte comando para testar a conectividade IPv6:

```
# ping6 -c 4 2001:db8:0:1:200:ff:feaa:0
```

O resultado deve ser:



***Obs**: Utilize o endereço IPv6 adquirido via DHCP no passo 8 b.

9. Configure o dhcpd no 'ServidorDHCPv6' para enviar as informações aos clientes:
   a. Abra o terminal do 'ServidorDHCPv6' com um duplo-clique.

   b. Crie os seguintes arquivos:

```
# touch dhcpd.conf
# touch dhcpd.leases
```

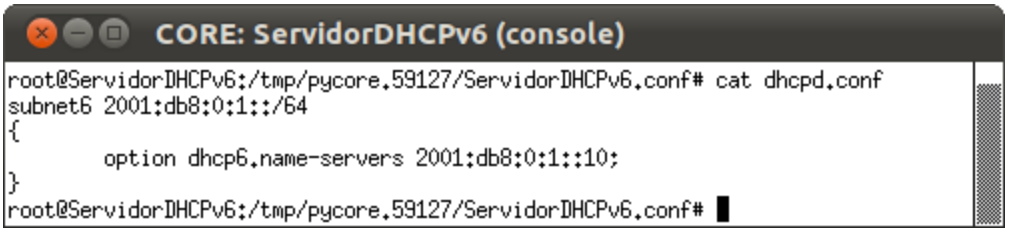
   O resultado deve ser:

   c. Adicione o seguinte conteúdo no arquivo dhcpd.conf:

```
subnet6 2001:db8:0:1::/64
{
      option dhcp6.name-servers 2001:db8:0:1::10;
}
```

   ***Obs**: o name-server é o endereço da máquina que funcionará como DNS server.

   O resultado deve ser:



d. Inicie o serviço dhcpd com o comando:

```
# dhcpd -6 -cf dhcpd.conf -lf dhcpd.leases
```

O resultado deve ser:

10. Configure o dibbler-client no ‘cliente’ para receber as informações do ‘ServidorDHCPv6’.
    a. Abra o terminal do ‘cliente’ com um duplo-clique.

    b. Crie um arquivo chamado client.conf, dentro da pasta /etc/dibbler com o comando:

```
# touch /etc/dibbler/client.conf
```

O resultado deve ser:

    c. Adicione as seguintes configurações nesse arquivo:

```
iface eth0 {
      stateless
      option dns-server
}
```

O resultado deve ser:

***Obs**: Na experiência a pasta /etc/dibbler/ foi mapeada pela plataforma CORE, porém na máquina real essa pasta contém um arquivo com uma configuração padrão do dibbler-client. A única ressalva que precisa ser feita é a de que não se pode acionar o modo stateless quando se faz requisição de endereço para servidor DHCP. Assim a seguinte linha do arquivo precisa estar comentada (adição do caracter #):

```
# ia
```

11. Observe a troca de mensagens DHCP entre a máquina 'ServidorDHCP' e a 'cliente'.
    h. Abra outro terminal do 'cliente' com um duplo-clique.

    i. Inicie o programa dibbler-client:

```
# dibbler-client start
```

O resultado deve ser:

12. Verifique o dns adquirido e teste a conectividade.

a. Para visualizar o dns obtido via dhcp, digite o comando:

```
# cat /etc/resolv.conf
```

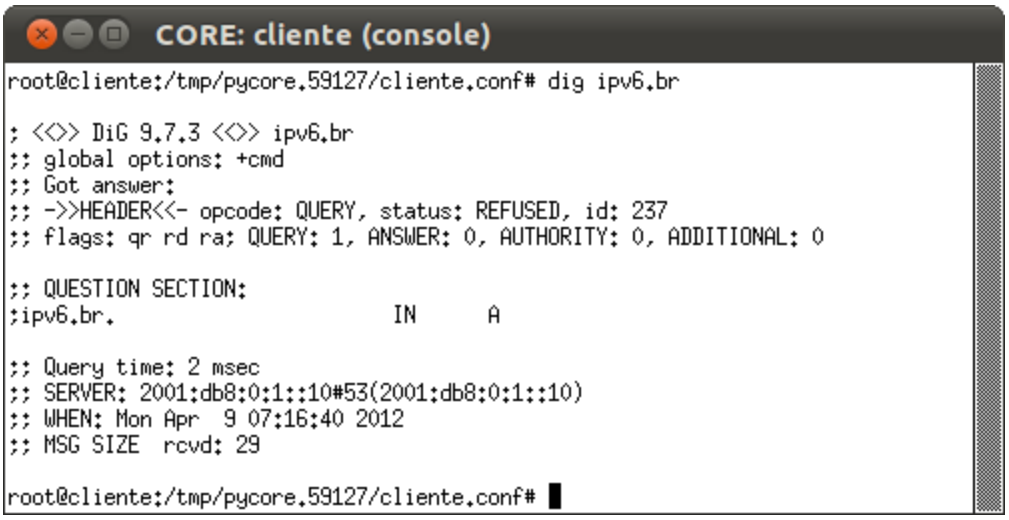
O resultado deve ser:

***Obs**: Observe o endereço do servidor DNS adquirido via DHCPv6.

b. Teste o serviço DNS através do comando:

```
# dig ipv6.br
```

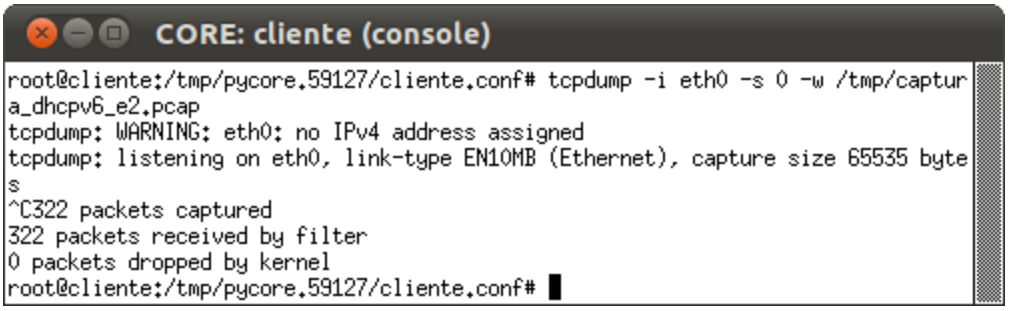
O resultado deve ser:



***Obs**: Note que o servidor de DNS agora responde a requisição.

c. No terminal do ‘cliente’, encerre a captura de pacotes digitando a sequência Ctrl+C.

O resultado deve ser:



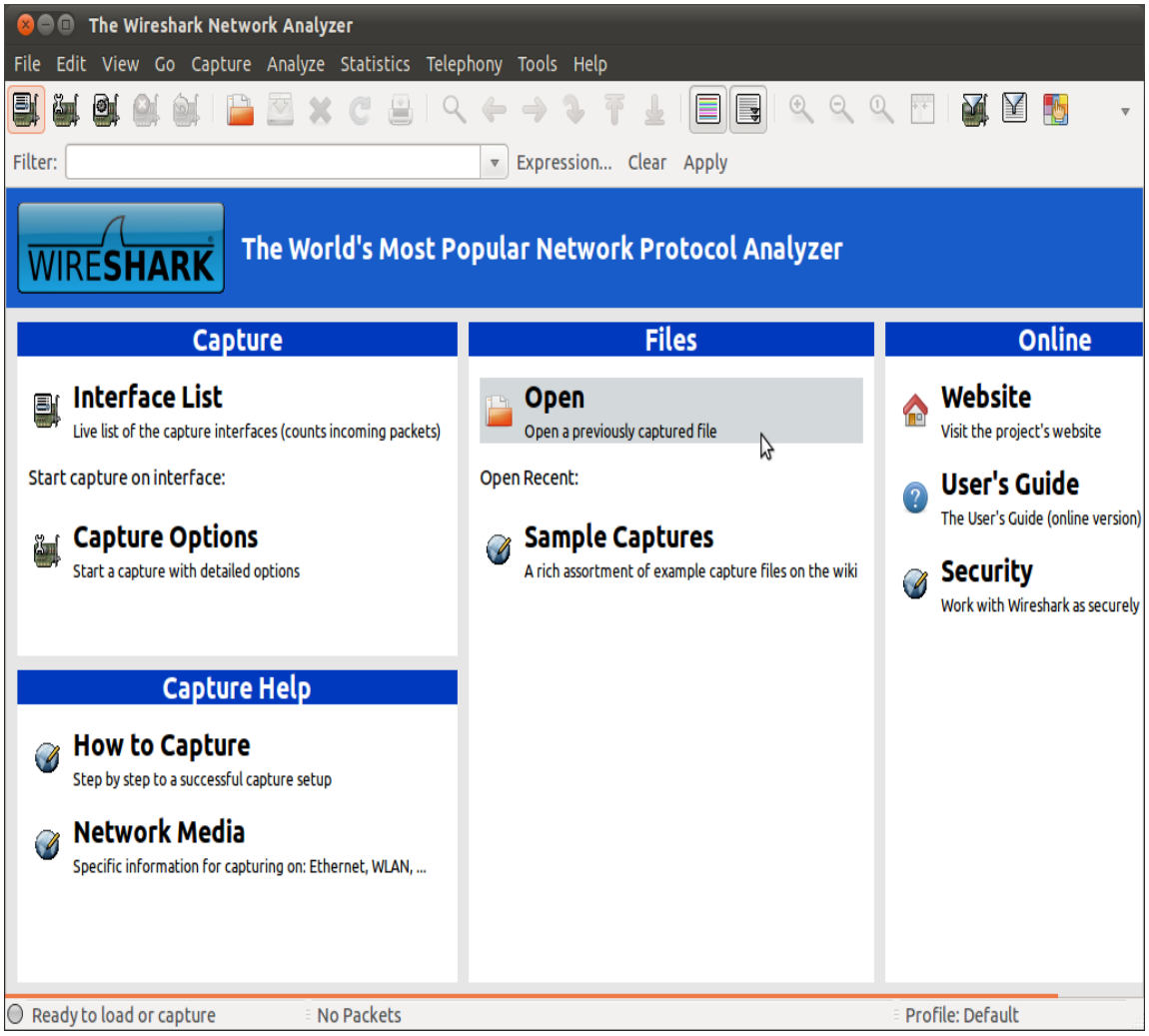
***Obs**: A quantidade de pacotes pode variar de acordo com o tempo esperado para dar o comando Ctrl+C.

13. Encerre a simulação com uma das seguintes ações:

    a. aperte o botão  ; ou
    b. utilize o menu Experiment > Stop.

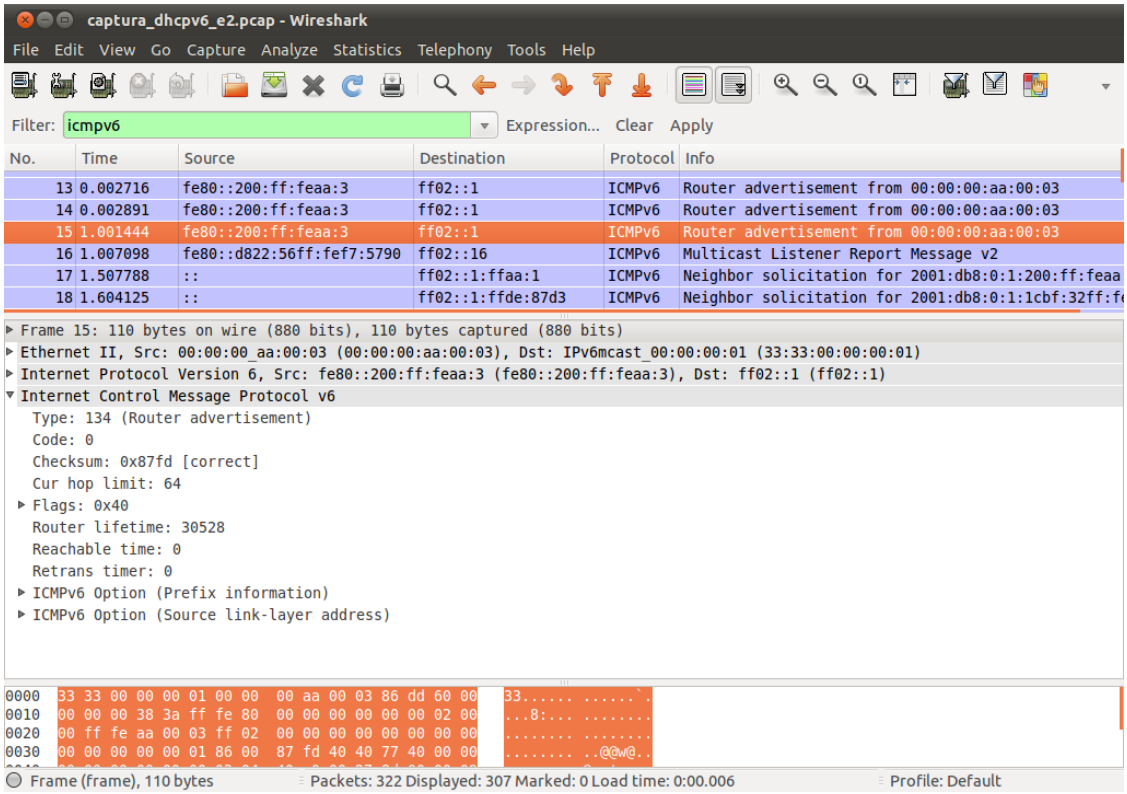
14. A verificação dos pacotes capturados será realizada através do programa Wireshark. Para iniciá-lo execute o seguinte comando em um terminal da máquina virtual:

```
$ wireshark
```



a. Abra o arquivo **/tmp/captura_dhcpv6_e2.pcap** com o menu File>Open;
b. Procure por um pacote '*Router Advertisement*' que contenha a opção ICMPv6 *Prefix Information*. Analise-o e veja que seus dados conferem com o que foi apresentado na teoria.

Router Advertisement:



***Obs**: o filtro icmpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço MAC (33:33:00:aa:00:01) sendo que o prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet e, o sufixo 00:aa:00:01 indica os últimos 32 bits do endereço multicast IPv6 dessa mensagem.
- **Source (Ethernet)**: a origem é o endereço MAC do roteador que enviou a mensagem (00:00:00:aa:00:03).
- **Type (Ethernet)**: indica que a mensagem utiliza IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho (de extensão do IPv6), no caso, o valor 58(0x3a) refere-se à uma mensagem ICMPv6.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface que originou a mensagem, neste caso, do roteador (fe80::200:ff:feaa:3).
- **Destination (IPv6)**: o destino é o endereço *Multicast All nodes* (ff02::1).
- **Type (ICMPv6)**: indica que a mensagem é do tipo 134 (*Router Advertisement*).
- **ICMPv6 Option (ICMPv6)**: indica as opções do pacote ICMPv6:
    - Prefix Information
        - *Type*: indica o tipo de dado da mensagem ICMPv6. No nosso caso, ela é do tipo "*Prefix information*";
        - *Autonomous Address-Configuration Flag (A)*: indica se o prefixo deve ser

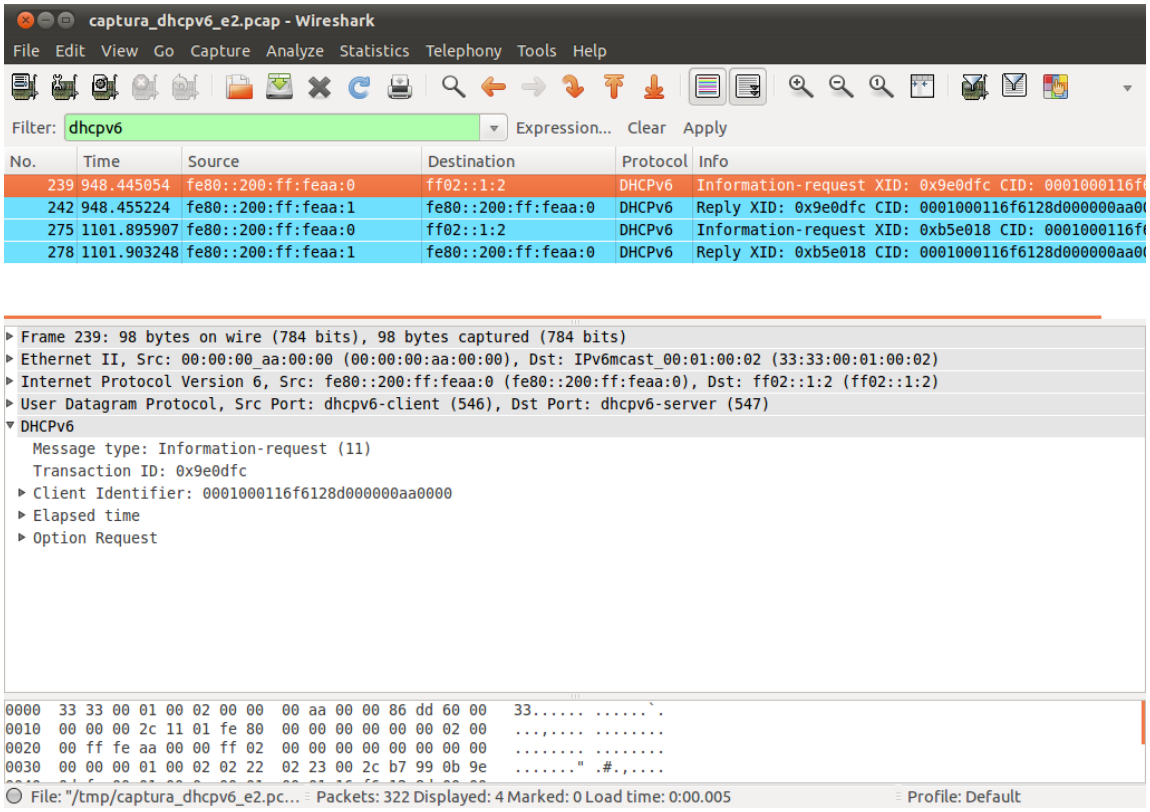
utilizado para autoconfiguração stateless (1) .

- *Preferred Lifetime*: marca o tempo, em segundos, que o endereço é preferencial, ou seja, um endereço que pode ser utilizado indistintamente. O valor (0xffffffff) indica infinito.
- *Valid Lifetime*: marca o tempo, em segundos, de expiração do endereço gerado. O valor (0xffffffff) indica infinito.
- *Prefix*: contém o prefixo de rede a ser utilizado (`2001:db8:0:1::`).
- *Prefix length*: contém o tamanho do prefixo da rede (64).

- Source Link Layer Address
  - *Type*: indica o tipo de dado da mensagem ICMPv6. No nosso caso, ela é do tipo "*Source link-layer address*";
  - *Link-layer address*: indica o endereço MAC da interface do roteador.

c. Procure por pacotes do tipo *Information-Request* e *Reply*. Analise-os e veja que seus dados conferem com o que foi apresentado na teoria.

Information-Request:

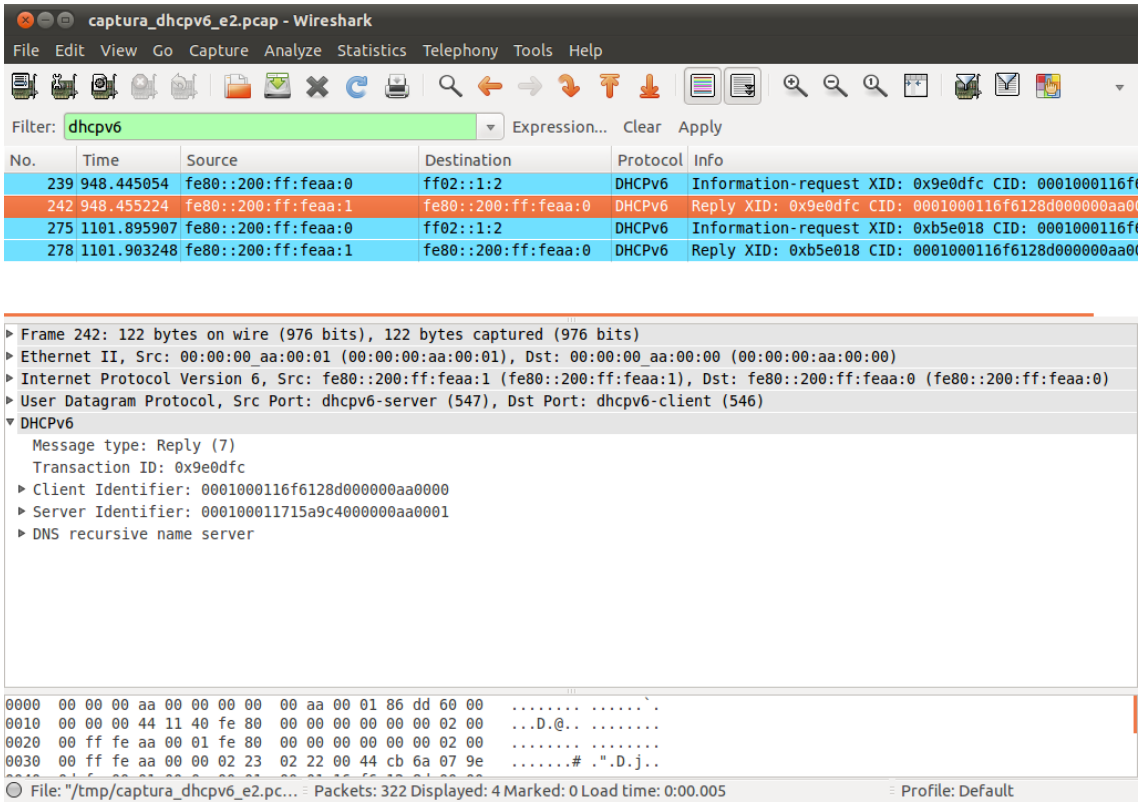


***Obs**: o filtro dhcpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço MAC (33:33:00:01:00:02) sendo que o prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet e, que o sufixo 00:01:00:02 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a origem é o endereço MAC da interface de rede do cliente (00:00:00:aa:00:00).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho, no caso, o valor 0x11 refere-se à uma mensagem UDP.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface de rede requisitante (fe80::200:ff:feaa:0).
- **Destination (IPv6)**: o destino é o endereço *Multicast Agent DHCP* (ff02::1:2).
- **Source port (UDP)**: indica a porta em que se encontra o serviço dhcpv6-client (546).
- **Destination port (UDP)**: indica a porta em que se encontra o serviço dhcpv6-server (547).
- **Message type (DHCPv6)**: indica através do valor 11 que o tipo da mensagem é *Information Request*;
- **Client Identifier (DHCPv6)**: contém a identificação única do cliente baseada em seu endereço fisico.
- **Option Request (DHCPv6)**
    - Requested Option Code: indica a informação que o dispositivo está solicitando ao servidor DHCP, no caso, *DNS recursive name server* (23);

Reply:



Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço MAC da interface da máquina cliente (00:00:00:aa:00:00).
- **Source (Ethernet)**: a origem é o endereço MAC da interface do servidor DHCPv6 que enviou a resposta (00:00:00:aa:00:01).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho, no caso, o valor 0x11 refere-se à uma mensagem UDP.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface do servidor que originou a mensagem, neste caso, o servidor DHCPv6 (fe80::200:ff:feaa:1).
- **Destination (IPv6)**: o destino é o endereço unicast de link local do cliente requisitante (fe80::200:ff:feaa:0).
- **Source port (UDP)**: indica a porta em que se encontra o serviço dhcpv6-server (547).
- **Destination port (UDP)**: indica a porta que se encontra o serviço dhcpv6-client (546).
- **Message type (DHCPv6)**: indica através do valor 7 que o tipo da mensagem é *Reply*;
- **Client Identifier(DHCPv6)**: contém dados da identificação única do cliente baseada em seu endereço fisico.

- **Server Identifier(DHCPv6)**: contém dados da identificação única do servidor baseada em seu endereço fisico.
- **DNS recursive name server(DHCPv6)**
    - DNS servers address: indica o endereço IPv6 do servidor DNS requisitado (2001:db8:0:1::10).

IPV6 - DHCPv6

Experiência3 - DHCPv6 Prefix Delegation

Objetivo

Esta experiência tem como objetivo apresentar o funcionamento do serviço de delegação de prefixos do DHCPv6 a roteadores. Estes, então, se encarregaram de administrar o prefixo recebido para assim realizar o serviço de Autoconfiguração Stateless(via Router Advertisement) de seus clientes.

Para a realização do presente exercicio será utilizada a topologia descrita no arquivo: **DHCPv6-E3.imn**.

Introdução Teórica

O Dynamic Host Configuration Protocol (DHCP) é um protocolo de autoconfiguração stateful, utilizado para distribuir endereços IP e informações de rede dinamicamente. Contudo, suas implementações IPv6 possuem significativas diferenças e particularidades com relação ao IPv4, o que torna estas implementações incompatíveis entre si.

Uma funcionalidade, que foi desenvolvida para o DHCPv6 e não existe no DHCPv4, é o *prefix delegation*, que serve para distribuir prefixos de rede. Seu mecanismo opera através de servidores DHCPv6 (podendo ser um roteador de delegação) e roteadores de requisição. Um roteador de requisição inicia o procedimento com uma requisição de prefixo enviada para rede com destino a todos os servidores DHCPv6. Os servidores que já estão pré-configurados com um pool de prefixos, respondem a este pedido feito pelo roteador enviando um prefixo IPv6. Ao receber esta resposta, o roteador fica encarregado de dividir o prefixo e redistribui-lo por suas interfaces. Os novos prefixos possuirão o tamanho /64 para que ao serem distribuidos aos host via router advertisement o procedimento de autoconfiguração stateless seja realizado.

Esta funcionalidade é utilizada em situações que o servidor DHCPv6 não possui nenhum conhecimento sobre a topologia de rede a qual o roteador requisitante está conectado e, também, desconhece outras informações além da indentidade do roteador requisitante para escolher o prefixo.

Roteiro Experimental

1. Caso não esteja utilizando a máquina virtual fornecida pelo NIC.br é preciso, antes de começar a experiência, instalar alguns softwares para auxiliar no aprendizado (caso contrário vá para o passo 2). Siga o passo a seguir para realizar as instalações:
    a. Para fazer algumas verificações durante o experimento será necessário a instalação do programa **Wireshark** que facilita a visualização dos pacotes enviados na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

b. O **dhcpd**, que é o serviço reponsável pelas tarefas de servidor do DHCP. Para instalá-lo, baixe a última versão do pacote ‘*tar.gz*’ no site http://www.isc.org/software/dhcp (acessado em 10/04/2012) e utilize os seguintes comandos em um Terminal:

```
$ cd <pasta onde o arquivo foi baixado>
$ tar xf dhcp-4.2.3-P2.tar.gz
$ cd dhcp-4.2.3-P2/
$ ./configure
$ make
$ sudo make install
```

***Obs**: Lembre-se de utilizar os números corretos de versão para extrair o pacote e acessar a pasta com os arquivos de instalação. Depois do comando ‘`sudo`‘ será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

c. O **dibbler-client**, que realiza as funções de cliente DHCP. Para instalá-lo, baixe a última versão (0.8.2 ou superior) do código fonte no site http://klub.com.pl/dhcpv6/ (acessado em 10/04/2012) e utilize os seguintes comandos em um Terminal:

```
$ cd <pasta onde o arquivo foi baixado>
$ tar xf dibbler-0.8.2.tar.gz
$ cd dibbler-0.8.2/
$ ./configure
$ make
$ sudo make install
```

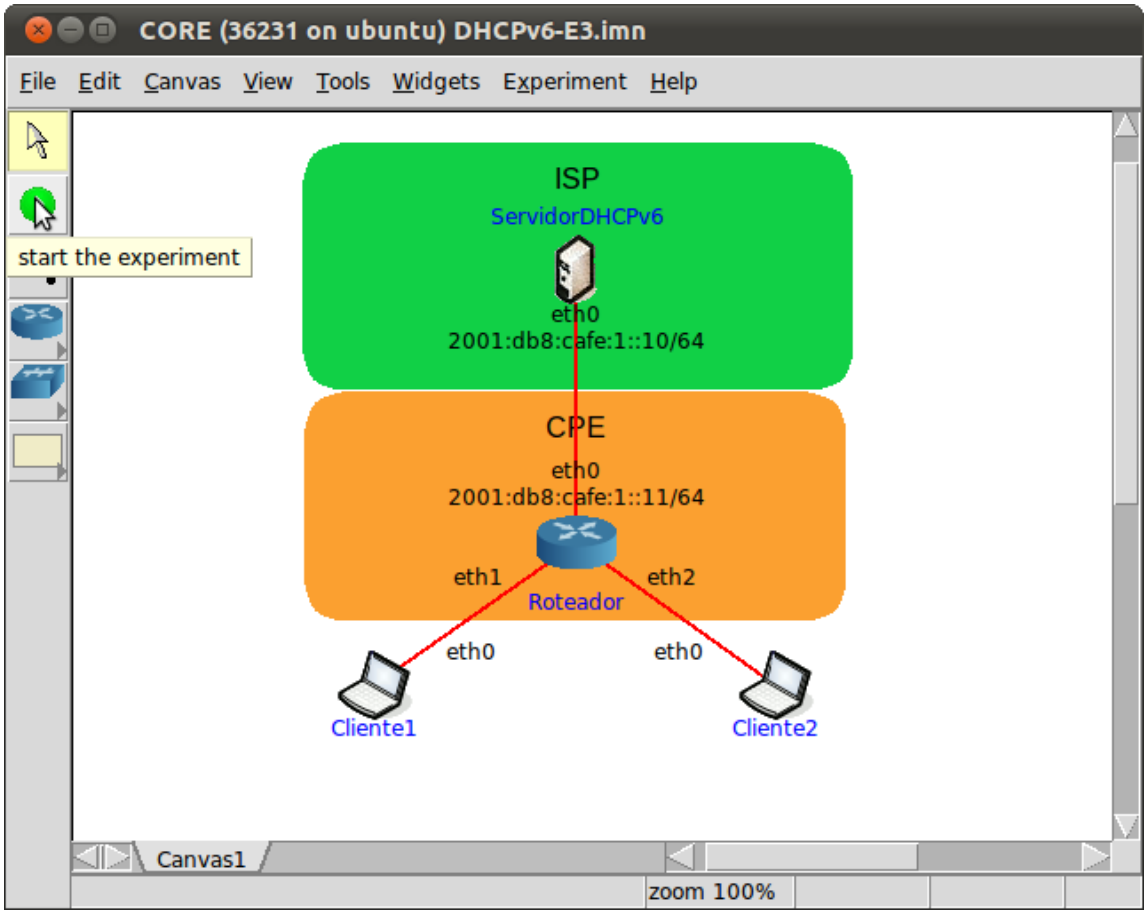
***Obs**: Lembre-se de utilizar os números corretos de versão para extrair o pacote e acessar a pasta com os arquivos de instalação. Depois do comando '`sudo`' será solicitada a senha do usuário core. Digite "core" para prosseguir com a instalação.

d. O **radvd**, que realiza o envio do tamanho do valor de MTU a ser configurado. Para instalá-lo use o comando:

```
$ sudo apt-get install radvd
```

Caso haja algum problema no pacote oferecido, pode-se baixa-lo no endereço: http://packages.ubuntu.com/hardy/radvd (acessado em 2012).

2. Inicie o CORE e abra o arquivo "**DHCPv6-E3.imn**" localizado no diretório do desktop "Funcionalidades/DHCPv6", da máquina virtual do NIC.br. A seguinte topologia deve aparecer:



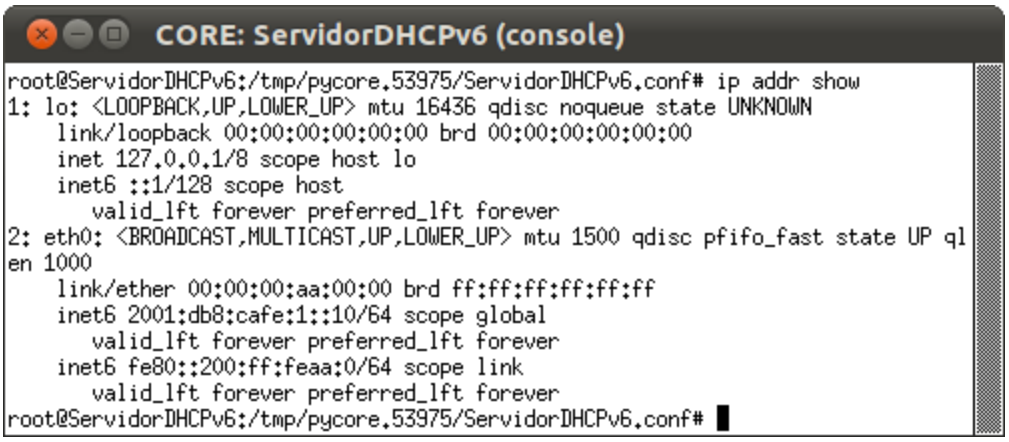
3. Verifique a configuração dos nós da topologia.
    a. Inicie a simulação realizando um dos seguintes passos:
        i. clique no botão 
        ii. utilize o menu Experiment > Start.

    b. Espere até que o CORE termine de iniciar a simulação e abra o terminal da máquina ServidorDHCPv6, com um duplo-clique sobre ela.

    c. Observe a configuração de rede do ServidorDHCPv6 através do seguinte comando:

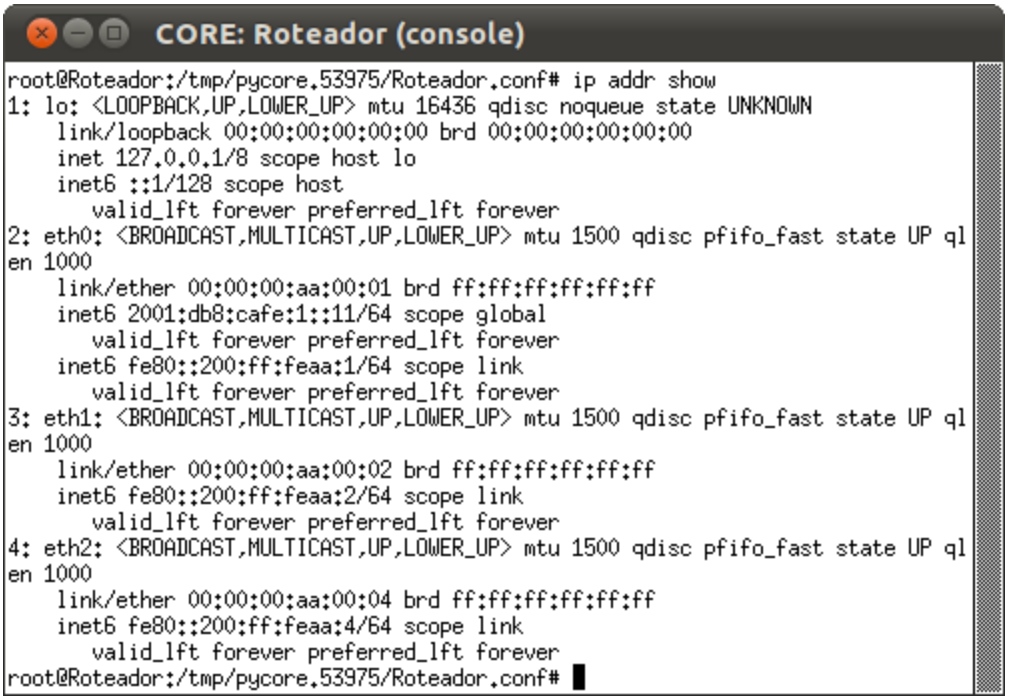
```
# ip addr show
```

O resultado deve ser:



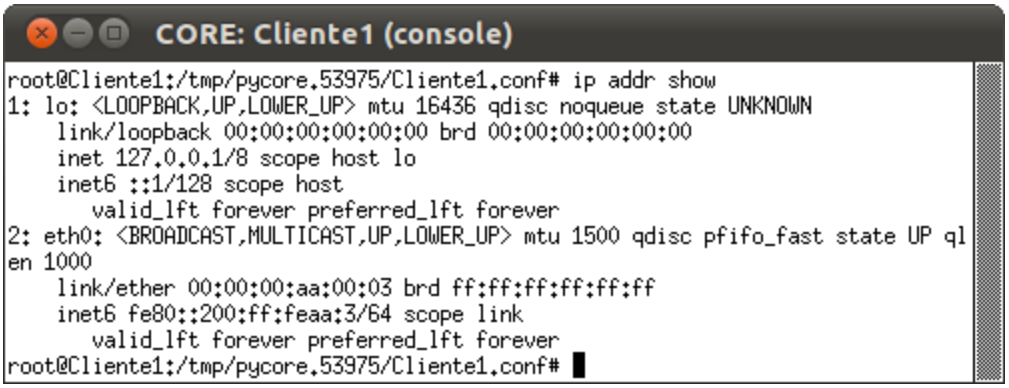
***Obs**: A partir deste comando é possível observar os endereços das interfaces.

d. Observe a configuração do Roteador com o mesmo comando.
O resultado deve ser:



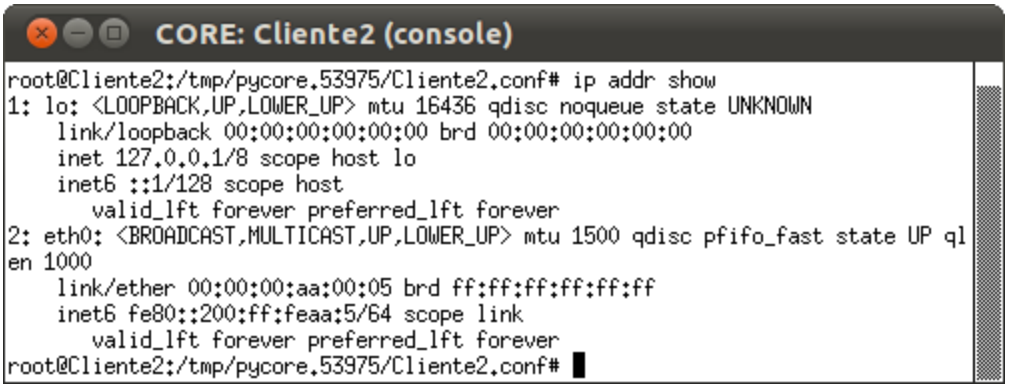
***Obs**: A partir deste comando é possível observar os endereços das interfaces.

e. Observe a configuração do Cliente1 com o mesmo comando.
O resultado deve ser:



***Obs**: A partir deste comando é possível observar os endereços das interfaces.

f. Observe a configuração do Cliente2 com o mesmo comando.
O resultado deve ser:



***Obs**: A partir deste comando é possível observar os endereços das interfaces.

4. Configure o dhcpd no Servidor para enviar prefixo ao roteador.
   a. Abra o terminal do ServidorDHCPv6;

   b. Crie dois arquivos com os seguintes nomes: dhcpd.conf e dhcp.leases. Para fazer isso digite os comandos:

```
# touch dhcpd.conf
# touch dhcpd.leases
```

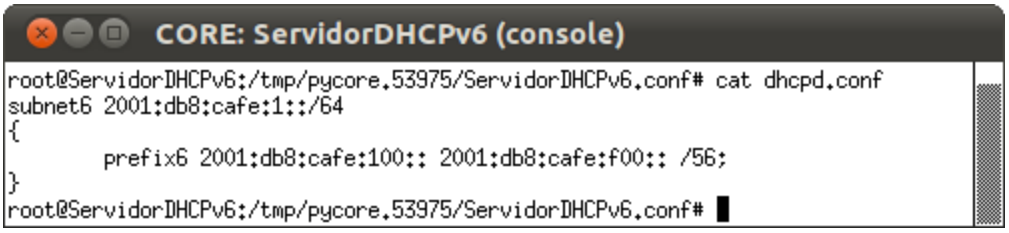
   O resultado deve ser:

   c. Edite o arquivo dhcpd.conf. Ele deverá conter somente as linhas:

```
subnet6 2001:db8:cafe:1::/64
{
      prefix6 2001:db8:cafe:100:: 2001:db8:cafe:f00:: /56;
}
```

   ***Obs**: O campo prefix6 contém uma faixa de prefixos de endereços ipv6 que devem ser distribuídos entre os roteadores requisitantes, com o devido tamanho de prefixos.

O resultado deve ser:



***Obs**: um editor de texto presente na máquina virtual que pode ser utilizado é o **nano**. Para usá-lo digite no terminal:

```
# nano dhcpd.conf
```

No nano, a sequência utilizada para salvar o arquivo é CTRL-O e para sair é CTRL-X.

d. Inicie o programa do dhcp através do comando:

```
# dhcpd -6 -cf dhcpd.conf -lf dhcpd.leases
```

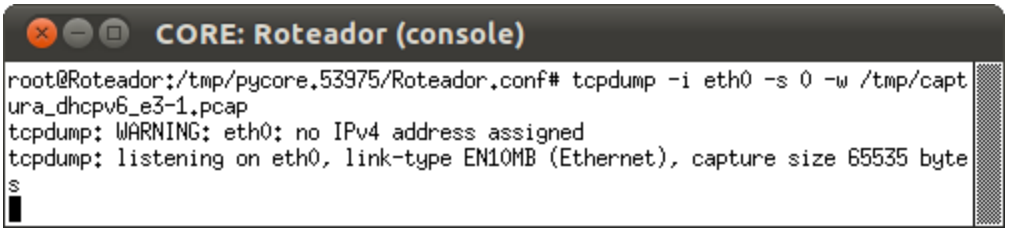
O resultado deve ser:

4. Configure o dibbler-client no roteador para receber o prefixo IPv6 do servidor.
   a. Abra o terminal da máquina Roteador.

   b. Utilize o seguinte comando para iniciar a captura de pacotes em sua interface interface:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_dhcpv6_e3-1.pcap
```

O resultado deve ser:



***Obs**: Não feche este terminal até o final do experimento, uma vez que, isso ocasionará no término da execução do comando "tcpdump" e prejudicará o andamento da experiência.

c. Abra um novo terminal na máquina Roteador.

d. Dentro da pasta '*/etc/dibbler*', crie um arquivo chamado '*client.conf*' com o comando:

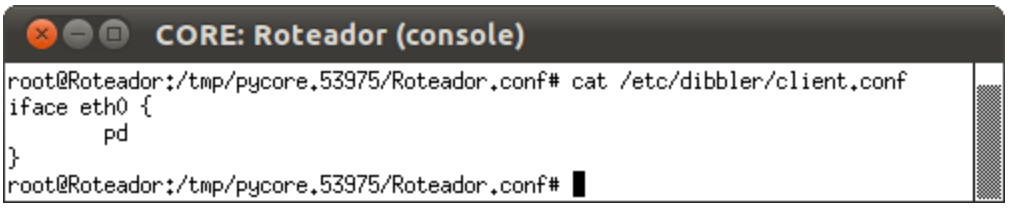
```
# touch /etc/dibbler/client.conf
```

O resultado deve ser:

e. Edite este arquivo deixando-o com o seguinte texto:

```
iface eth0 {
      pd
}
```

O resultado deve ser:

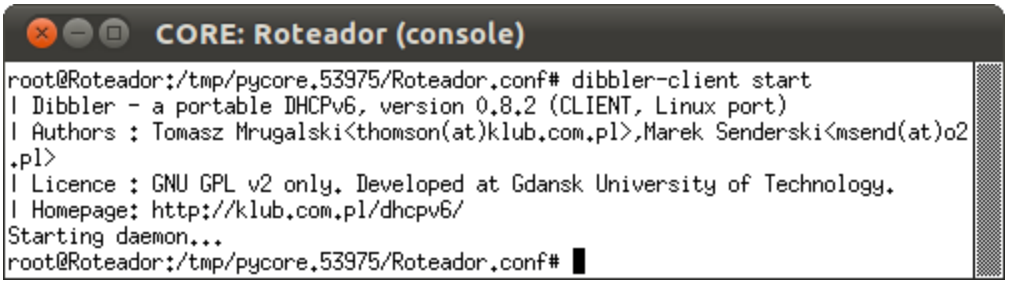


***Obs**: A configuração "pd" significa que o cliente irá requisitar o serviço de delegação de prefixos ao servidor (alcançado pela interface eth0).

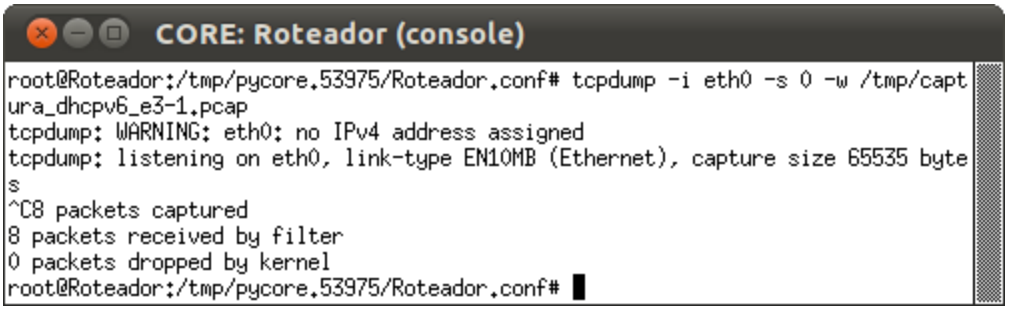
f. Inicie o programa dibbler-client através do comando:

```
# dibbler-client start
```

O resultado deve ser:



g. Espere alguns segundos e reabra o terminal do roteador, que contem o programa tcpdump funcionando, e encerre a captura de pacotes através da sequência Ctrl+C.
O resultado deve ser:



***Obs**: A quantidade de pacotes pode variar de acordo com o tempo esperado para dar o comando Ctrl+C.

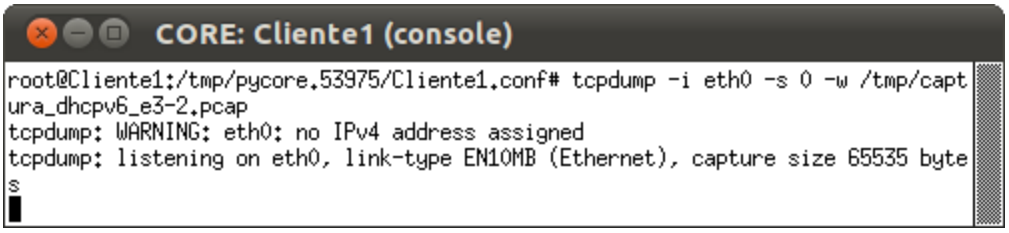
A partir deste momento o roteador recebeu um prefixo /56 para administrar. Ele irá então dividir este prefixo entre todas as suas interfaces com tamanhos /64.

5. Analise o envio de prefixos aos clientes. Com estes prefixos os clientes então poderão autoconfigurar suas interfaces.
   a. Abra o terminal do Cliente1.

   b. Utilize o seguinte comando para iniciar a captura de pacotes em sua interface interface:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_dhcpv6_e3-2.pcap
```

O resultado deve ser:



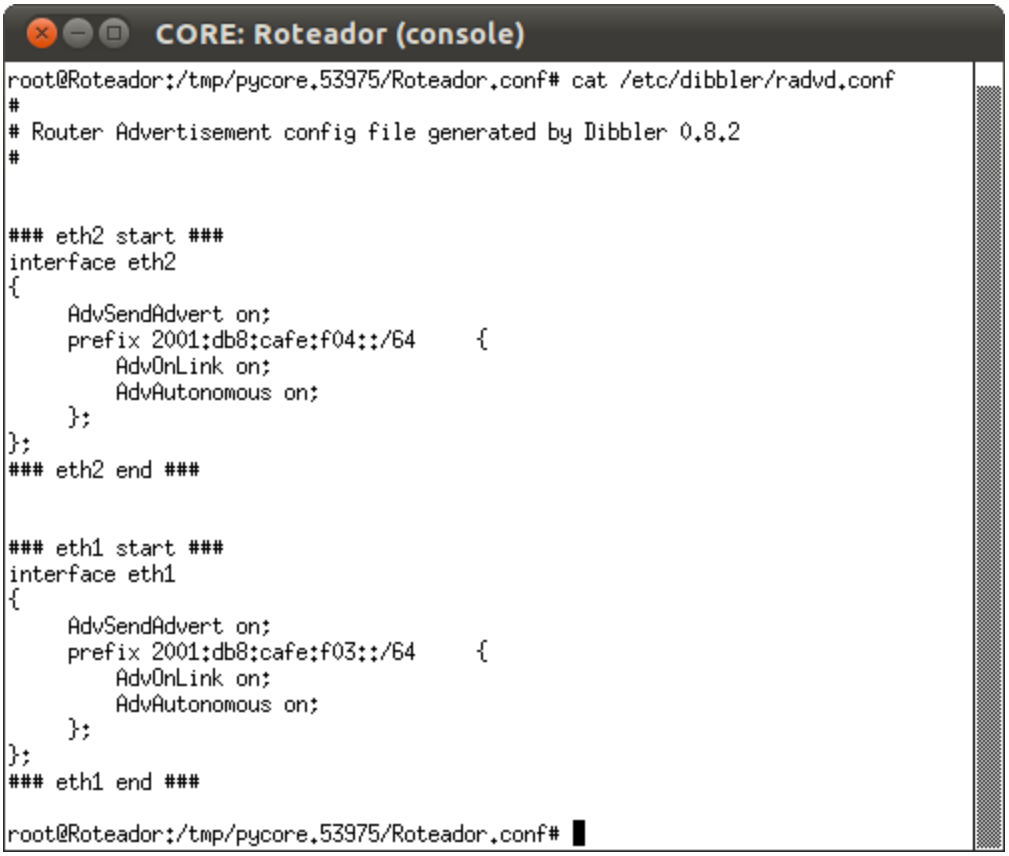
***Obs**: Não feche este terminal até o final do experimento, uma vez que, isso ocasionará no término da execução do comando "tcpdump" e prejudicará o andamento da experiência.

c. Abra o terminal da máquina Roteador.

d. Observe o conteudo do arquivo 'radvd.conf' localizado dentro da pasta '/etc/dibbler'. Utilize o comando:

```
# cat /etc/dibbler/radvd.conf
```

O resultado deve ser:



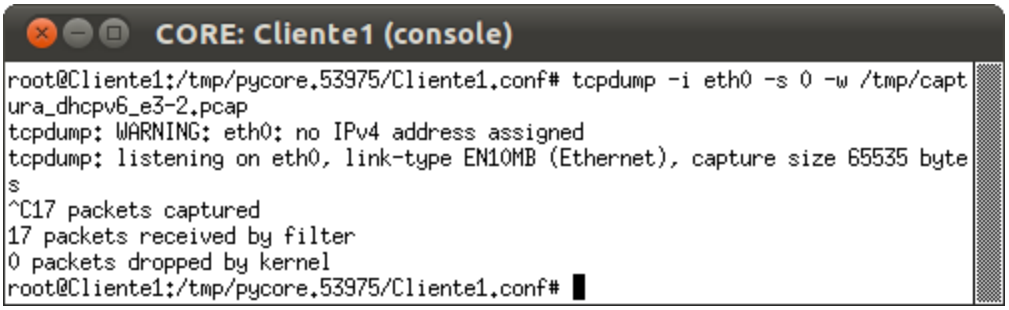
***Obs**: Este arquivo é gerado automáticamente ao receber o prefixo do Servidor /56 , para ministrar prefixos com tamanhos /64 entre suas interfaces habilitadas.

e. Inicie o programa radvd utilizando o arquivo de configuração gerado pelo dibbler, através do comando:

```
# radvd -C /etc/dibbler/radvd.conf
```

O resultado deve ser:

f. Espere alguns segundos e reabra o terminal do Cliente1, que contem o programa tcpdump funcionando, e encerre a captura de pacotes através da sequência Ctrl+C.
O resultado deve ser:



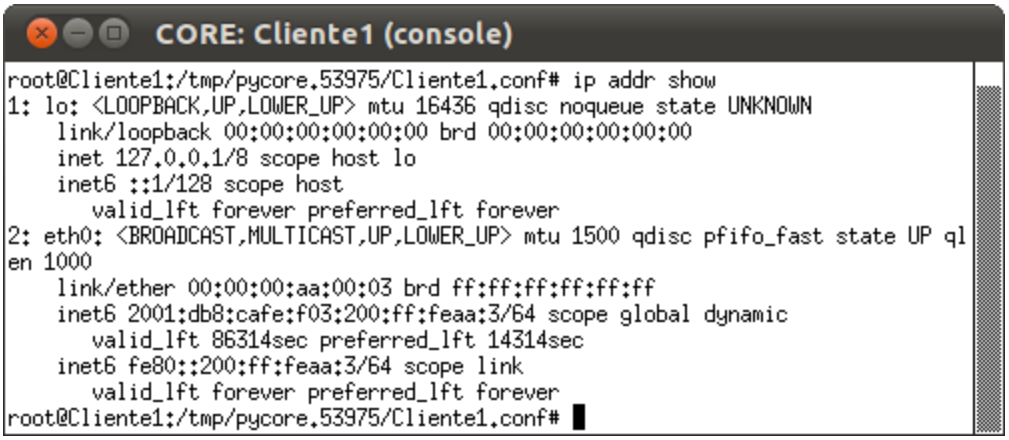
***Obs**: A quantidade de pacotes pode variar de acordo com o tempo esperado para dar o comando Ctrl+C.

6. Verifique os endereços IPv6 adquiridos pelas interfaces dos clientes e teste a sua conectividade.
   a. Abra o terminal do Cliente1.

   b. Observe a configuração de rede do Cliente1 através do seguinte comando:

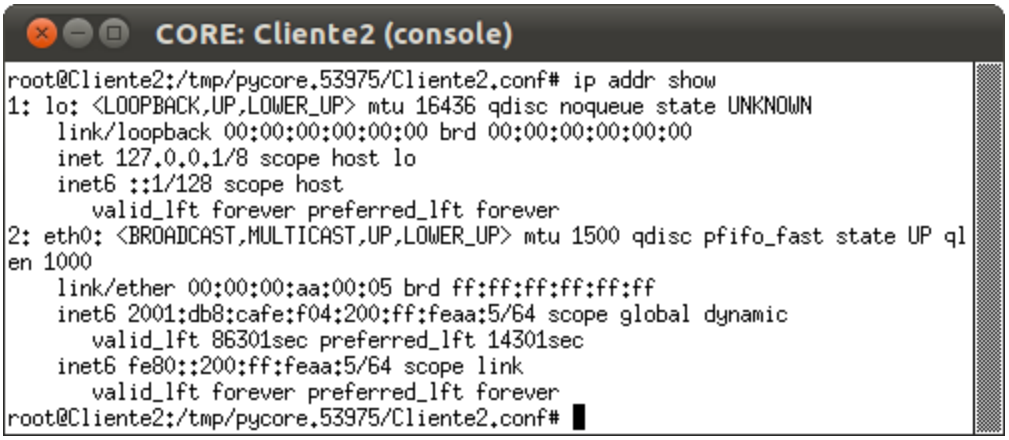
```
# ip addr show
```

O resultado deve ser:



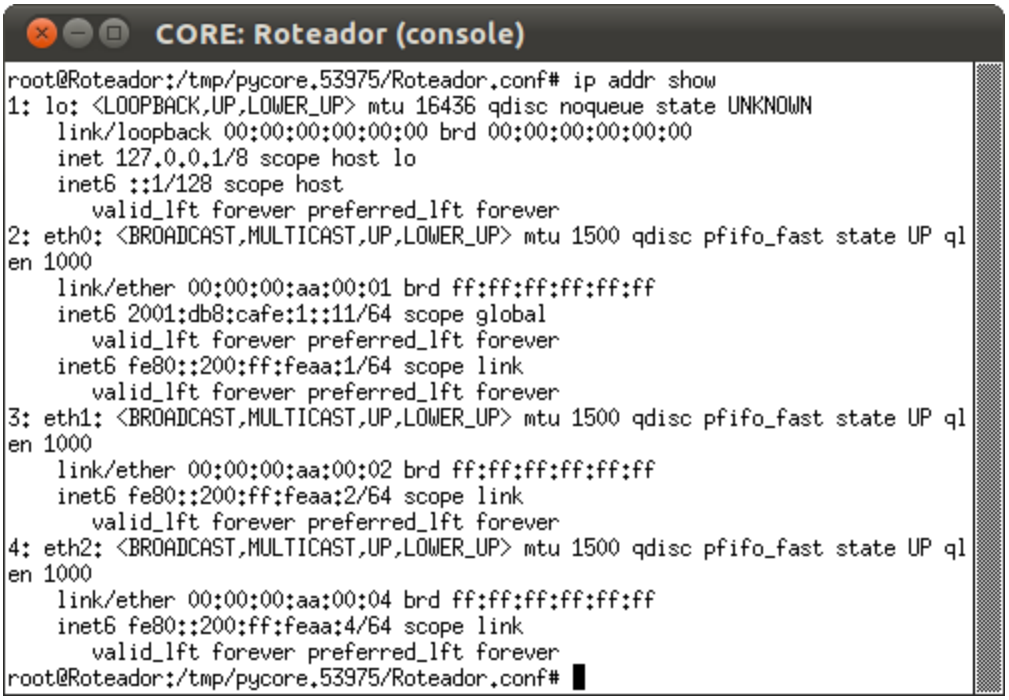
***Obs**: Note que o Cliente1 montou um endereço IPv6 global baseado no prefixo 2001:db8:cafe:f03::/64. Isso aconteceu por ele estar conectado diretamente a interface eth1 do roteador.

c. Observe a configuração do Cliente2 com o mesmo comando.
O resultado deve ser:



***Obs**: Note que o Cliente2 montou um endereço IPv6 global baseado no prefixo 2001:db8:cafe:f04::/64. Isso aconteceu por ele estar conectado diretamente a interface eth2 do roteador.

d. Observe a configuração do Roteador com o mesmo comando.
O resultado deve ser:

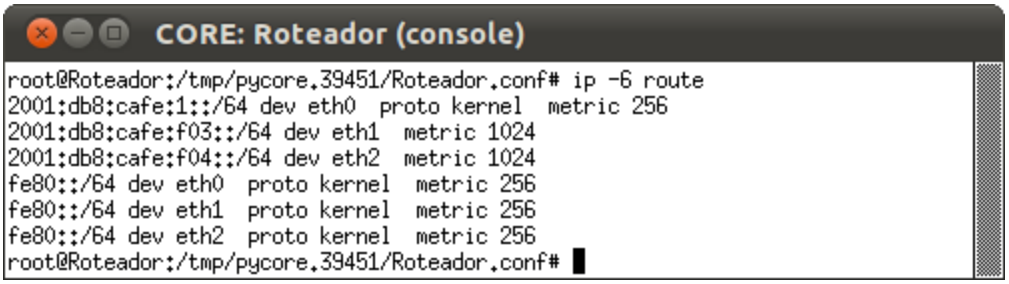


***Obs**: Note que as interfaces apesar de delegarem prefixos, não configuram automáticamente endereços IPv6 globais.

e. Observe as rotas do Roteador através do seguinte comando:

```
# ip -6 route
```

O resultado deve ser:



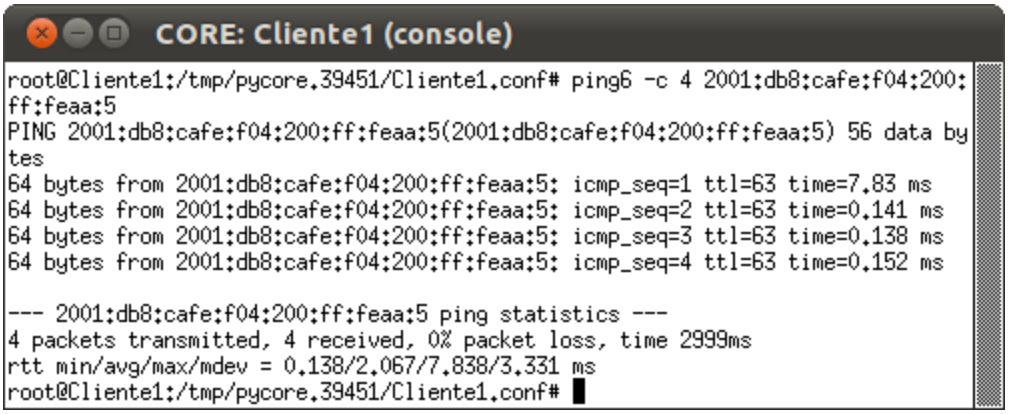
***Obs**: Note que ao mesmo tempo que o roteador delega prefixos, ele já configura rotas para eles.

f. Abra o terminal do Cliente1 para testar a conectividade IPv6.

g. Inicie uma comunicação ping6 com o Cliente2 através do seguinte comando:

```
# ping6 -c 4 2001:db8:cafe:f04:200:ff:feaa:5
```

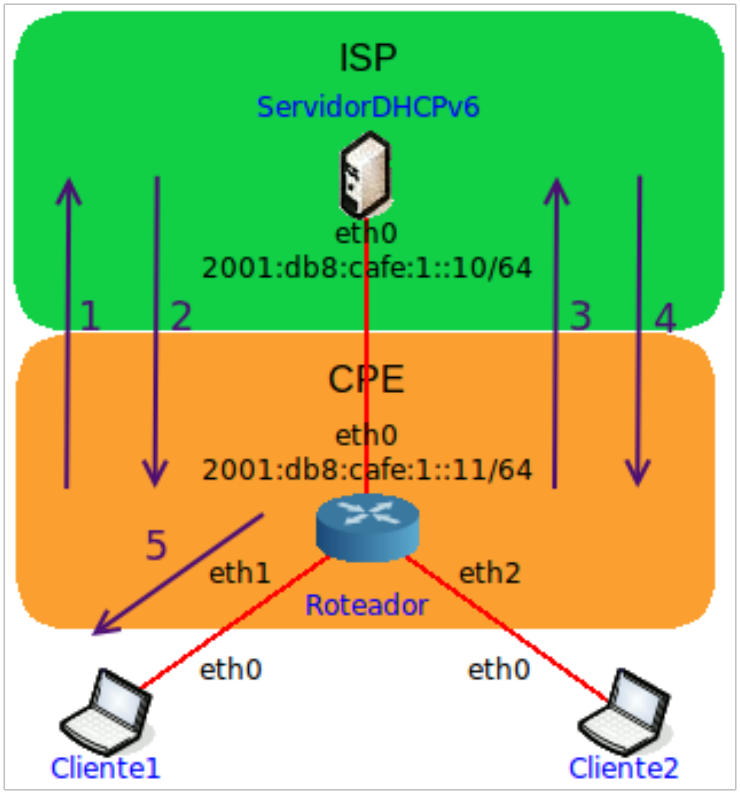
O resultado deve ser:



***Obs**: O endereço IPv6 deve ser o mesmo do Cliente2 adquirido via Router Advertisement, mostrado anteriormente.

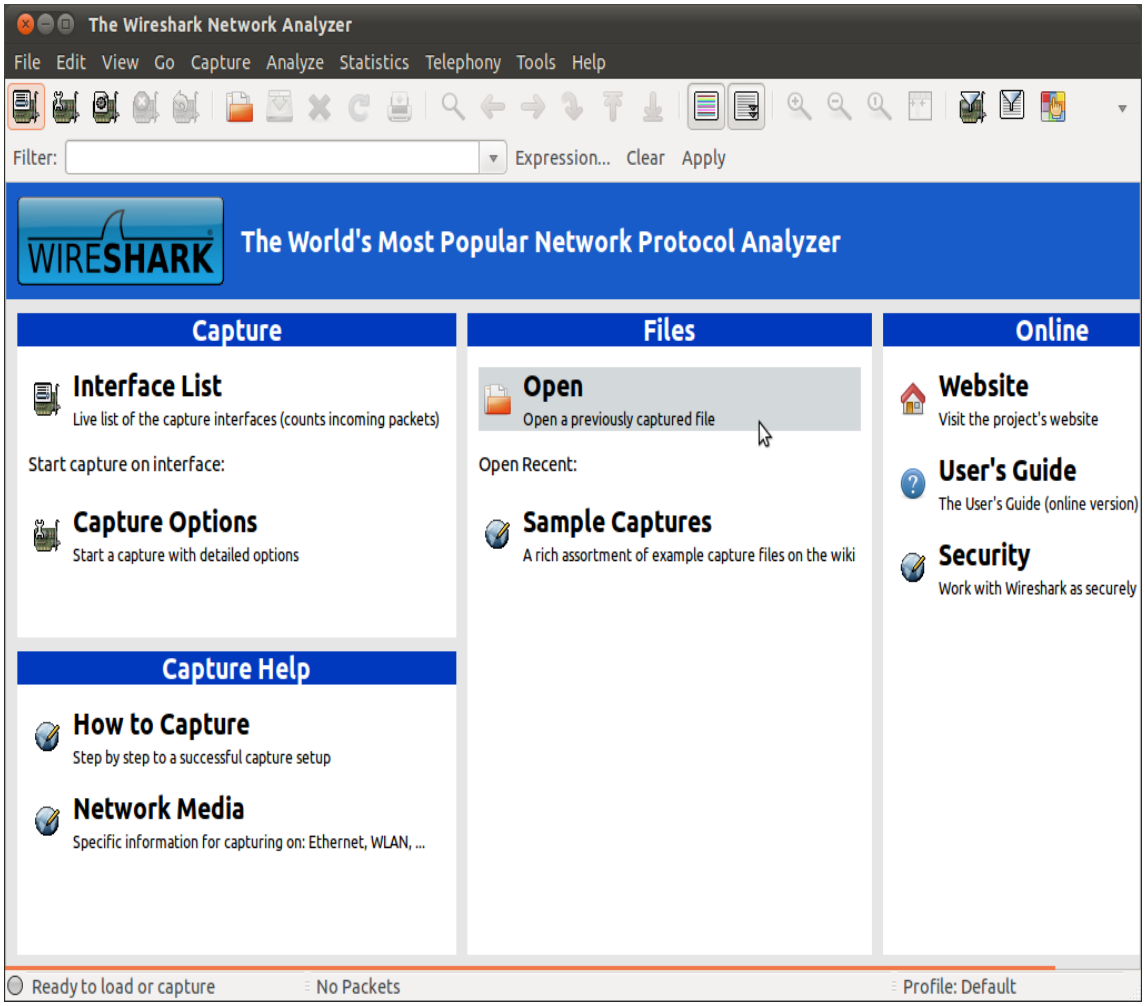
7. Encerre a simulação com uma das seguintes ações:
   a. aperte o botão 
   b. utilize o menu Experiment > Stop.

Observe a troca de mensagens ocorrida, que será analisada posteriormente:



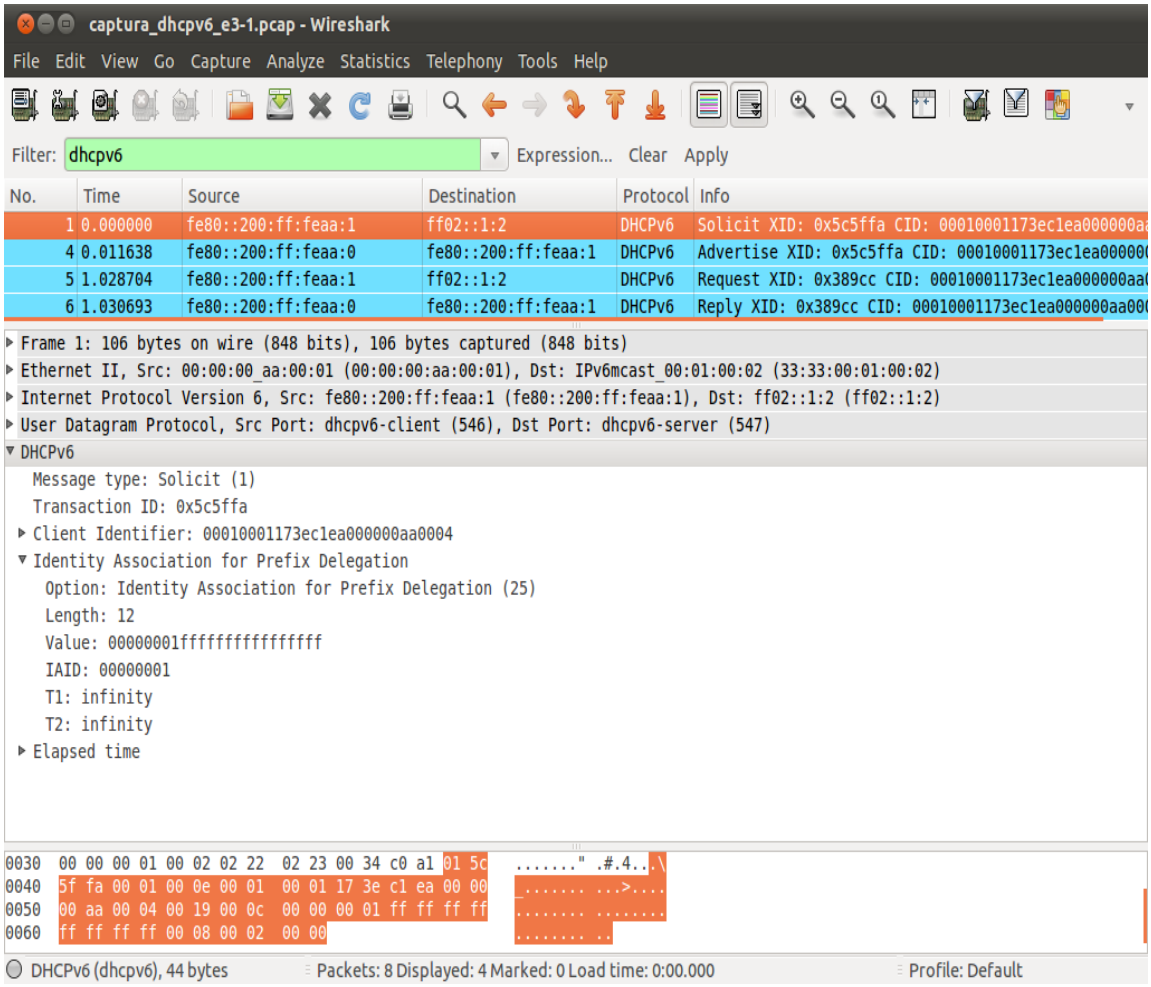
8. A verificação dos pacotes capturados será realizada através do programa Wireshark. Para iniciá-lo execute o seguinte comando em um terminal da máquina virtual:

```
$ wireshark
```



a. Abra o arquivo **/tmp/captura_dhcpv6_e3-1.pcap** com o menu File>Open;
b. Procure pelos pacotes *Solicit*, *Advertise*, *Request* e *Reply*. Analise-os e veja que os dados contidos nos pacotes conferem com o que foi passado na teoria.

1 - Solicit:



***Obs**: o filtro dhcpv6 pode ser usado para ajudar a filtrar as mensagens.

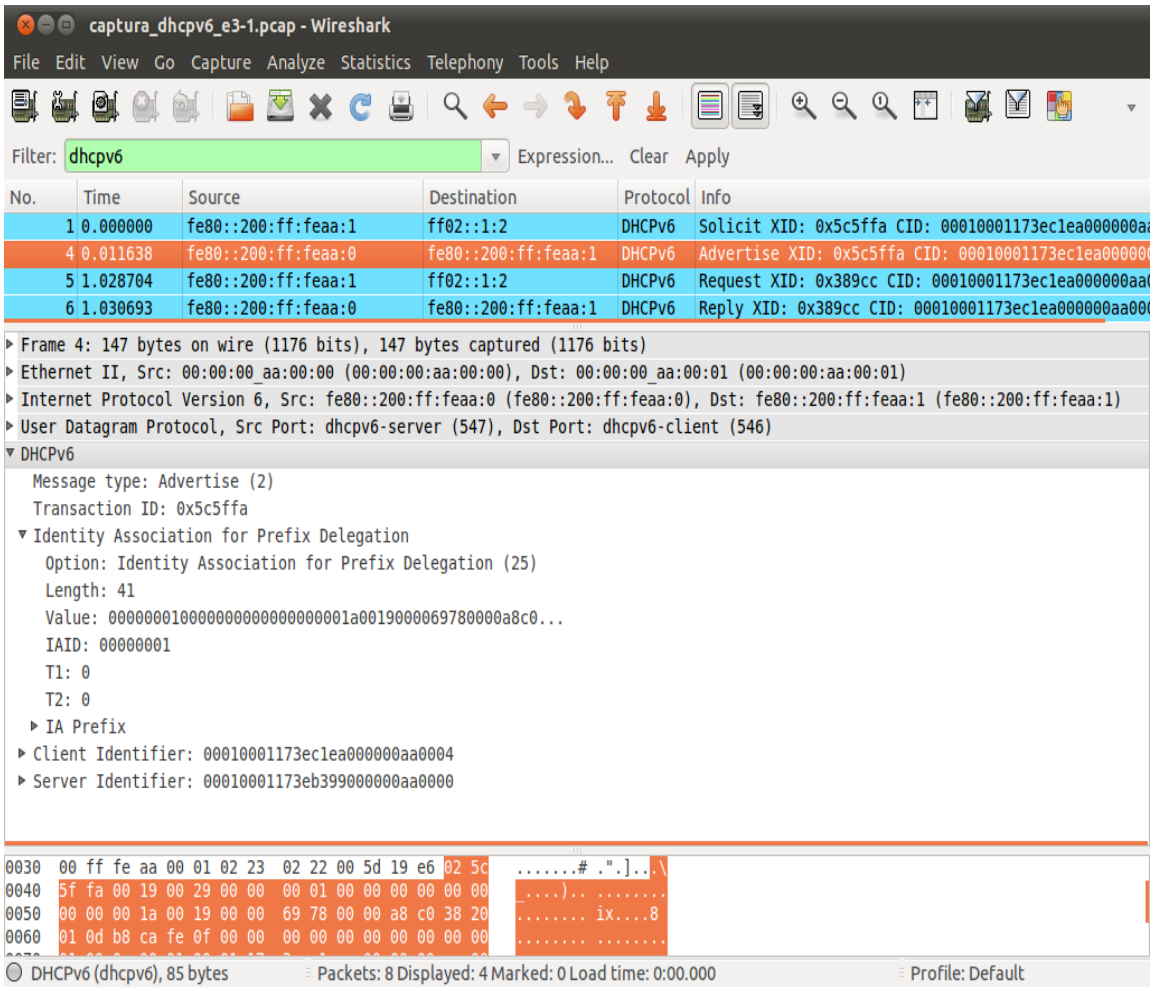
Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço (33:33:00:01:00:02) sendo que o prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet e o sufixo 00:01:00:02 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a origem é o MAC address da interface da máquina que enviou a solicitação (00:00:00:aa:00:01).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho, no caso, o valor 0x11 refere-se à uma mensagem UDP.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface diretamente conectada ao enlace ao qual se fez a solicitação (fe80::200:ff:feaa:1).
- **Destination (IPv6)**: o destino é o endereço *Multicast Agent DHCP* (ff02::1:2).
- **Source port (UDP)**: indica a porta que se encontra o serviço dhcpv6-client cujo o valor

é 546.

- **Destination port (UDP)**: indica a porta que se encontra o serviço dhcpv6-server no servidor. Seu valor é 547.
- **Message type (DHCPv6)**: indica através do valor 1 que o tipo da mensagem é Solicit;
- **Client Identifier (DHCPv6)**: contém dados da identificação única do cliente baseada no endereço fisico.
- **Identity Association for Prefix Delegation (DHCPv6)**: serve para requisitar um prefixo IPv6 para o servidor.

2 - Advertise:



***Obs**: o filtro dhcpv6 pode ser usado para ajudar a filtrar as mensagens.

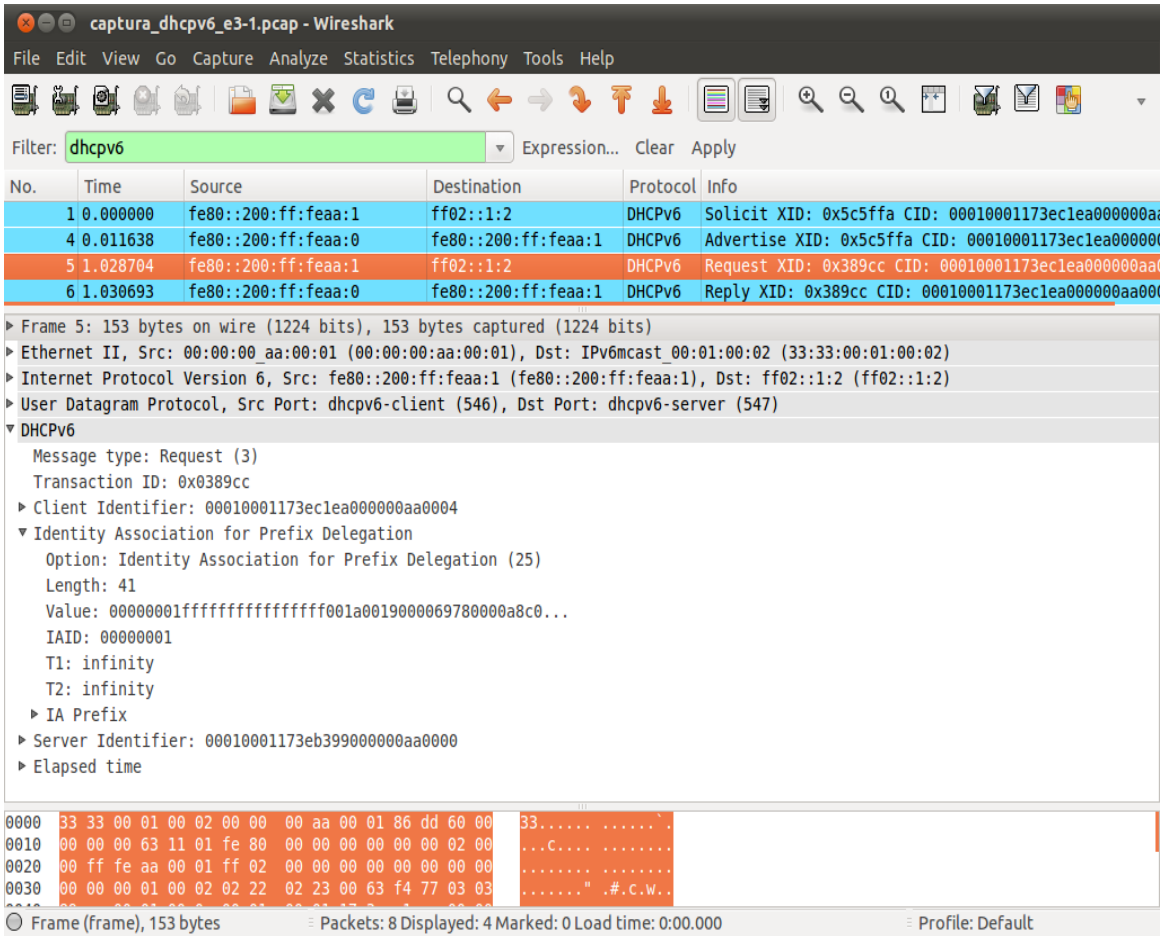
Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço MAC da interface da máquina solicitante (00:00:00:aa:00:01).
- **Source (Ethernet)**: a origem é o MAC address da interface da máquina que enviou a

resposta (00:00:00:aa:00:00).

- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho, no caso, o valor 0x11 refere-se à uma mensagem UDP.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface do dispositivo que enviou a mensagem, ou seja, do Servidor DHCP6 (fe80::200:ff:feaa:0).
- **Destination (IPv6)**: o destino é o endereço unicast de link local da máquina solicitante (fe80::200:ff:feaa:1).
- **Source port (UDP)**: indica a porta que se encontra o serviço dhcpv6-server cujo o valor é 547.
- **Destination port (UDP)**: indica a porta que se encontra o serviço dhcpv6-client cujo o valor é 546.
- **Message type (DHCPv6)**: indica através do valor 2 que o tipo da mensagem é Advertise;
- **Identity Association for prefix delegation (DHCPv6)**: serve para carregar o prefixo IPv6 e suas caracteristicas para o cliente.
    - IA Prefix: contém o prefixo e suas caracteristicas, que o cliente irá utilizar em sua autoconfiguração (2001:db8:cafe:f00::/56).
- **Client Identifier (DHCPv6)**: contém dados da identificação única do cliente baseada em seu endereço fisico.
- **Server Identifier (DHCPv6)**: contém dados da identificação única do servidor baseada em seu endereço fisico.

3 - Request:



***Obs**: o filtro dhcpv6 pode ser usado para ajudar a filtrar as mensagens.

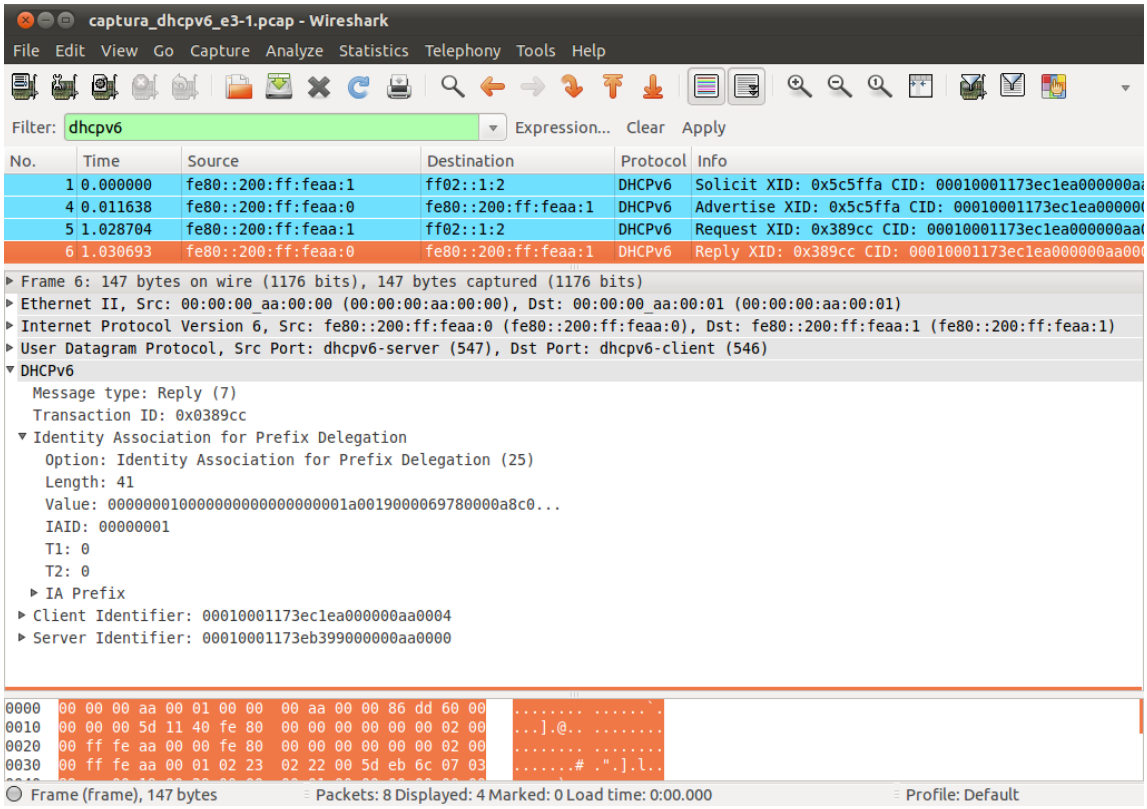
Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço (33:33:00:01:00:02) sendo que o prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet e o sufixo 00:01:00:02 indica os últimos 32 bits do endereço multicast IPv6 da mensagem.
- **Source (Ethernet)**: a origem é o MAC address da interface da máquina cliente (00:00:00:aa:00:01).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho, no caso, o valor 0x11 refere-se à uma mensagem UDP.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface do dispositivo que enviou a mensagem, ou seja, do cliente (fe80::200:ff:feaa:1).
- **Destination (IPv6)**: o destino é o endereço *Multicast Agent DHCP* (ff02::1:2).
- **Source port (UDP)**: indica a porta que se encontra o serviço dhcpv6-client cujo o valor é 546.
- **Destination port (UDP)**: indica a porta que se encontra o serviço dhcpv6-server cujo o

valor é 547.

- **Message type (DHCPv6)**: indica através do valor 3 que o tipo da mensagem é Request;
- **Client Identifier (DHCPv6)**: contém dados da identificação única do cliente baseada em seu endereço fisico.
- **Identity Association for prefix delegation (DHCPv6)**: serve para confirmar o prefixo IPv6 e suas caracteristicas recebidas.
    - IA Prefix: contém o prefixo e suas caracteristicas, que o cliente irá utilizar em sua autoconfiguração (2001:db8:cafe:f00::/56).
- **Server Identifier (DHCPv6)**: contém dados da identificação única do servidor baseada em seu endereço fisico.

4 - Reply:



***Obs**: o filtro dhcpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

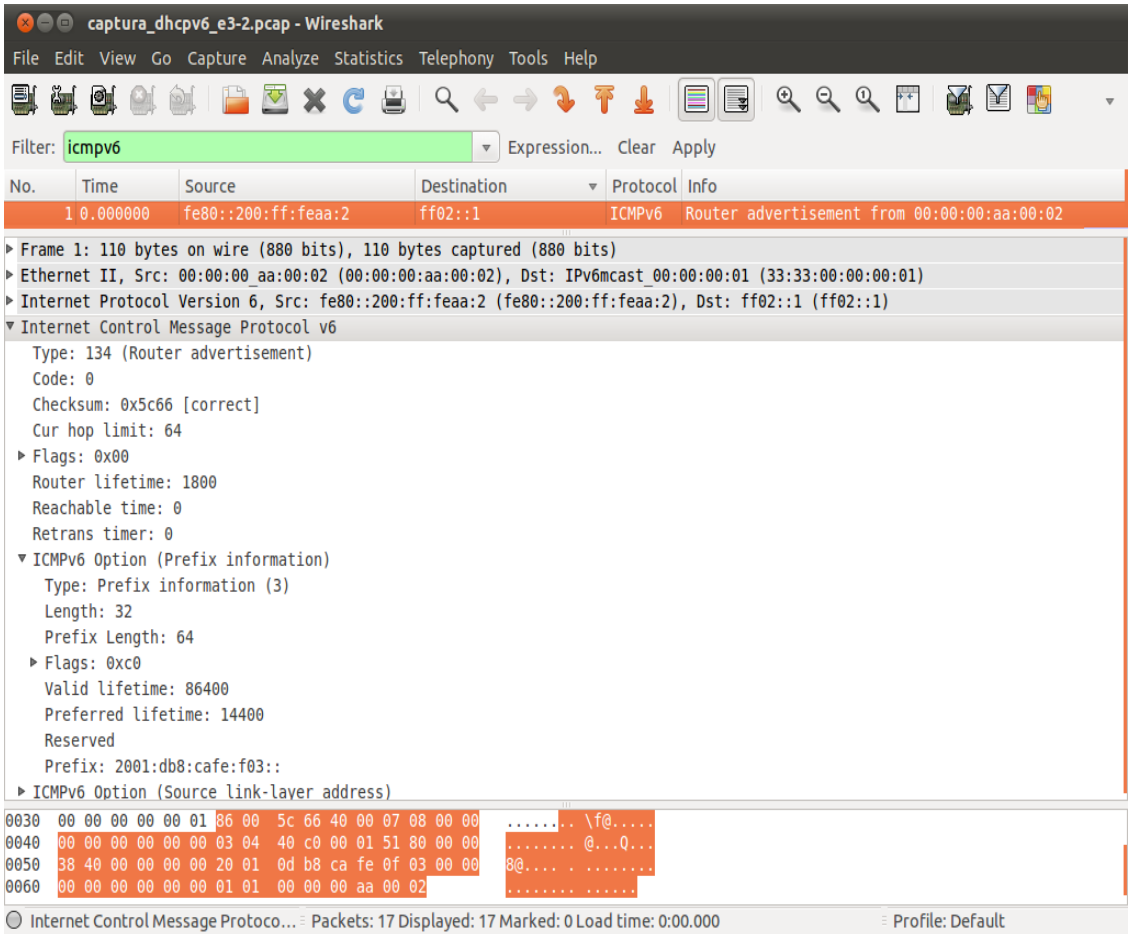
- **Destination (Ethernet)**: o destino é o endereço MAC da interface da máquina cliente (00:00:00:aa:00:01).
- **Source (Ethernet)**: a origem é o MAC address da máquina que está enviando a resposta (00:00:00:aa:00:00).
- **Type (Ethernet)**: indica que a mensagem utiliza o protocolo IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho, no caso, o valor 0x11 refere-se

à uma mensagem UDP.

- **Source (IPv6)**: a origem é o endereço IP de link local da interface do dispositivo que enviou a mensagem, ou seja, do Servidor DHCPv6 (fe80::200:ff:feaa:0).
- **Destination (IPv6)**: o destino é o endereço IPv6 unicast de link local do cliente (fe80::200:ff:feaa:1).
- **Source port (UDP)**: indica a porta que se encontra o serviço dhcpv6-server cujo o valor é 547.
- **Destination port (UDP)**: indica a porta que se encontra o serviço dhcpv6-client cujo o valor é 546.
- **Message type (DHCPv6)**: indica através do valor 7 que o tipo da mensagem é Reply;
- **Identity Association for prefix delegation (DHCPv6)**: serve para confirmar o prefixo IPv6 e suas caracteristicas fornecidas.
    - IA Prefix: contém o prefixo e suas caracteristicas, que o cliente irá utilizar em sua autoconfiguração (2001:db8:cafe:f00::/56).
- **Client Identifier (DHCPv6)**: contém dados da identificação única do cliente baseada em seu endereço fisico.
- **Server Identifier (DHCPv6)**: contém dados da identificação única do servidor baseada em seu endereço fisico.

c. Abra o arquivo **/tmp/captura_dhcpv6_e3-2.pcap** com o menu File>Open;
d. Procure por um pacote '*Router Advertisement*' que contenha a opção ICMPv6 *Prefix Information*. Analise-o e veja que seus dados conferem com o que foi apresentado na teoria.

5 - Router Advertisement:



***Obs**: o filtro icmpv6 pode ser usado para ajudar a filtrar as mensagens.

Campos importantes:

- **Destination (Ethernet)**: o destino é o endereço MAC (33:33:00:00:00:01) sendo que o prefixo 33:33 indica que a mensagem é um multicast na camada Ethernet e, o sufixo 00:aa:00:01 indica os últimos 32 bits do endereço multicast IPv6 desta mensagem.
- **Source (Ethernet)**: a origem é o endereço MAC do roteador que enviou a mensagem (00:00:00:aa:00:02).
- **Type (Ethernet)**: indica que a mensagem utiliza IPv6 (x86dd).
- **Next Header (IPv6)**: indica qual é o próximo cabeçalho (de extensão do IPv6), no caso, o valor 58(0x3a) refere-se à uma mensagem ICMPv6.
- **Source (IPv6)**: a origem é o endereço IP de link local da interface que originou a mensagem, neste caso, do roteador (fe80::200:ff:feaa:2).
- **Destination (IPv6)**: o destino é o endereço *Multicast All nodes* (ff02::1).
- **Type (ICMPv6)**: indica que a mensagem é do tipo 134 (*Router Advertisement*).
- **ICMPv6 Option (ICMPv6)**: indica as opções do pacote ICMPv6:
    - Prefix Information

- *Type*: indica o tipo de dado da mensagem ICMPv6. No nosso caso, ela é do tipo "*Prefix information*";
- *Autonomous Address-Configuration Flag (A)*: indica se o prefixo deve ser utilizado para autoconfiguração stateless (1) .
- *Preferred Lifetime*: marca o tempo, em segundos, que o endereço é preferencial, ou seja, um endereço que pode ser utilizado indistintamente. O valor (0xffffffff) indica infinito.
- *Valid Lifetime*: marca o tempo, em segundos, de expiração do endereço gerado. O valor (0xffffffff) indica infinito.
- *Prefix*: contém o prefixo de rede a ser utilizado (`2001:db8:cafe:f03::`).
- *Prefix length*: contém o tamanho do prefixo da rede (64).

- Source Link Layer Address
  - *Type*: indica o tipo de dado da mensagem ICMPv6. No nosso caso, ela é do tipo "*Source link-layer address*";
  - *Link-layer address*: indica o endereço MAC da interface do roteador (00:00:00:aa:00:02).

IPV6 - Path MTU Discovery

Objetivo

O objetivo desta experiência é mostrar o funcionamento do mecanismo de *Path MTU (Maximum Transmit Unit) Discovery* do IPv6. Nela serão utilizadas duas redes ligadas por um roteador e configuradas com diferentes valores de MTU para mostrar como se dá a fragmentação de pacotes quando esses valores diferem no caminho da comunicação.

Para isso será utilizada a topologia descrita no arquivo: **Path-MTU-E1.imn**.

Introdução Teórica

Em uma rede de computadores, cada enlace possui uma limitação do tamanho máximo dos pacotes ser trafegados. Essa limitação chamada de MTU, tem de ser configurada em cada um dos nós da rede e, em geral, é dependente do meio físico do enlace. Caso pacotes maiores do que esse limiar tenham de ser transmitidos, estes devem ser fragmentados e remontados quando chegarem aos seus destinos.

Na transmissão de um pacote IPv4, cada roteador ao longo do caminho pode fragmentar pacotes caso estes sejam maiores do que o MTU do próximo enlace. Dependendo do desenho da rede, um pacote IPv4 pode ser fragmentado mais de uma vez durante seu trajeto. Ainda assim, independentemente da quantidade de fragmentações, ele só é reagrupado em seu destino final.

Já no IPv6, a fragmentação dos pacotes é realizada apenas na origem. Este procedimento não é realizado(e nem permitido) em roteadores intermediários com intuito de reduzir o overhead causado pelo cálculo dos cabeçalhos alterados.

Assim surgiu o protocolo *Path MTU Discovery* [RFC 1981] utilizado no início do processo de fragmentação para determinar, de forma dinâmica, qual o tamanho máximo permitido ao pacote. Ao realizar uma tranmissão de dados, ele identifica os MTUs de cada enlace no caminho até o destino.

Para funcionar, tal protocolo deve ser suportado por todos os nós do caminho. No entanto, implementações minimalistas de IPv6 podem omitir esse suporte ao utilizar 1280 Bytes como tamanho máximo de pacote.

Ao ser iniciado, o processo de Path MTU Discovery (PMTUD) assume que o MTU de todo o caminho é igual ao MTU do primeiro salto. Caso o tamanho dos pacotes enviados seja maior do que o suportado por algum roteador ao longo do caminho, este irá descartá-lo e enviará uma mensagem ICMPv6 *Packet Too Big* contendo tanto a mensagem de erro

quanto o valor do MTU do enlace seguinte. Após o recebimento dessa mensagem, o nó de origem passa a limitar o tamanho dos pacotes de acordo com o MTU indicado.

Esse procedimento termina quando o tamanho do pacote for igual ou inferior ao menor MTU do caminho, sendo que as iterações de troca de mensagens e redução do tamanho dos pacotes podem ocorrer diversas vezes até que se encontre o menor MTU. Caso o pacote seja enviado a um grupo multicast, o tamanho utilizado será o do menor PMTU de todo o conjunto de destinos.

De certo ponto de vista, o PMTUD pode parecer imperfeito dado que o roteamento dos pacotes é dinâmico e que cada pacote pode ser entregue através de uma rota diferente. Contudo, essas mudanças não são tão frequentes e, caso o valor do MTU diminua devido a uma mudança de rota, a origem receberá a mensagem de erro e reduzirá o valor do Path MTU.

Roteiro Experimental

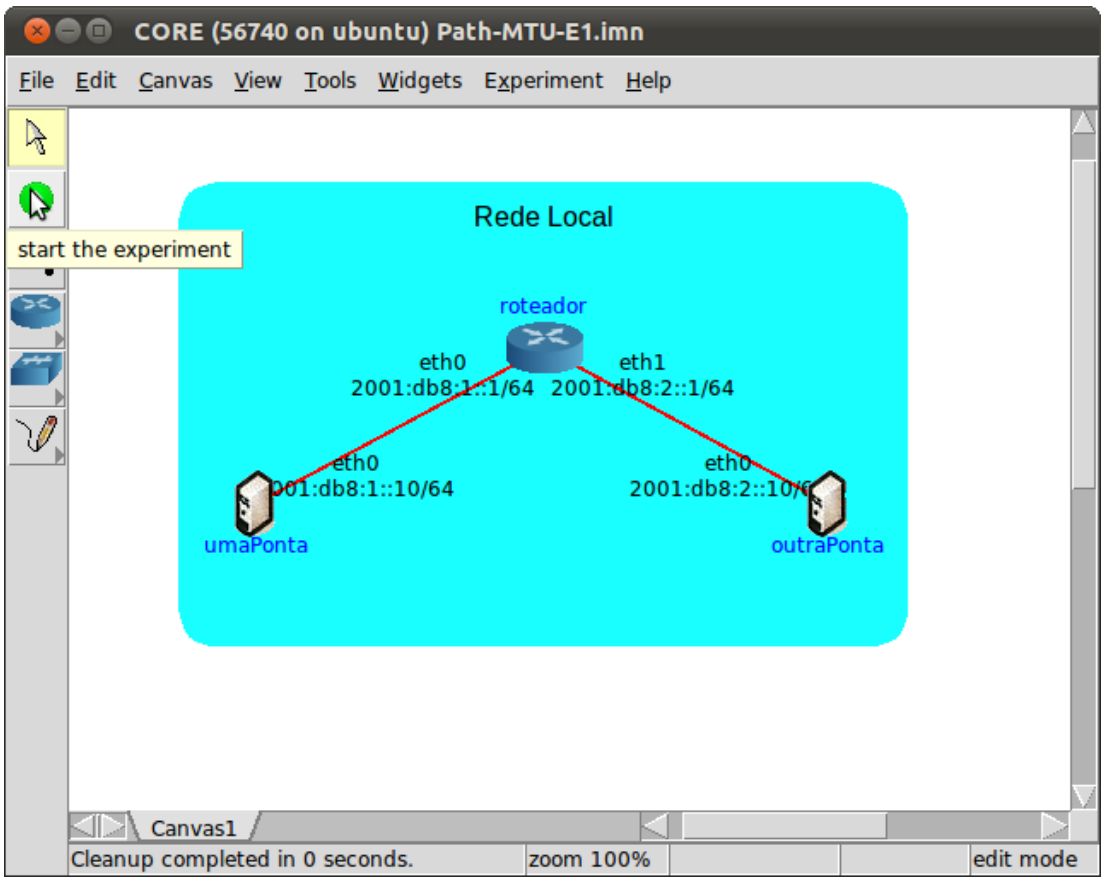
Experiência1 - ICMPv6: Packet Too Big

1. Caso não esteja utilizando a máquina virtual fornecida pelo NIC.br é preciso, antes de começar a experiência, instalar alguns softwares para auxiliar no aprendizado (caso contrário vá para o passo 2).
   Siga o passo a seguir para realizar a instalação:
   a. Para fazer algumas verificações durante o experimento será necessária a instalação do programa **Wireshark** que facilita a visualização dos pacotes enviados na rede. Para isso, na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

   Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

2. Inicie o CORE e abra o arquivo "**Path-MTU-E1.imn**" localizado no diretório do desktop "Funcionalidades/PathMTU", da máquina virtual do NIC.br. A seguinte topologia inicial deve aparecer:



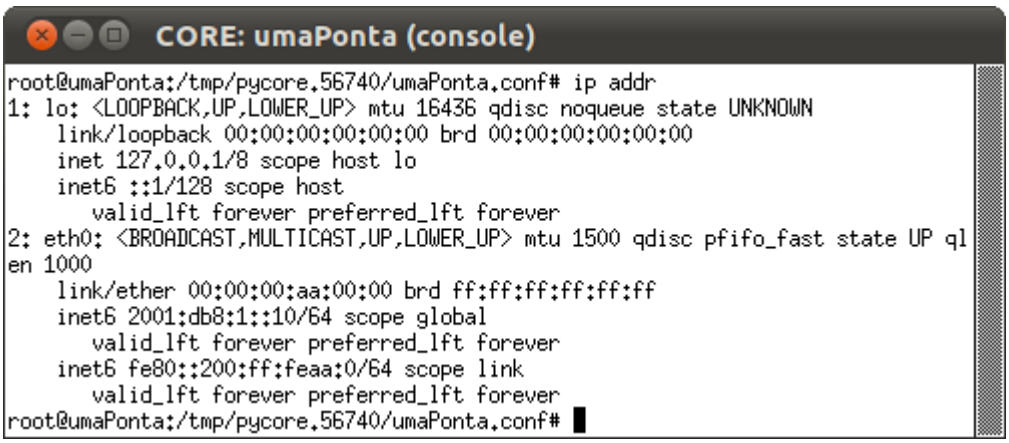
3. Verifique a configuração dos nós da topologia.
   a. Inicie a simulação realizando um dos seguintes passos:
      i. aperte o botão  ;
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine a inicialização da simulação e abra o terminal da máquina umaPonta, com um duplo-clique.

   c. Verifique que sua configuração através do seguinte comando:

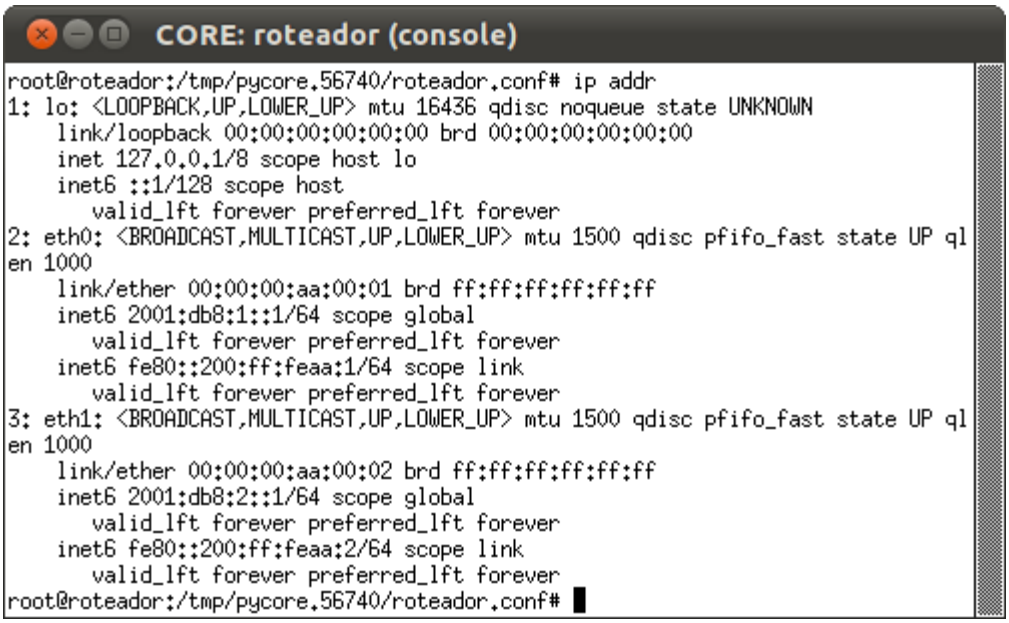
```
# ip addr
```

O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces, incluindo o MTU.

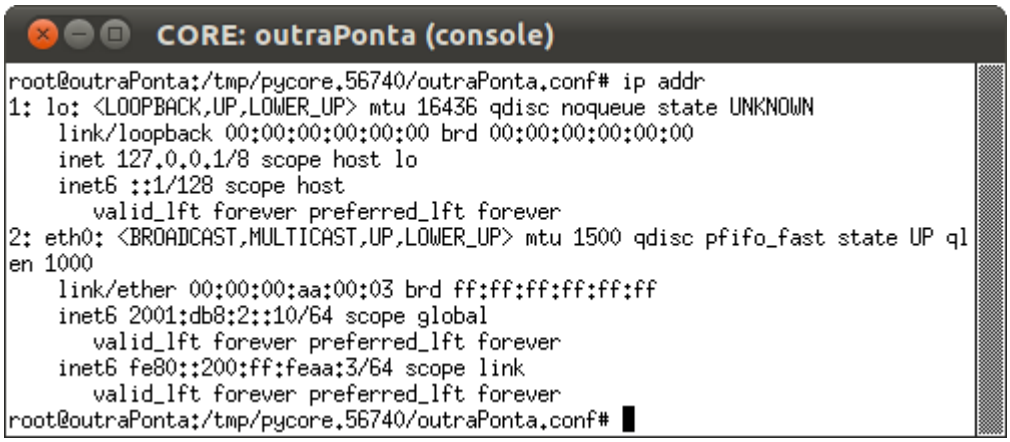
d. Verifique a configuração do roteador através do mesmo comando.
O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces, incluindo o MTU.

e. Verifique a configuração do servidor outraPonta através do mesmo comando.

O resultado deve ser:



***Obs**: A partir desse comando é possível observar os endereços das interfaces, incluindo o MTU.

4. Configure o MTU do enlace no qual o dispositivo outraPonta se encontra.
   a. Abra o terminal do roteador.

   b. Digite o seguinte comando para mudar o MTU da interface:

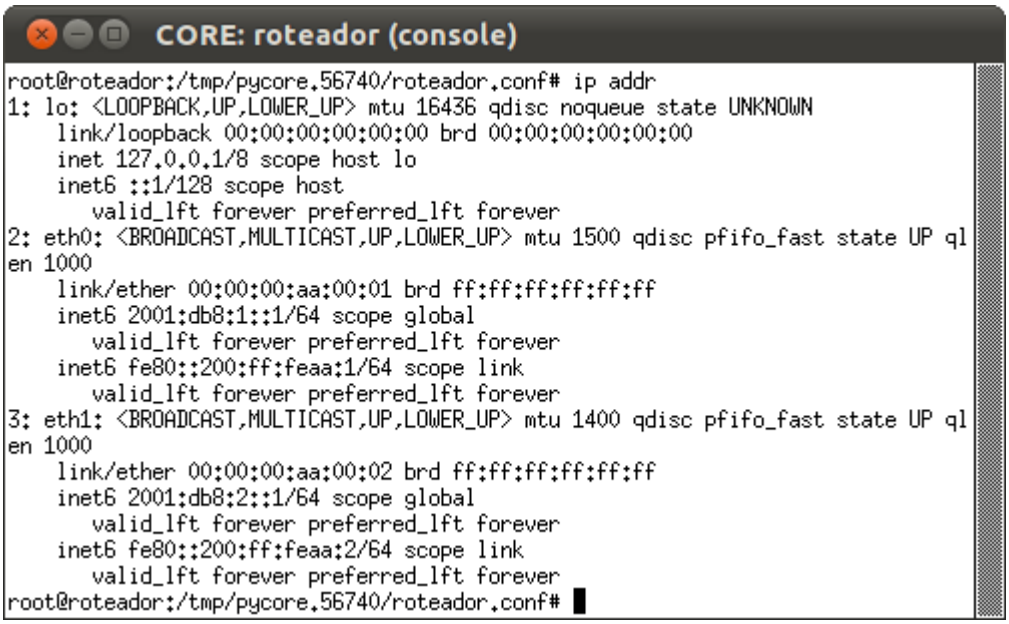
```
# ip link set eth1 mtu 1400
```

   O resultado deve ser:

   c. Observe a alteração através do comando:

```
# ip addr
```

O resultado deve ser:



***Obs**: Observe que houve uma alteração no tamanho do MTU na interface eth1.

d. Abra o terminal do dispositivo outraPonta.

e. Digite o seguinte comando para mudar o MTU da interface:

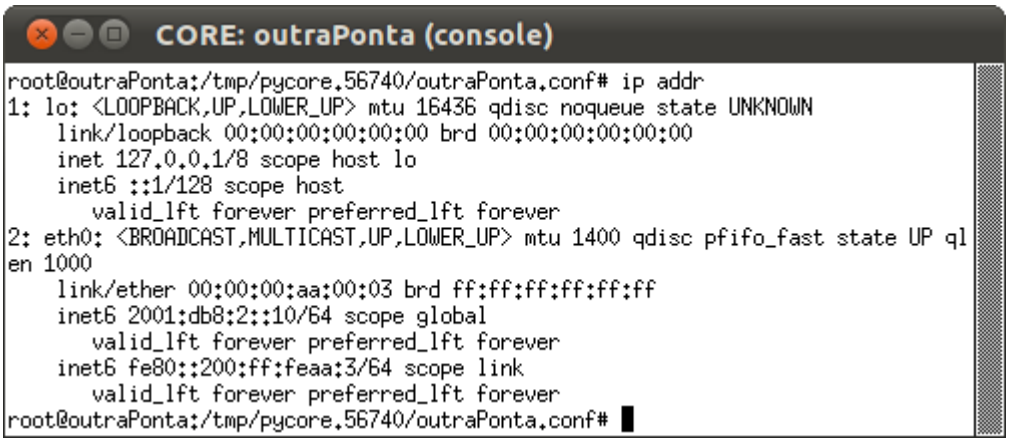
```
# ip link set eth0 mtu 1400
```

O resultado deve ser:

f. Observe a alteração através do comando:

```
# ip addr
```

O resultado deve ser:



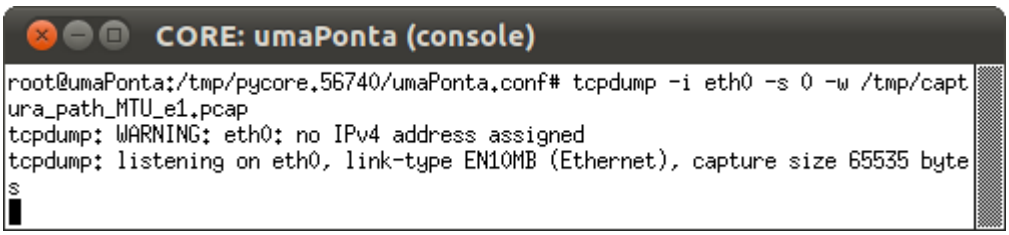
***Obs**: Note que houve uma alteração no tamanho do MTU na interface eth0.

5. Verifique a conectividade ipv6 entre as pontas com tamanho de pacote superior ao MTU configurado.
   a. Abra o terminal do dispositivo umaPonta;

   b. Utilize o seguinte comando para iniciar a captura de pacotes:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_path_MTU_e1.pcap
```

   O resultado deve ser:



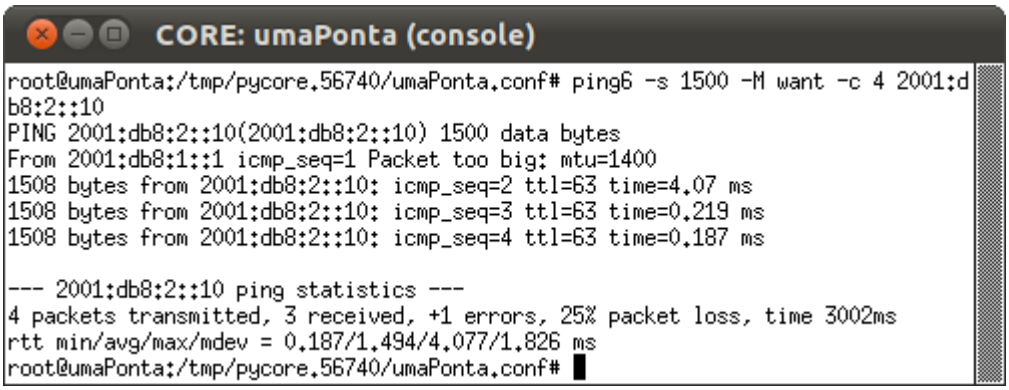
   ***Obs**: Não feche esse terminal até o final do experimento, uma vez que, isso ocasionará no término da execução do comando “tcpdump” e prejudicará o andamento da experiência.

   c. Abra um outro terminal do dispositivo umaPonta;

   d. O utilize o seguinte comando para testar a conectividade ipv6:

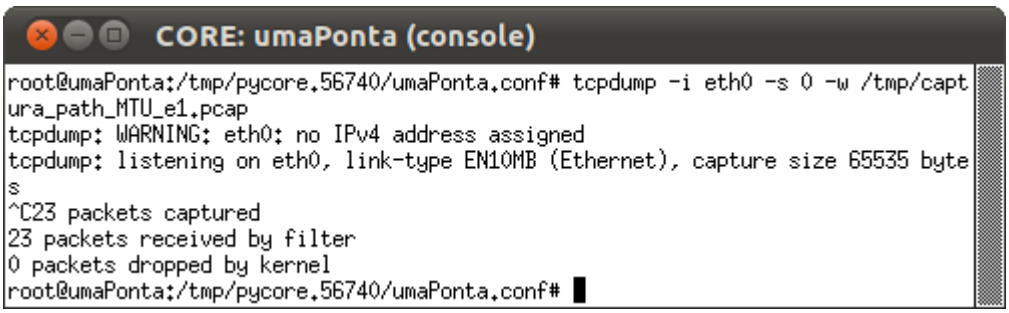
```
# ping6 -s 1500 -M want -c 4 2001:db8:2::10
```

O resultado deve ser:



***Obs**: Note que esse comando configura os pacotes enviados para conterem 1500 bytes de tamanho(com a opção "-s 1500") e a interface para permitir a fragmentação de pacotes (com a opção "-M want"). O resultado mostra que a partir do primeiro pacote descartado o PMTU é configurado corretamente para o limite de 1400 bytes e os pacotes passam a transitar corretamente pela rede.

e. No primeiro terminal da máquina umaPonta, encerre a captura de pacotes através da sequência Ctrl+C.
O resultado deve ser:

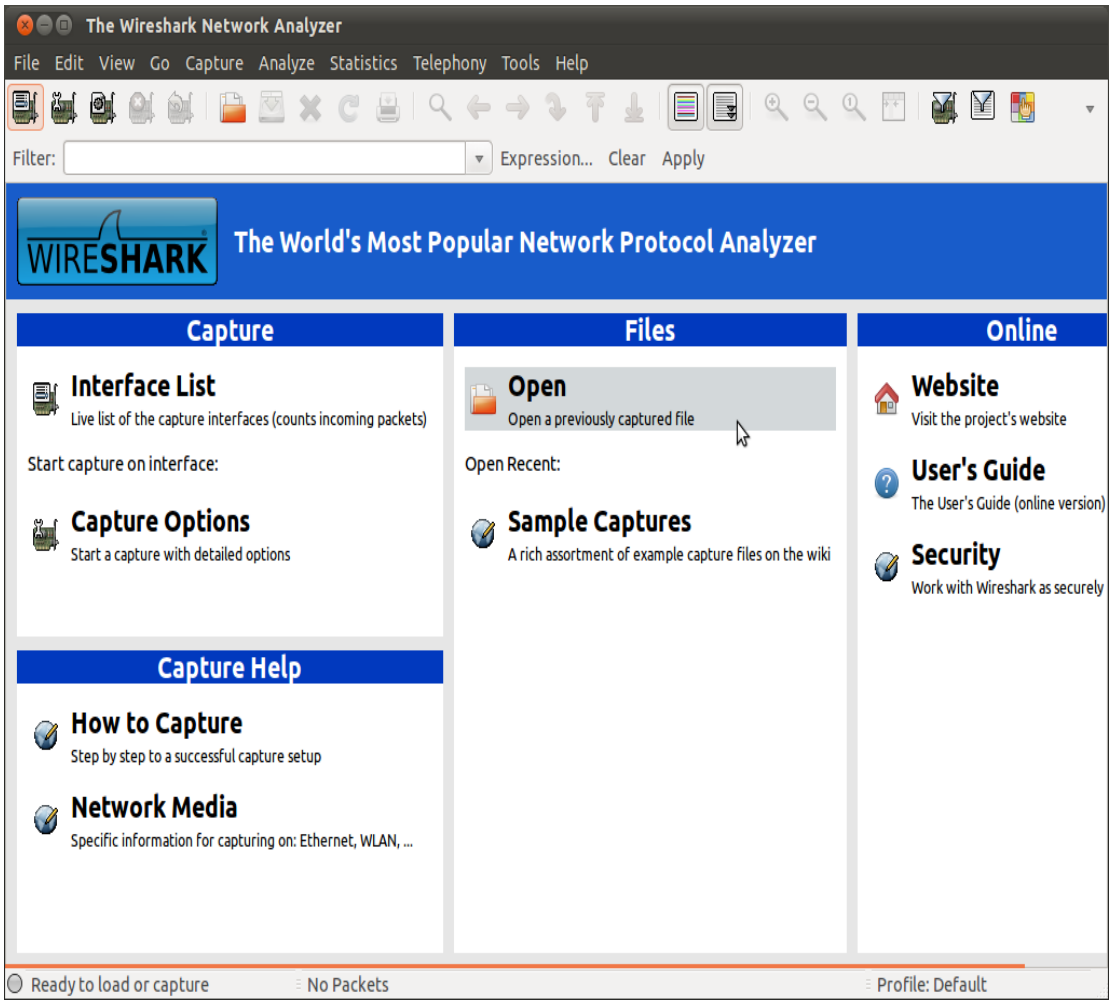


***Obs**: A quantidade de pacotes pode variar de acordo com o tempo esperado para dar o comando Ctrl+C.

6. Encerre a simulação com uma das seguintes ações:
   a. aperte o botão  ;
   b. utilize o menu Experiment > Stop.

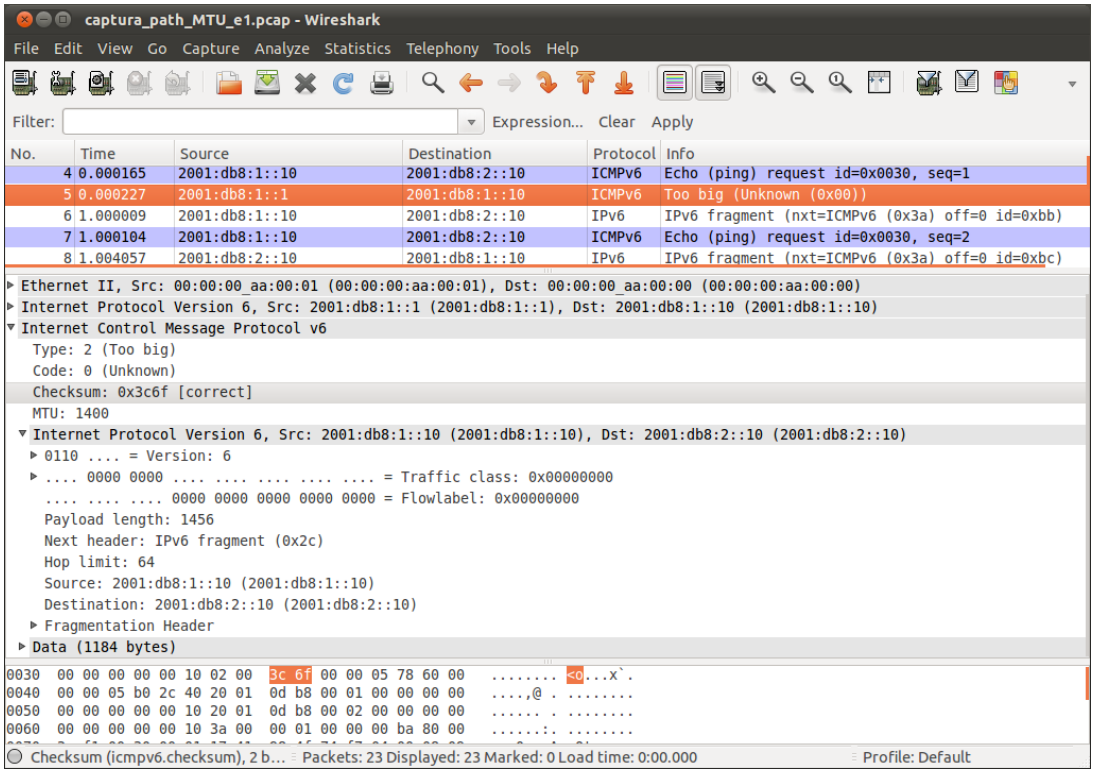
7. A verificação dos pacotes capturados será realizada através do programa Wireshark. Para iniciá-lo execute o seguinte comando em um terminal da máquina virtual:

```
$ wireshark
```



a. Abra o arquivo **/tmp/captura_path_MTU_e1.pcap** com o menu File>Open;
b. Procure pelo pacote icmpv6 "*Too big*". Analise-o e confirme que ele contém o valor do MTU que o roteador aceita trafegar em seu link para assim informar o dispositivo como deve ser feito a fragmentação.

Too Big:



***Obs**: o filtro icmpv6 pode ser usado para ajudar a filtrar as mensagens.

A partir dessa mensagem é possivel observar que o pacote não pode ser enviado em seu tamanho normal e necessita de fragmentação. Logo essa mensagem informa a origem da comunicação o tamanho limite que precisa ser utilizado para ser transmitido até seu destino.

IPv6 - Serviços IPv6 - DNS

Objetivo

O principal objetivo desse laboratório é apresentar o funcionamento do serviço de DNS (*Domain Name System*) em uma rede IPv6 utilizando o *software* BIND. Para isso, o laboratório será dividido em dois exercícios. O primeiro apresentará dois aspectos: a capacidade dos servidores DNS em armazenar tanto registros do tipo A, para endereços IPv4, quanto registros AAAA (quad-A), para endereços IPv6; e o fato das respostas às consultas DNS serem independentes do protocolo de rede utilizado, ou seja, um servidor é capaz responder tanto consultas AAAA quanto A mesmo que possua conexão apenas IPv4 ou apenas IPv6.

No segundo experimento, será trabalhada a configuração de um servidor DNS autoritativo para responder a requisições por registros do tipo AAAA e resolução de endereçamento reverso IPv6, destacando alguns pontos relacionados a utilização deste serviço em uma rede em Pilha Dupla, ou seja, com conectividade IPv4 e IPv6.

Para a realização destes exercícios serão utilizadas as topologias descritas nos arquivos **servicos-dns1.imn** e **servicos-dns2.imn**.

Introdução Teórica

O protocolo *Domain Name System* (DNS) é uma imensa base de dados distribuída em uma estrutura hierárquica, utilizada para a tradução de nomes de domínios em endereços IP e vice-versa.

Os dados associados aos nomes de domínio estão contidos em *Resource Records* ou RRs (Registro de Recursos). Atualmente existe uma grande variedade de tipos de RRs, sendo os mais comuns:

- **SOA** - Indica onde começa a autoridade sobre uma zona;
- **NS** - Indica um servidor de nomes para uma zona;
- **A** - Mapeamento de nome a endereço (IPv4);
- **AAAA** - Mapeamento de nome a endereço (IPv6);
- **MX** - Indica um *mail exchanger* para um nome (servidor de email);
- **CNAME** - Mapeia um nome alternativo (apelido);
- **PTR** - Mapeamento de endereço a nome.

O funcionamento do serviço de DNS baseia-se em uma arquitetura cliente/servidor, onde o cliente realiza requisições por RRs aos Servidores Recursivos. Ao receber requisições, os Servidores Recursivos as encaminham para Servidores Autoritativos e conforme a

resposta recebida, continuam a encaminhar as requisições para outros Servidores Autoritativos até obterem uma resposta satisfatória. Dentro da estrutura hierárquica do DNS, os Servidores Autoritativos respondem as requisições sobre as zonas ou domínios pelos quais possuem autoridade ou uma referência caso conheçam o caminho para a resposta, ou uma negação caso não conheçam.

Para que o DNS trabalhe com a versão 6 do protocolo Internet, algumas mudanças foram definidas na **RFC 3596**.

- Um novo tipo de RR foi criado para armazenar os endereços IPv6 de 128 bits, o AAAA ou quad-A. Sua função é a de traduzir nomes para endereços IPv6, de forma equivalente a do registro do tipo A no IPv4. Caso um dispositivo possua mais de um endereço IPv6, ele deverá ter um registro quad-A para cada. Os registros são representados como se segue:

Exemplo:	www.ipv6.br.	IN	A	200.160.4.22
	www.ipv6.br.	IN	AAAA	2001:12ff:0:4::22

- Para resolução de reverso, foi adicionado ao registro PTR o domínio ip6.arpa, responsável por traduzir endereços IPv6 em nomes. Em sua representação, o endereço é escrito com o bit menos significativo colocado mais a esquerda, como é possível observar no exemplo a seguir:
  Exemplo:

  22.4.160.200.in-addr.arpa PTR www.ipv6.br.
  2.2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.4.0.0.0.0.0.0.0.f.f.2.1.1.0.0.2.ip6.arpa PTR www.ipv6.br.

Todos os outros tipos de registro DNS não sofreram alterações em sua forma de configuração, apenas foram adaptados para suportar o novo tamanho dos endereços.

Roteiro Experimental

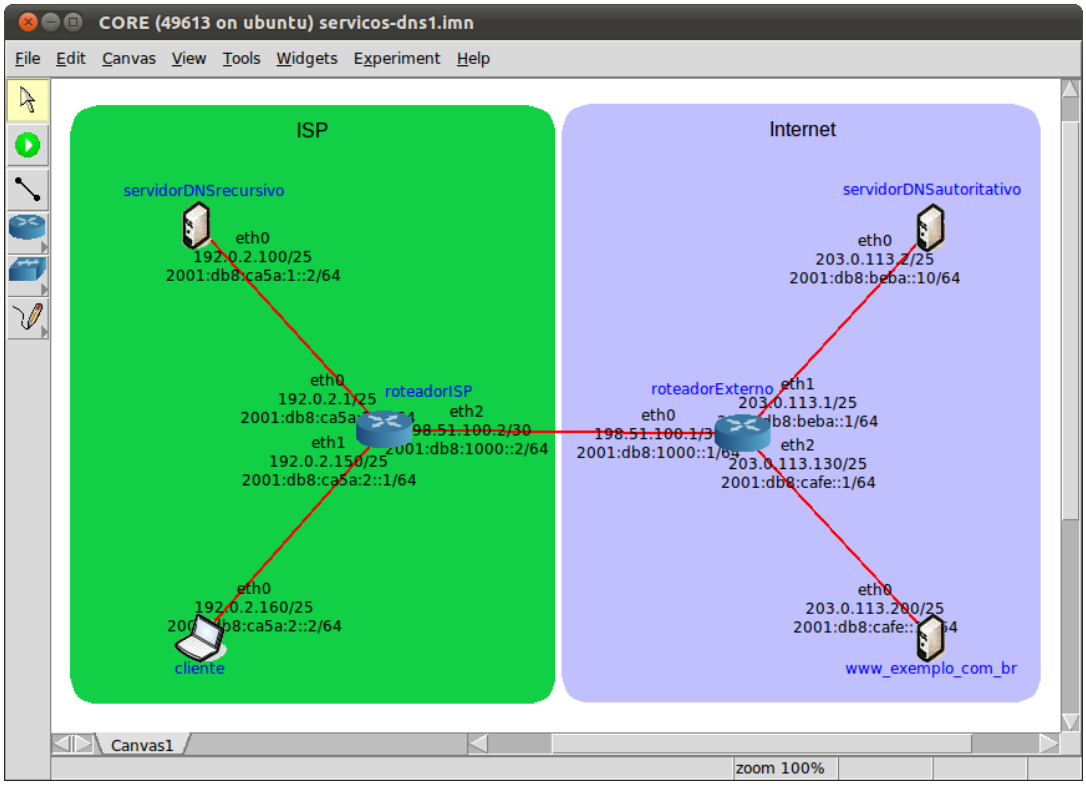
Experiência 1 - Consultas DNS

1. Caso esteja utilizando a Máquina Virtual fornecida pelo NIC.br pule para o passo 2. Caso não esteja utilizando a Máquina Virtual fornecida pelo NIC.br instale o programa BIND, que é uma implementação do protocolo DNS, com versão superior a 9.8. Na máquina virtual, utilize um Terminal para rodar os comandos:

```
$ wget ftp://ftp.isc.org/isc/bind9/9.8.1-P1/bind-9.8.1-P1.tar.gz
$ tar xvzf bind-9.8.1-P1.tar.gz
$ cd xvzf bind-9.8.1-P1
$ ./configure
$ make
$ sudo make install
```

Para verificar qual a versão mais atual do BIND acesse www.isc.org/products/BIND.

2. Agora, Inicie o CORE e abra o arquivo "**servicos-dns1.imn**" localizado no diretório /home/core/Desktop/servicos/DNS, da máquina virtual do NIC.br. A seguinte topologia deve aparecer:



Nesta topologia temos, localizados na Internet, a representação de um servidor 'www_exemplo_com_br' que responde pelo nome de domínio www.exemplo.com.br e um servidor DNS, com nome 'servidorDNSautoritativo', que possui autoridade sobre este domínio. Temos também, localizados no ISP, um servidor DNS recursivo, 'servidorDNSrecursivo', que recebe requisições de resolução de nomes, feitas pela máquina 'cliente', e as encaminha para o 'servidorDNSautoritativo'.

3. Verifique a configuração dos nós da topologia:

    a. Inicie a simulação realizando um dos seguintes passos:

        i. clique no botão  ; ou
        ii. utilize o menu Experiment > Start.

4. Neste laboratório, as configurações serão realizadas apenas nos equipamentos do ISP e elas serão iniciadas pelo 'servidorDNSrecursivo'. Crie o arquivo "`named.conf`" no diretório "`/etc/named/`":

    a. Acesse o terminal da máquina 'servidorDNSrecursivo' (duplo clique sobre ela) e digite os seguintes comando:

```
# nano /etc/named/named.conf
```

b. Adicione as seguintes linhas no arquivo criado:

```
options {
  allow-query {192.0.2.0/24;};
  forward only;
  forwarders {203.0.113.2;};
};
```

Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

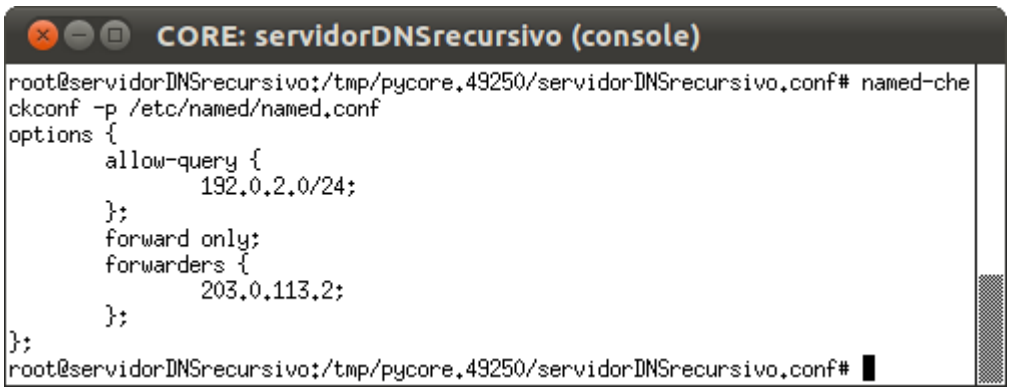
Caso esteja utilizando a Máquina Virtual fornecida pelo NIC.br, o conteúdo deste arquivo pode ser visto em :
/home/core/Desktop/servicos/DNS/named.conf1.

Este arquivo contém as configurações mínimas para o funcionamento de um servidor DNS recursivo. A opção "allow-query" indica a faixa de endereços IP que têm permissão para realizar consultas ao servidor; a opção "forward only" indica que este servidor não tem autoridade sobre nenhum domínio, ele apenas encaminha requisições para outros servidores, funcionando como um cache DNS; por fim, na opção "forwarders" é definida uma lista de endereços IP de servidores DNS para os quais as requisições devem ser encaminhadas. Observe que este servidor recursivo só recebe e encaminha requisições via IPv4.

5. O BIND possui ferramentas que verificam a sintaxe dos arquivos de configuração que podem auxiliar na resolução de problemas relacionados ao funcionamento do DNS. Deste modo, antes de iniciar o processo do BIND, verifique se o arquivo "`named.conf`" foi gerado corretamente:

```
# named-checkconf -p /etc/named/named.conf
```

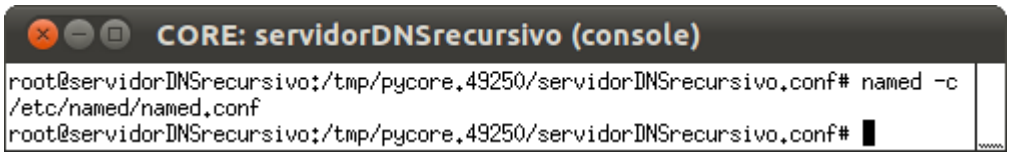
O resultado deve ser:



6. Caso o passo anterior não tenha apresentado erros de execução, inicie o processo do BIND através do seguinte comando:

```
# named -c /etc/named/named.conf
```

O resultado deve ser:



7. Abra o terminal da máquina 'cliente' e configure o arquivo "`resolv.conf`", localizado no diretório "`/etc/`", de forma que a máquina começe a utilizar o 'servidorDNSrecursivo' para a realização de consultas DNS:

    a. Acesse o terminal da máquina 'cliente' (duplo clique sobre ela) e digite o seguinte comando:

    ```
    # nano /etc/resolv.conf
    ```

    b. Adicione o seguinte conteúdo ao arquivo resolv.conf:

    ```
    nameserver 192.0.2.100
    ```

    Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

    Note que essa regra configura apenas o endereço IPv4 do 'servidorDNSrecursivo'.

8. Ainda na máquina 'cliente', realize uma consulta DNS ao registro AAAA do domínio www.exemplo.com.br, ou seja, ao endereço IPv6 associado a este domínio.

a. Esta consulta pode ser feita com a utilização do comando host:

```
# host -t AAAA www.exemplo.com.br
```

O resultado deve ser:

Mesmo com o 'servidorDNSrecursivo' configurado para ser acessado apenas via IPv4, ele é capaz de responder à requisições de endereços IPv6. Isso demonstra que as informações armazenadas no banco de dados do servidor DNS são independentes da versão do protocolo de rede utilizado na comunicação.

b. Faça agora uma consulta sem o parâmetro "-t AAAA":

```
#host www.exemplo.com.br
```

O resultado deve ser:

Pode-se observar que a resposta contém tanto o endereço IPv4 quanto o endereço IPv6 associados ao domínio pesquisado.

9. O próximo passo é habilitar o servidor DNS recursivo para aceitar requisições via IPv6.

a. Para isso, abra um terminal da máquina 'servidorDNSrecursivo' (duplo clique sobre ela) e digite os seguintes comandos:

```
# nano /etc/named/named.conf
```

b. Adicione a linha "`listen-on-v6 { any; };`" e adicione na opção "`allow-query`" a rede IPv6 do ISP. O arquivo "`named.conf`" deverá ficar da seguinte forma:

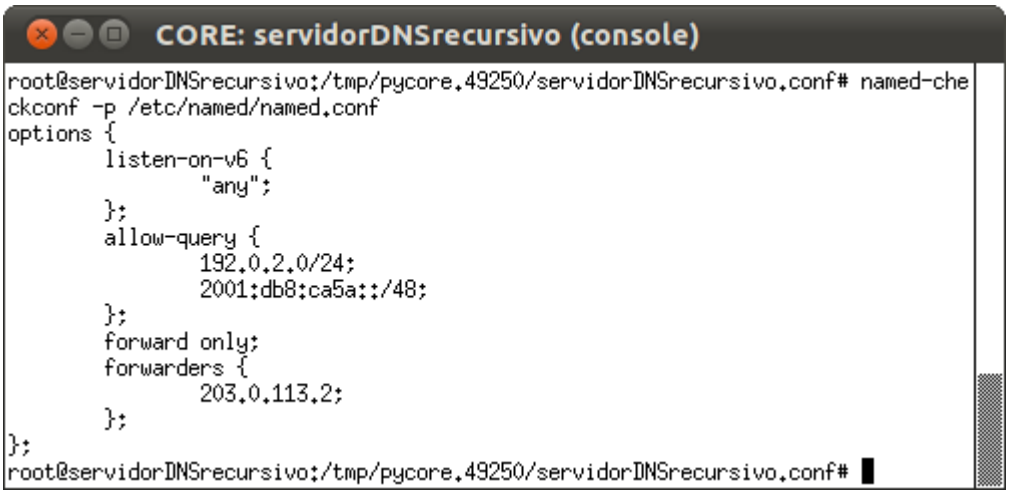
```
options {
allow-query {192.0.2.0/24; 2001:db8:ca5a::/48;};
  forward only;
  forwarders {203.0.113.2;};
  listen-on-v6 { any; };
};
```

Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

c. Antes de iniciar o processo do BIND, verifique se o arquivo "`named.conf`" foi gerado corretamente. Isso pode ser feito através do seguinte comendo:

```
# named-checkconf -p /etc/named/named.conf
```

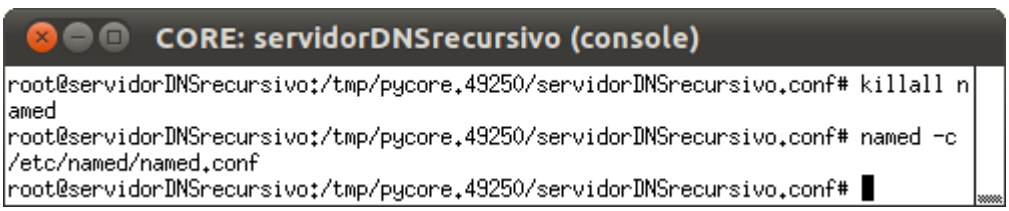
O resultado deve ser:



d. Caso o passo anterior não tenha apresentado erros de execução, reinicie o processo do BIND para que a alteração seja aplicada. Para isso, digite os seguintes comandos:

```
# killall named
# named -c /etc/named/named.conf
```

O resultado deve ser:



10. Abra o terminal da máquina 'cliente' e edite o arquivo "`resolv.conf`", localizado no diretório "`/etc/`", de forma que a máquina começe a utilizar também o endereço

IPv6 do 'servidorDNSrecursivo' para a realização de consultas DNS:

a. Acesse o terminal da máquina 'cliente' (duplo clique sobre ela) e digite o seguinte comando:

```
# nano /etc/resolv.conf
```

b. Adicione ao conteúdo existente no arquivo "`resolv.conf`" a seguinte linha:

```
nameserver 2001:db8:ca5a:1::2
```

Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

11. Ainda na máquina 'cliente', realize uma consulta DNS ao registro A do domínio www.exemplo.com.br, ou seja, ao endereço IPv4 associado a este domínio, porém force que a consulta seja feita via IPv6

a. Utilize o comando *host* com a opção '-6':

```
# host -t A -6 www.exemplo.com.br
```

O resultado deve ser:

Assim como ocorreu nos teste do item 8, mesmo com a realização da requisição via IPv6, o servidor DNS recursivo foi capaz de responder por endereços IPv4.

b. É possível realizar uma série de testes para verificar o funcionamento do servidor DNS. Algumas opções podem ser feitas com a utilização dos comandos *dig*, *ping* e *ping6* para verificar a conectividade, resolução de endereço reverso, etc. Alguns exemplos são:

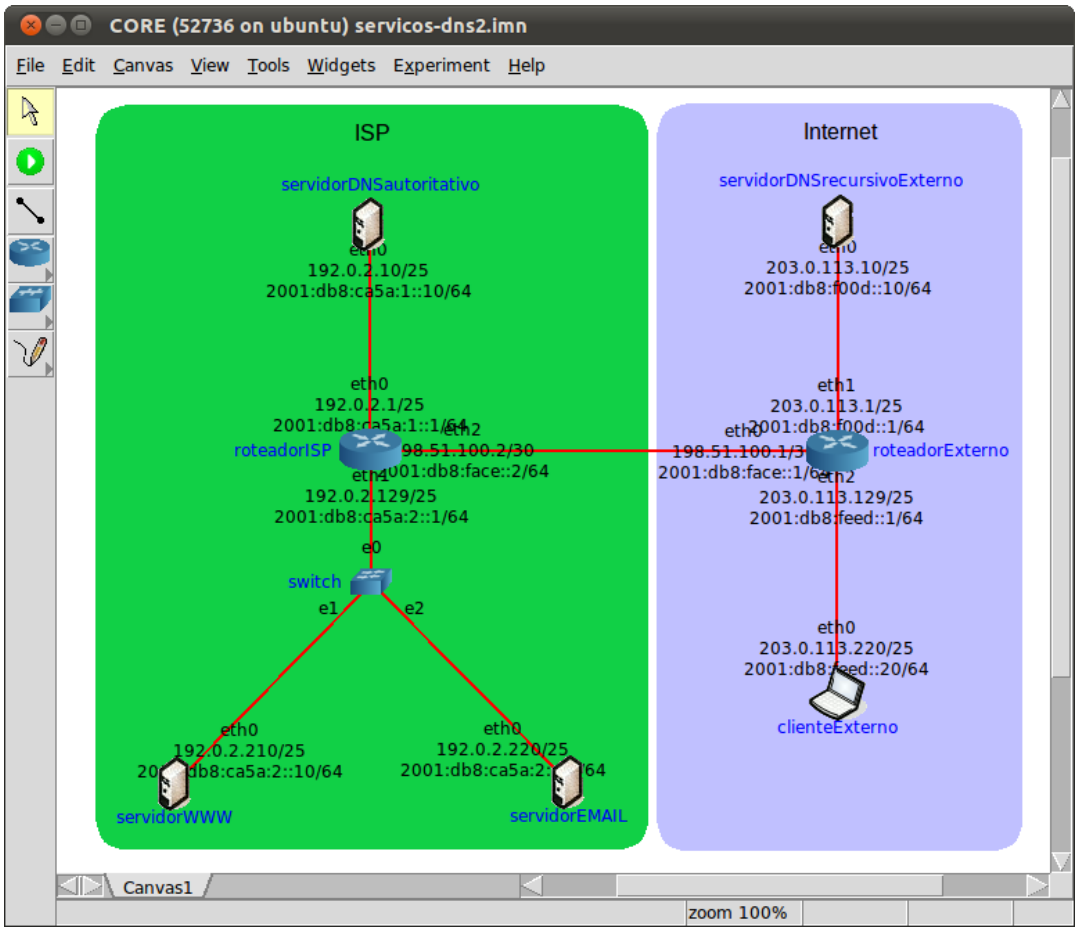
```
# host 2001:db8:cafe::10
# ping www.exemplo.com.br
# ping6 www.exemplo.com.br
# nslookup -type=ANY exemplo.com.br
```

12. Pare a simulação do CORE :
    a. aperte o botão 
    b. ou utilize o menu Experiment > Stop.

Experiência 2 - DNS: Configurando um Servidor Autoritativo

1. Para a realização desta segunda experiência, também é necessária a instalação do BIND, como descrita no item 1 do exercício anterior.I

2. Agora, Inicie o CORE e abra o arquivo "**servicos-dns2.imn**" localizado no diretório /home/core/Desktop/servicos/DNS, da máquina virtual do NIC.br. A seguinte topologia deve aparecer:



Nesta topologia temos, localizados na Internet, a representação de um servidor DNS recursivo (servidorDNSrecursivoExterno), responsável por receber requisições de nomes de seus clientes e reencaminhá-las para servidores autoritativos, e de um cliente (clienteExterno) que será utilizado para realizar as requisições DNS, testando assim as configurações aplicadas. No ISP, encontram-se dois servidores representando os serviços de e-mail e web, e um servidor DNS autoritativo (servidorDNSautoritativo) responsável por responder requisições ao domínio exemplo.psi.br.

3. Verifique a configuração dos nós da topologia.

    a. Inicie a simulação realizando um dos seguintes passos:

i. clique no botão 
ii. utilize o menu Experiment > Start.

4. Neste laboratório, as configurações serão realizadas apenas nos equipamentos do ISP. Inicialmente, analise os arquivos de configuração do BIND localizados no servidor DNS autoritativo do ISP. Este servidor já está configurado para responder requisições por registros tipo A e de endereçamento reverso IPv4.

   a. Para isso, acesse o terminal da máquina 'servidorDNSautoritativo' (duplo clique sobre ela) e visualize o arquivo named.conf localizado no diretório /etc/named/ digitando o seguinte comando:

```
# cat /etc/named/named.conf
```

   O arquivo named.conf deverá conter as linhas:

```
options {
 directory "/etc/named/";
 listen-on { any; };
 allow-query { any; };
 recursion no;
};

zone "." {
 type hint;
 file "named.root";
};

zone "exemplo.psi.br" {
 type master;
 file "exemplo.psi.br.zone";
};

zone "2.0.192.in-addr.arpa" {
 type master;
 file "192-0-2.db";
};
```

   Este arquivo contem as configurações básicas necessárias para o funcionamento do servidor DSN autoritativo. No primeiro bloco de comandos, o *options*, temos as especificações que controlam o comportamento global do servidor. Neste exemplo, são listadas as seguintes opções: *directory*, que indica em qual diretório encontram-se os arquivos utilizados pelo BIND; *listen-on*, lista os endereços IPv4 e Portas habilitadas para responderem as requisições DNS, neste exemplo está a configuração padrão, responder em qualquer interface e na porta 53; *allow-query*, lista de quais endereços IP têm permissão para realizar requisições, neste exemplo são aceitas requisições vindas de qualquer IP; e *recursion*, indica se o servidor é capaz (yes) ou não

(no) de reencaminhar requisições a outros servidores autoritativos.

Abaixo das opções, temos a lista de arquivos com as zonas conhecidas pelo servidor e dos arquivos com as respectivas informações. A zona "." representa a zona raiz da Internet e o arquivo named.root contem os endereços dos servidores raiz DNS (a.root-servers.net - m.root-servers.net) onde o BIND irá buscar as informações sobre os domínios de primeiro nível (por exemplo: .br, .com, .net, .org....). A zona "exemplo.psi.br" indica o domínio sobre o qual o servidor DNS tem autoridade para responder requisições (type master) e a "2.0.192.in-addr.arpa" indica qual a zona de endereçamento reverso IPv4 o servidor responde. Agora, analise cada um dos arquivos relacionados a estas duas zonas.

b. Primeiro analise o arquivo exemplo.psi.br.zone localizado no diretório /etc/named/, para isso, digite o seguinte comando:

```
# cat /etc/named/exemplo.psi.br.zone
```

O arquivo exemplo.psi.br.zone deverá conter as linhas:

```
;; Recomenda-se utilizar o valor do TTL igual a 1 dia (86400 segundos),
;; mas para efeito de demostração utilizaremos um valor reduzido
$TTL 1s
exemplo.psi.br.    IN    SOA ns.exemplo.psi.br.  root.exemplo.psi.br. (
                             15 ; serial
                             28800 ; refresh
                             7200 ; retry
                             604800 ; expire
                             1s ; ttl
                             )

;;Servidor que responde pelo dominio
                   IN    NS        ns.exemplo.psi.br.
ns                 IN    A         192.0.2.10

;;Servidor de e-mail
exemplo.psi.br.    IN    MX    10  mail.exemplo.psi.br.
mail               IN    A         192.0.2.220

;;Politica de SPF
exemplo.psi.br.    IN    TXT       "v=spf1 mx ip4:192.0.2.0/24 -all"
exemplo.psi.br.    IN    SPF       "v=spf1 mx ip4:192.0.2.0/24 -all"

;;Servidor Web
exemplo.psi.br.    IN    A         192.0.2.210
www                IN    CNAME     exemplo.psi.br.
```

Este arquivo apresenta os registros e diretivas relacionados a zona exemplo.psi.br.
A diretiva $TTL (Time To Live) indica o tempo que os registros devem

permanecer no cache sem que sejam atualizados, podendo ser expresso em segundos, minutos, horas, dias ou semanas (em nosso exemplo está setado para um segundo, mas apenas para que os exercícios possam ser demonstrados. A recomendação é que seja pelo menos um dia).

c. Agora, analise o arquivo 192-0-2.db localizado no diretório /etc/named/. Para isso, digite o seguinte comando:

```
# cat /etc/named/192-0-2.db
```

O arquivo 192-0-2.db deverá conter as linhas:

```
$TTL 86400
2.0.192.in-addr.arpa. IN SOA ns.exemplo.psi.br.  root.exemplo.psi.br. (
                                15 ; serial
                                28800 ; refresh
                                7200 ; retry
                                604800 ; expire
                                86400 ; ttl
                                )

;; Servidor DNS que responde por esta zona reverso
                        IN   NS   ns.exemplo.psi.br.

;; Enderecos reversos
10                      IN   PTR   ns.exemplo.psi.br.
210                     IN   PTR   www.exemplo.psi.br.
220                     IN   PTR   mail.exemplo.psi.br.
```

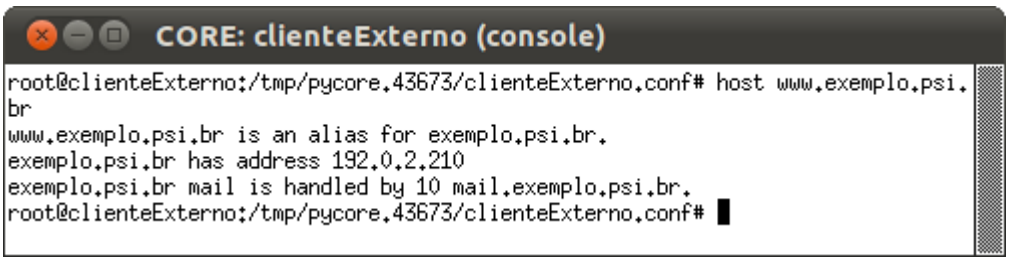
Este arquivo apresenta os registros e diretivas relacionados a zona de endereçamento reversa IPv4.

5. Faça agora algumas consultas DNS para testar as configurações do servidor DNS autoritativo da rede ISP.

   a. Para isso, acesse a máquina ‘clienteExterno’ (duplo clique sobre ela) e digite o seguinte comando:

```
# host www.exemplo.psi.br
```

O resultado deve ser:



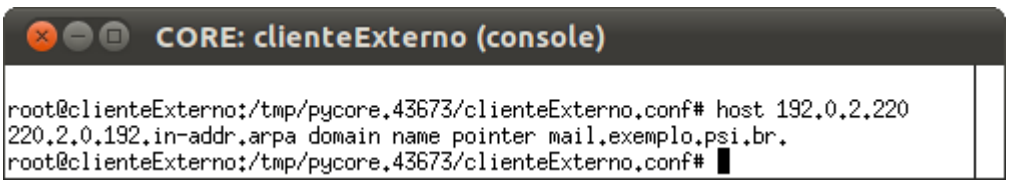
A partir da resposta obtida pode-se observar que:

- o nome www.exemplo.psi.br é um alias (apelido) para o domínio exemplo.psi.br;
- para o nome consultado há um endereço IPv4 associado, o 192.0.2.210;
- e que existe um servidor de e-mail associado ao domínio exemplo.psi.br, o mail.exemplo.psi.br.

b. Outra consulta que pode ser realizada é a de resolução de endereço reverso. Acesse a máquina 'clienteExterno' e digite o seguinte comando:

```
# host 192.0.2.220
```

O resultado deve ser:



O conteúdo das respostas é baseado nas informações contidas nos arquivos 192-0-2.db e exemplo.psi.br.zone, existentes no 'servidorDNSrecursivo' e analisadas nos itens 4b e 4c deste exercício. Note que, apesar dos servidores do ISP terem endereços IPv6 em suas interfaces de rede, nenhuma consulta DNS feita retornou um endereço IPv6 como resposta. Isto ocorreu porque não há essa informação cadastrada nos arquivos de zona do Servidor DNS Autoritativo do ISP.

6. Deste modo, o próximo passo é configurar o Servidor Autoritativo para que ele seja capaz tanto de receber requisições via IPv6 quanto responder endereços IPv6 às consultas realizadas.

a. Primeiro, acesse o terminal da máquina 'servidorDNSautoritativo' (duplo clique sobre ela) e edite o arquivo named.conf localizado no diretório /etc/named/ digitando o seguinte comando:

```
# nano /etc/named/named.conf
```

b. Adicione às opções a linha listen-on-v6 { any; };, habilitando o servidor a receber consultas via IPv6. O campo options do arquivo named.conf ficará desta forma:
ATENÇÃO: Apenas adicione a linha indicada. O restante do arquivo não deve ser alterado!

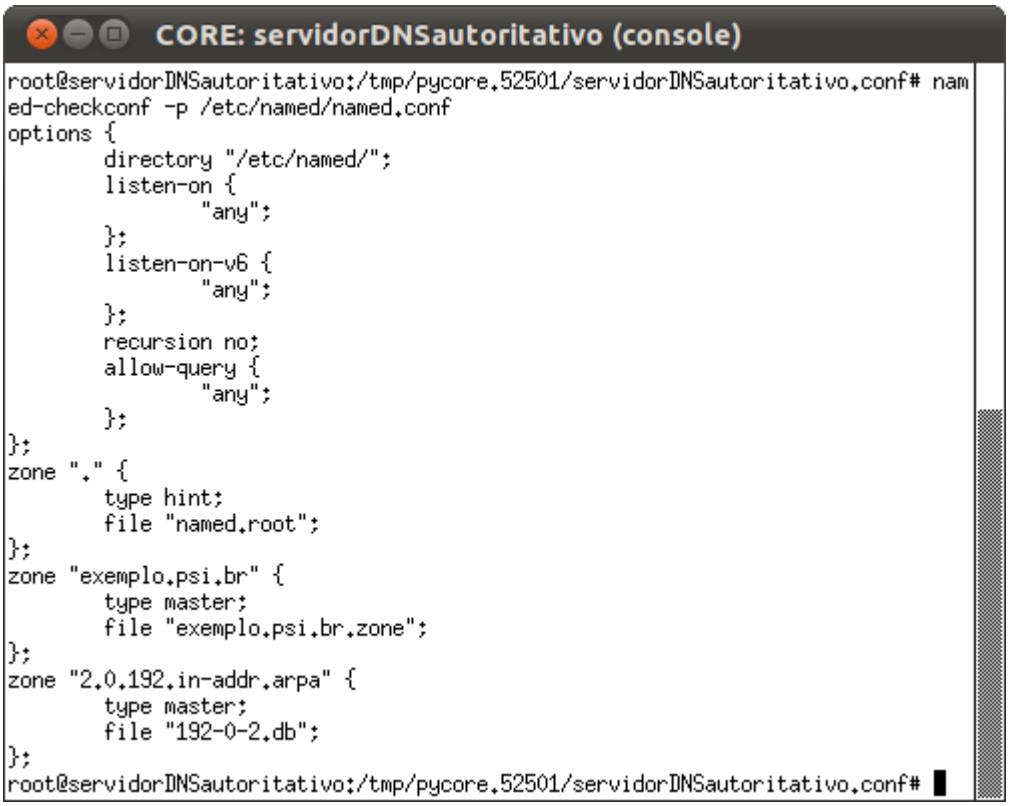
```
options {
 directory "/etc/named/";
 listen-on { any; };
 listen-on-v6 { any; };
 allow-query { any; };
 recursion no;
};
```

Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

c. Antes de reiniciar o processo do BIND, verifique se o arquivo "`named.conf`" foi gerado corretamente. Isso pode ser feito através do seguinte comendo:

```
# named-checkconf -p /etc/named/named.conf
```

O resultado deve ser:



d. Edite também o arquivo exemplo.psi.br.zone localizado no diretório /etc/named/, adicionando os endereços IPv6 aos nomes e registros já configurados e alterando os parâmetros das políticas de SPF. Para isso, digite o seguinte comando:

```
# nano /etc/named/exemplo.psi.br.zone
```

O arquivo exemplo.psi.br.zone deverá conter as linhas:

ATENÇÃO: As linhas que devem ser adicionadas ou alteradas estão destacadas abaixo!

```
;; Recomenda-se utilizar o valor do TTL igual a 1 dia (86400 segundos),
;; mas para efeito de demostração utilizaremos um valor reduzido
$TTL 1s
exemplo.psi.br.   IN   SOA ns.exemplo.psi.br.  root.exemplo.psi.br. (
                           15 ; serial
                           28800 ; refresh
                           7200 ; retry
                           604800 ; expire
                           1s ; ttl
                           )
;;Servidor que responde pelo dominio
                IN   NS       ns.exemplo.psi.br.
ns              IN   A        192.0.2.10
ns              IN   AAAA     2001:db8:ca5a:1::10
```

```
;;Servidor de e-mail
exemplo.psi.br. IN   MX   10  mail.exemplo.psi.br.
mail            IN   A        192.0.2.220
mail            IN   AAAA     2001:db8:ca5a:2::220

;;Politica de SPF
exemplo.psi.br. IN  TXT   "v=spf1 mx ptr ip4:192.0.2.0/24 \
ip6:2001:db8:ca5a::/48 -all"
exemplo.psi.br. IN  SPF   "v=spf1 mx ptr ip4:192.0.2.0/24 \
ip6:2001:db8:ca5a::/48 -all"

;;Servidor Web
exemplo.psi.br.   IN   A        192.0.2.210
exemplo.psi.br.   IN   AAAA     2001:db8:ca5a:2::210
www               IN   CNAME    exemplo.psi.br.
```

Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

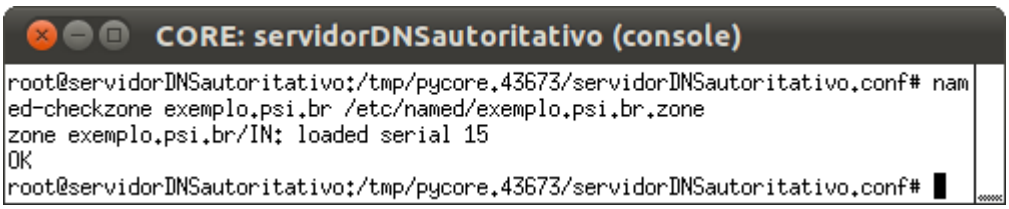
Caso esteja utilizando a Máquina Virtual fornecida pelo NIC.br, o conteúdo deste arquivo pode ser visto em : /home/core/Desktop/servicos/DNS/exemplo.psi.br.zone.

Observe que além de adicionar os registros AAAA aos nomes de domínios previamente cadastrados, também foram adicionados parâmetros às políticas de SPF.

e. Para verificar se não existem erros no arquivo de zona gerado, digite o seguinte comando no terminal do 'servidorDNSautoritativo':

```
# named-checkzone exemplo.psi.br /etc/named/exemplo.psi.br.zone
```

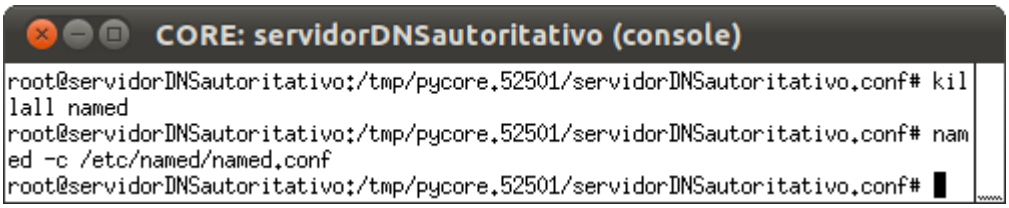
O resultado deve ser:



f. Caso os passos anteriores não tenham apresentado erros de execução, reinicie o processo do BIND para que as alterações sejam aplicadas. Para isso, digite os seguintes comandos:

```
# killall named
# named -c /etc/named/named.conf
```

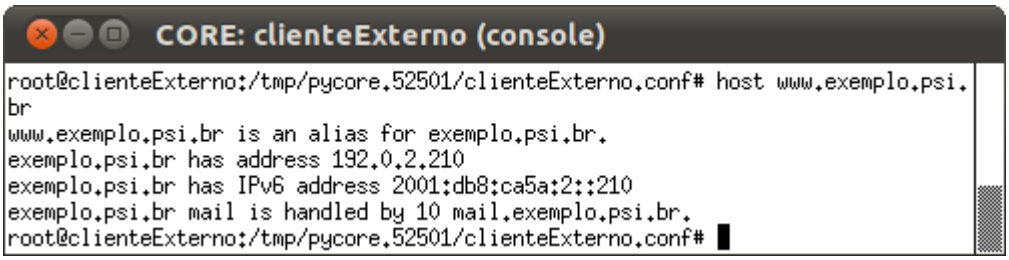
O resultado deve ser:



g. Para verificar se as alterações foram realizadas corretamente, acesse a máquina 'clienteExterno' (duplo clique sobre ela) e realize algumas requisições DNS. Para isso, utilize o comando host digitando o seguinte:

```
# host www.exemplo.psi.br
```

O resultado deve ser:

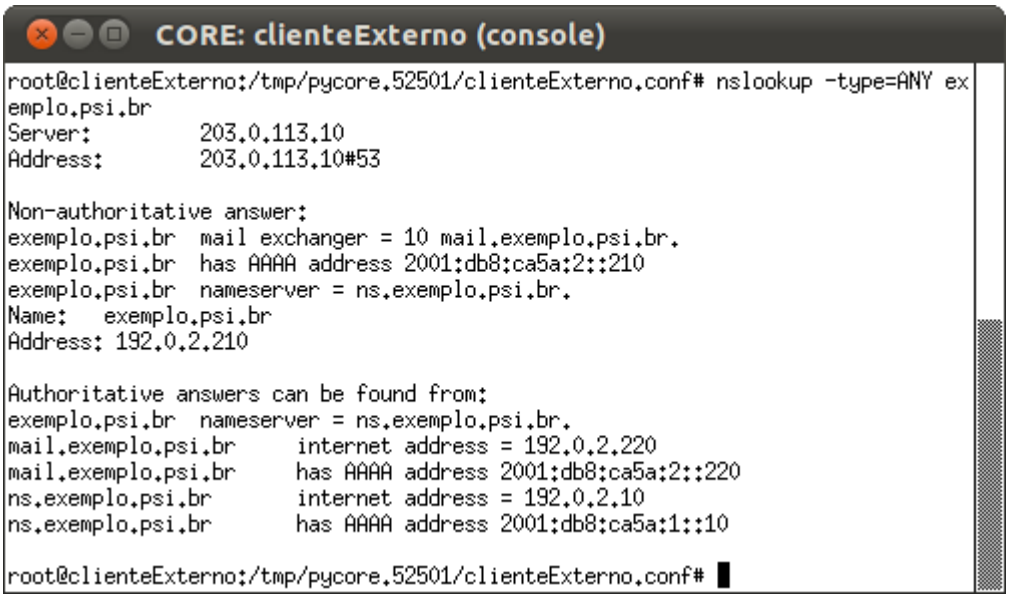


Além das informações obtidas no item 5a, também temos agora um endereço IPv6 associado ao nome www.exemplo.psi.br.

h. Repita os testes para os outros endereços e compare os resultados. Utilize, por exemplo, o comando nslookup para obter uma resposta mais completa digitando o seguinte:

```
# nslookup -type=ANY exemplo.psi.br
```

O resultado deve ser:



7. Para finalizar o exercício, configure o Servidor Autoritativo para responder a requisições de endereçamento reverso IPv6.

   a. Para isso, acesse a máquina 'servidorDNSautoritativo' (duplo clique sobre ela) e crie o arquivo 2001-db8-ca5a.db dentro do diretório /etc/named/, digitando o seguinte comando:

```
# nano /etc/named/2001-db8-ca5a.db
```

   Adicione as linhas abaixo ao arquivo 2001-db8-ca5a.db:

```
$TTL 86400
a.5.a.c.8.b.d.0.1.0.0.2.ip6.arpa.    IN   SOA   ns.exemplo.psi.br. \
root.exemplo.psi.br. (
                      15 ; serial
                      28800 ; refresh
                      7200 ; retry
                      604800 ; expire
                      86400 ; ttl
                      )

;; Servidor DNS que responde por esta zona reverso
                                        IN   NS   ns.exemplo.psi.br.

;; Enderecos reversos
$ORIGIN      a.5.a.c.8.b.d.0.1.0.0.2.ip6.arpa.
0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.1.0.0.0  IN  PTR  ns.exemplo.psi.br.
0.1.2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.0.0  IN  PTR  www.exemplo.psi.br.
0.2.2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.0.0  IN  PTR  mail.exemplo.psi.br.
```

   Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

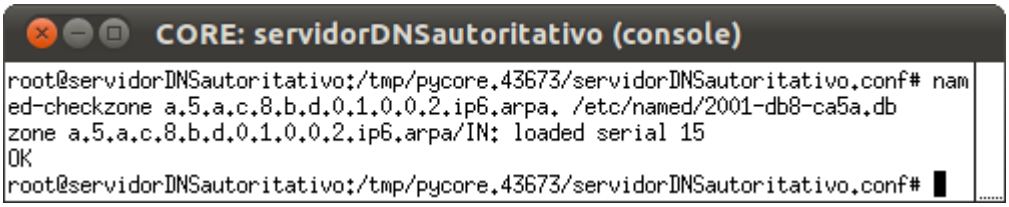
Caso esteja utilizando a Máquina Virtual fornecida pelo NIC.br, o conteúdo deste arquivo pode ser visto em : /home/core/Desktop/servicos/DNS/2001-db8-ca5a.db.

Este arquivo apresenta os registros e diretivas relacionados a zona de endereçamento reversa IPv6. Ele possui as mesmas funcionalidade e características do arquivo 192-0-2.db analisado no item 4c.

b. Para verificar se não existem erros no arquivo de zona gerado, digite o seguinte comando no terminal do 'servidorDNSautoritativo':

```
# named-checkzone a.5.a.c.8.b.d.0.1.0.0.2.ip6.arpa
/etc/named/2001-db8-ca5a.db
```

O resultado deve ser:



c. Agora, cadastre essa zona no arquivo named.conf, localizado no diretório /etc/named/. Para isso, digite o seguinte comando no terminal da máquina 'servidorDNSautoritativo':

```
# nano /etc/named/named.conf
```

d. Adicione ao final do arquivo as seguintes linha:

```
zone "a.5.a.c.8.b.d.0.1.0.0.2.ip6.arpa" {
  type master;
  file "2001-db8-ca5a.db";
};
```

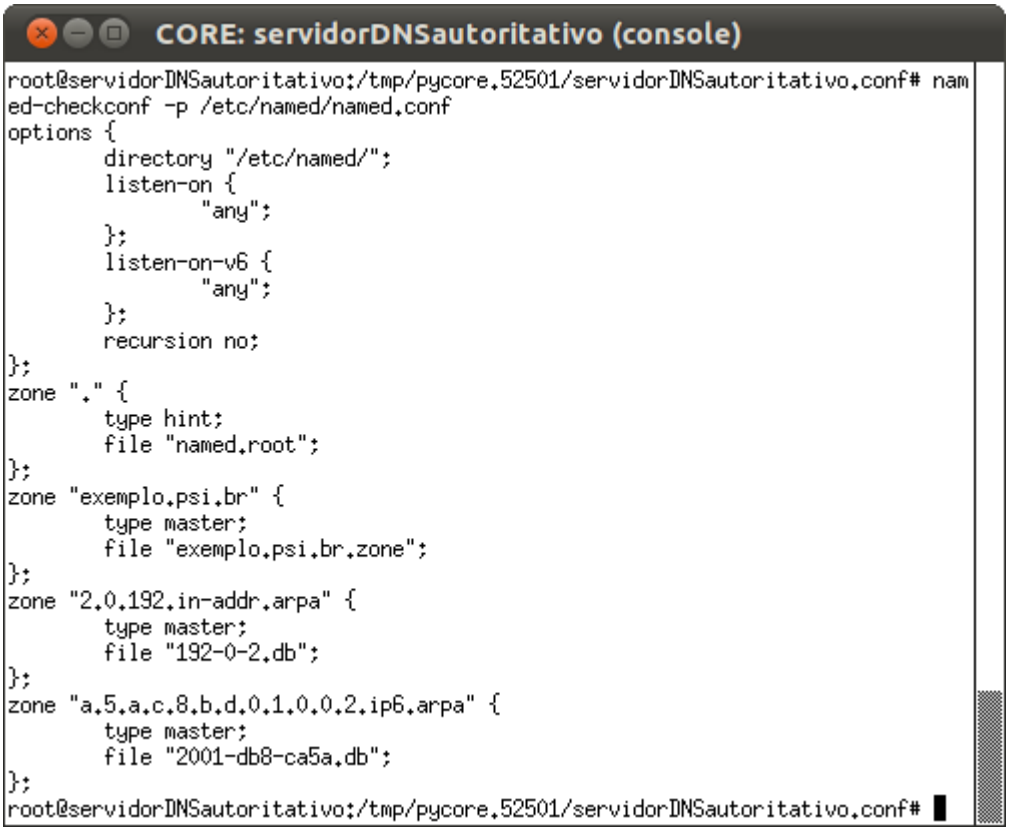
Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

Caso esteja utilizando a Máquina Virtual fornecida pelo NIC.br, o conteúdo deste arquivo pode ser visto em : /home/core/Desktop/servicos/DNS/named.conf2.

e. Antes de reiniciar o processo do BIND, verifique se o arquivo "`named.conf`" foi gerado corretamente. Isso pode ser feito através do seguinte comendo:

```
# named-checkconf -p /etc/named/named.conf
```

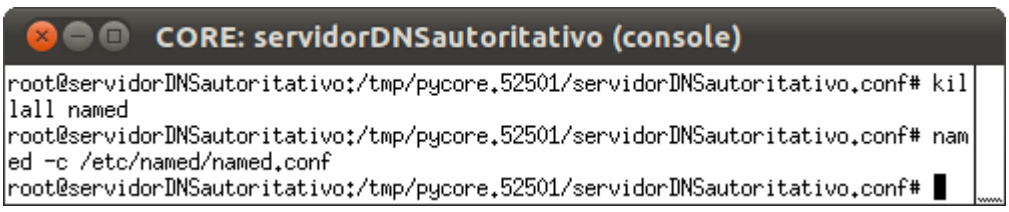
O resultado deve ser:



f. Caso os passos anteriores não tenham apresentado erros de execução, reinicie o processo do BIND para que as alterações sejam aplicadas. Para isso, digite os seguintes comandos:

```
# killall named
# named -c /etc/named/named.conf
```

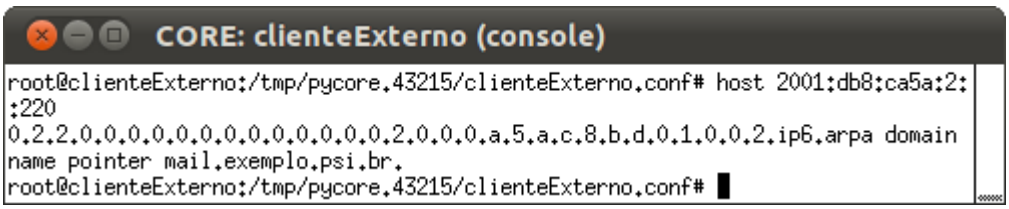
O resultado deve ser:



g. Para verificar se as alterações foram realizadas corretamente, acesse a máquina 'clienteExterno' (duplo clique sobre ela) e realize algumas requisições DNS. Para isso, utilize o comando host digitando o seguinte:

```
# host 2001:db8:ca5a:2::220
```

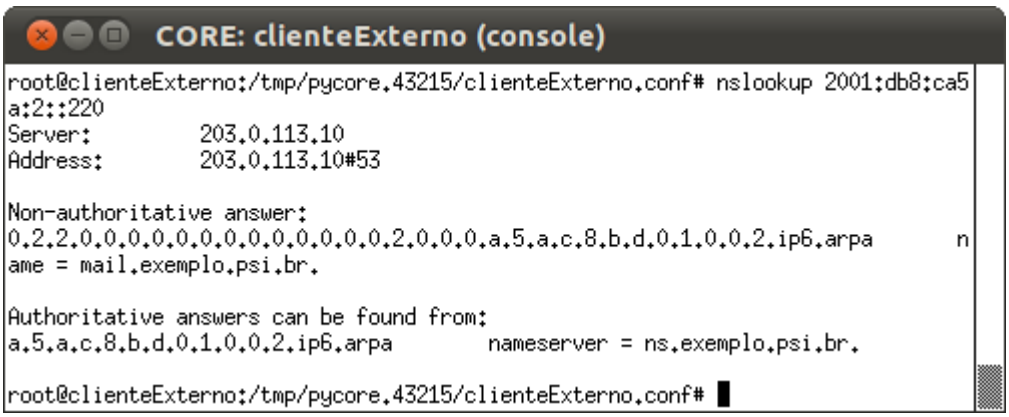
O resultado deve ser:



h. Repita os testes para os outros endereços e compare os resultados. Utilize, por exemplo, o comando nslookup para obter uma resposta mais completa digitando o seguinte:

```
# nslookup 2001:db8:ca5a:2::220
```

O resultado deve conter no mínimo as informações abaixo:



8. Para encerrar, é possível realizar uma série de testes de conectividade através do nome de uma máquina. Algumas opções podem ser feitas com a utilização dos comandos *ping* ou *ping6*, *traceroute* ou *traceroute6* e *mtr*. Alguns exemplos são:

```
# ping www.exemplo.psi.br
# ping6 www.exemplo.psi.br
# mtr exemplo.psi.br
# traceroute6 mail.exemplo.psi.br
```

9. Pare a simulação do CORE :
   c. aperte o botão 
   d. ou utilize o menu Experiment > Stop.

IPV6 - Serviços IPv6 - Servidores HTTP

Objetivo

O objetivo desse laboratório é apresentar o funcionamento em IPv6 dos servidores HTTP, Apache2 e Nginx, dois dos mais populares na Web. Para isso, ele está dividido em três experiências: na primeira é mostrado o funcionamento básico em IPv6 do Apache2; na segunda, são abordadas algumas das diferenças entre as configurações IPv4 e IPv6 do servidor Apache2 e; na última, são apresentadas as mudanças necessárias para que o servidor Nginx funcione via IPv6.

Para a realização do presente exercicio será utilizada a topologia descrita nos arquivos: **servicos-http1.imn**, **servicos-http2.imn** e **servicos-http3.imn**.

Introdução Teórica

Atualmente a Internet vive um momento de transição do IPv4 para o IPv6 devido à escassez de endereços nesta primeira versão do protocolo. Além das mudanças relacionadas às interfaces físicas dos dispositivos na rede, são necessárias alterações na maneira em que os serviços são providos, uma vez que precisam ser adaptados para suportar os novos formatos de endereços e especificidades dos protocolos de comunicação. Neste sentido, apesar de existirem outras técnicas, o ideal é a utilização de pilha dupla nos equipamentos para possibilitar a execução de ambos os protocolos em paralelo.

No presente laboratório dois dos mais populares servidores Web serão utilizados para exemplificar algumas das alterações necessárias na configuração do novo protocolo neste tipo de servidor. O primeiro exemplo será o Apache2 que é o servidor HTTP mais utilizado na Web e que suporta por padrão o IPv6 desde sua versão 2.0. E, o segundo, será o Nginx que tem aumentado a participação entre os servidores de grande porte e que recebeu suporte ao IPv6 em sua versão 08.22. Ao contrário do que acontece no Apache, o suporte à IPv6 não é habilitado por padrão neste servidor, porém, sua configuração é bastante rápida.

Apesar da configuração básica de habilitação do suporte a IPv6 ser, na maioria dos casos, bastante simples, é necessário se atentar a configurações mais complexas. Nos casos em que o servidor possui *VirtualServers* atrelados a endereços IPv4 específicos, é necessário que a configuração seja modificada para que eles fiquem também atrelados aos endereços IPv6 das interfaces de rede da máquina. Um exemplo desse tipo de caso será mostrado na segunda experiência desse laboratório.

Roteiro Experimental

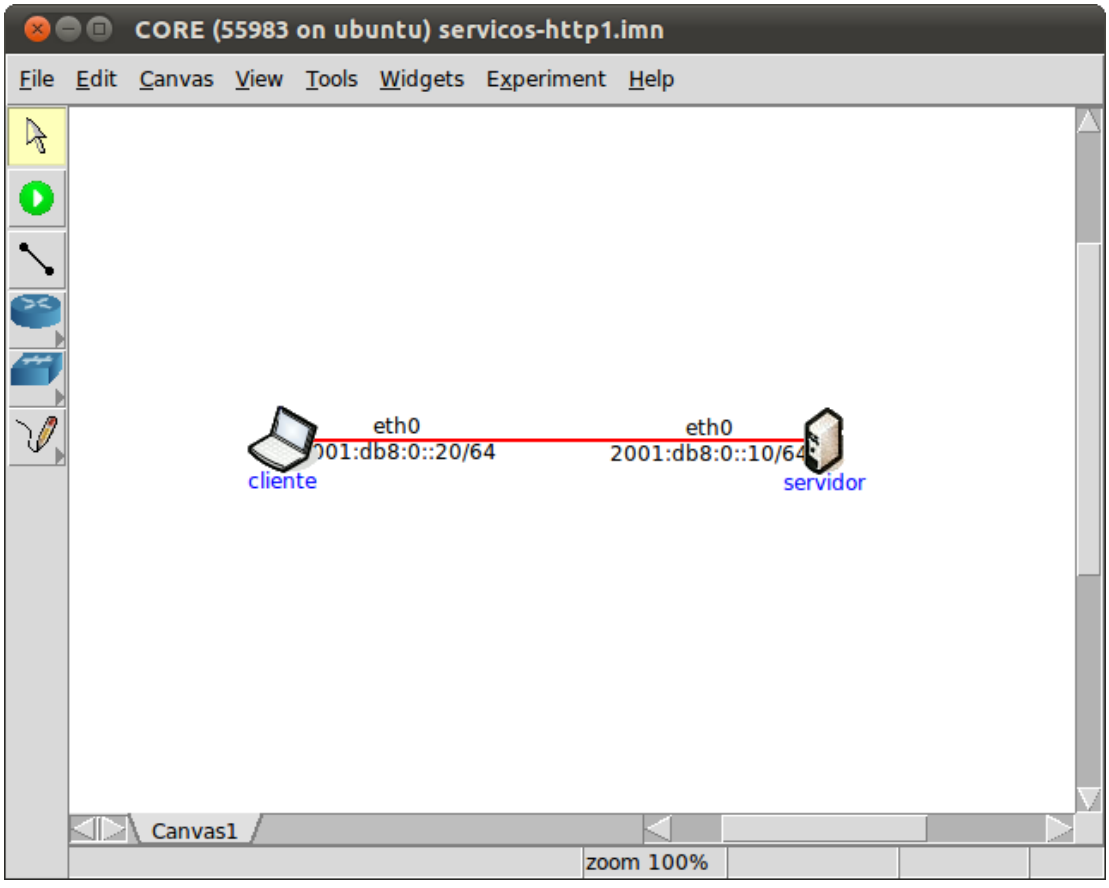
Experiência 3 - Funcionamento Básico

1. Caso esteja utilizando a máquina virtual fornecida pelo NIC.br, pule para o passo 2:

    a. O **apache2** é um servidor HTTP bastante utilizado na Web. Para instalá-lo, basta rodar o seguinte comando no Terminal:

```
$ sudo apt-get install apache2
```

2. Inicie o CORE e abra o arquivo "**servicos-http1.imn**" localizado no diretório /home/core/Desktop/servicos/http, da máquina virtual do NIC.br. A seguinte topologia deve aparecer:



3. Análise a topologia, observando os endereços, e comece o experimento:
    a. Inicie a simulação realizando um dos seguintes passos:

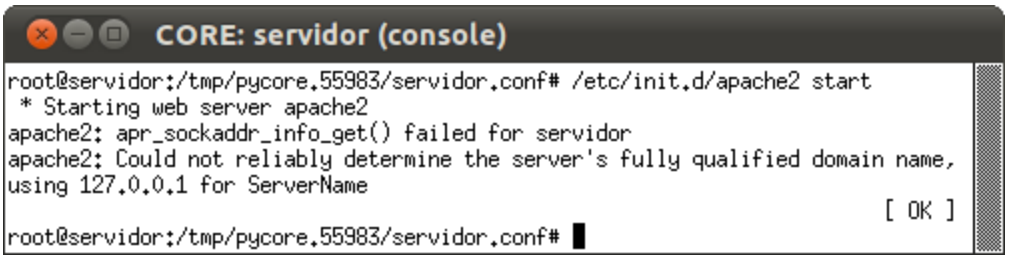
i. aperte o botão  ; ou
ii. utilize o menu Experiment > Start.

4. Inicie o apache2 no 'servidor':
   a. Abra um terminal do 'servidor' com um duplo-clique.

   b. Utilize o seguinte comando para iniciar o servidor HTTP apache2:

```
# /etc/init.d/apache2 start
```

O resultado deve ser:



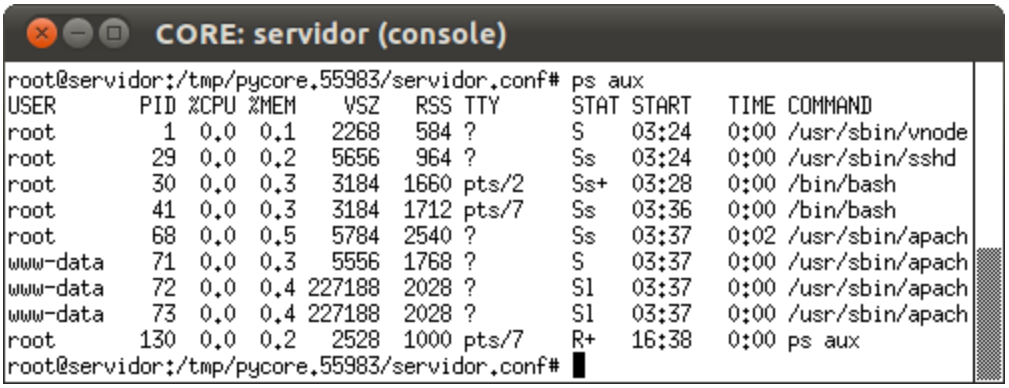
***OBS**: os erros mostrados acontecem porque os nomes de domínio e do servidor não foram configurados para a máquina, porém, eles não afetam o funcionamento do serviço para a experiência.

5. Verifique que o serviço apache2 está ativo e escutando na porta 80 das interfaces de rede da máquina 'servidor':
   a. Abra o terminal do 'servidor' com um duplo-clique.

   b. Utilize o seguinte comando para verificar que o processo apache2 está ativo:

```
# ps aux
```

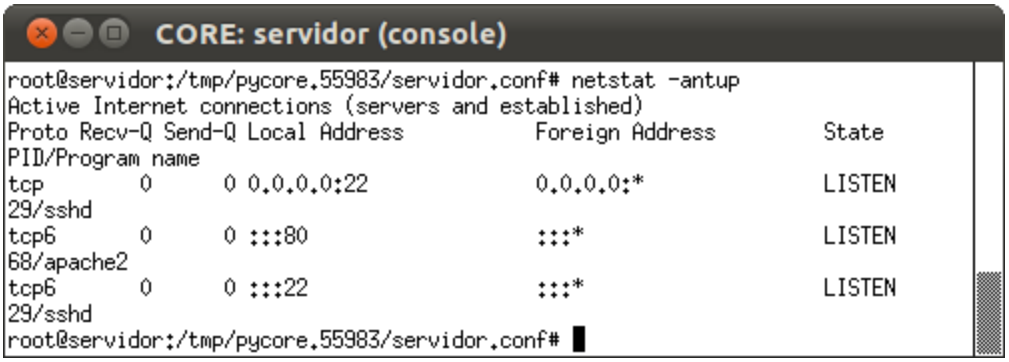
O resultado deve ser parecido com a tela seguinte:



c. Utilize o seguinte comando para verificar que o serviço apache2 está escutando a porta 80 de todas as suas interfaces de rede:

```
# netstat -antup
```

O resultado deve ser:



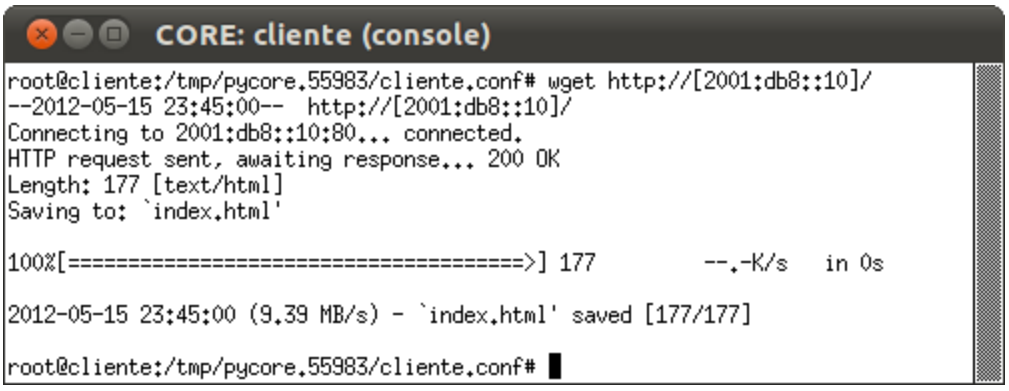
***Obs**: A string ":::80" indica a utilização da porta 80 do endereço IPv6 [::/128] que é utilizado em programação para mostrar que o serviço não está atrelado a nenhum dos endereços IPv6 do dispositivo, ou seja, que ele pode ser acessado em qualquer um dos endereços das interfaces de rede da máquina.

6. Verifique o funcionamento do servidor HTTP a partir do 'cliente':
   a. Abra um terminal do 'cliente' com um duplo-clique.

   b. Realize uma requisição HTTP GET ao 'servidor':

```
# wget http://[2001:db8::10]/
```

O resultado deve ser:



c. Veja que o arquivo foi baixado corretamente:

```
# cat index.html
```

d. Abra um terminal do servidor e verifique os logs gerados pelo apache:

```
# cat /var/log/apache2/access.log
```

***Obs**: Note que o endereço registrado pelo servidor Apache é um endereço IPv6. Isso é importante pois, caso o servidor utilize scripts ou outras ferramentas de análise de logs, essas ferramentas deverão ser capazes de analisar o formato dos endereços IPv6.

7. Encerre a simulação do experimento:
   a. Execute o seguinte comando em um terminal do 'servidor' para parar a execução do serviço apache2:

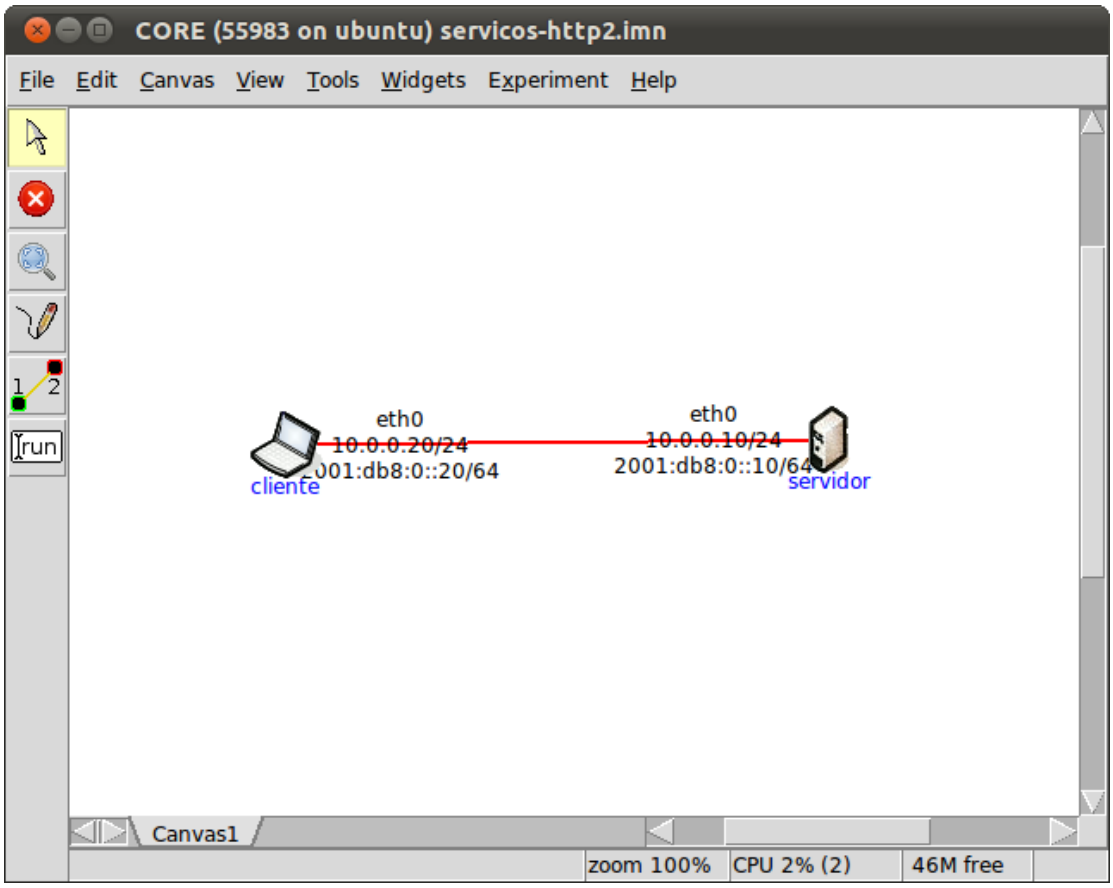
```
# /etc/init.d/apache2 stop
```

   b. Pare a simulção do CORE :
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Stop.

Expeiência 4 - Configuração IPv6 no Apache

1. Essa segunda experiência apresenta algumas especificidades da configuração de servidores Apache em redes IPv6.
   Para realizá-la, também é necessária a instalação do pacote “apache2” como indicado no item 1.a. da experiência anterior.

2. Inicie o CORE e abra o arquivo “**servicos-http2.imn**” localizado no diretório do desktop “Serviços”, da máquina virtual do NIC.br. A seguinte topologia deve aparecer:



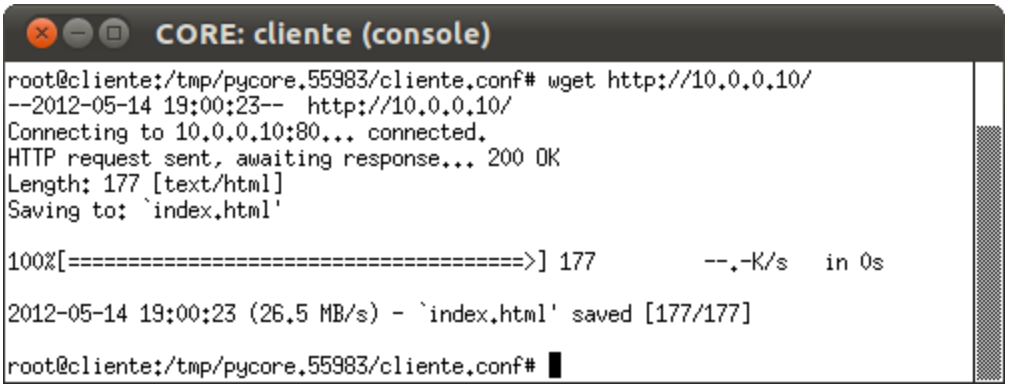
3. Verifique a configuração dos nós da topologia:
   a. Inicie a simulação executando um dos seguintes passos:
      i. aperte o botão  , ou;
      ii. utilize o menu Experiment > Start.

4. Verifique que o servidor apache está ativo apenas via IPv4:
   a. Abra um terminal do ‘cliente’ com um duplo-clique.

b. Acesse o servidor HTTP já configurado na máquina ‘servidor’:

```
# wget 10.0.0.10
```

O resultado deve ser:

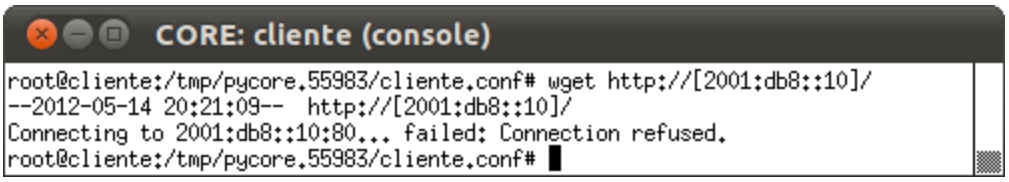


*__Obs__: Note que o arquivo ‘index.html’ foi acessado corretamente.

c. Agora utilize o endereço IPv6:

```
# wget http://[2001:db8::10]/
```

O resultado deve ser:



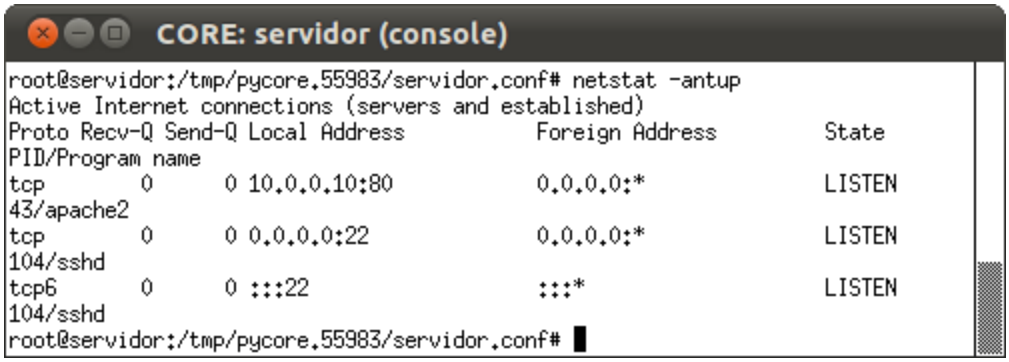
***Obs**: Note que o acesso foi recusado.

d. Verifique agora a configuração do *apache* no ‘servidor’. Abra um terminal dessa com um duplo-clique.

e. Utilize o seguinte comando para verificar os serviços ativos na máquina:

```
# netstat -antup
```

O resultado deve ser:



***Obs**: Note que a porta 80 do 'servidor' só está aberta para o endereço IPv4.

5. Nos passos a seguir o servidor será configurado para escutar também a porta 80 de sua interface IPv6:

    a. No terminal do servidor, abra o arquivo "`/etc/apache2/ports.conf`". Um editor de texto que pode ser utilizado é o nano. Para utilizá-lo, digite o seguinte comando no terminal:

    ```
    # nano /etc/apache2/ports.conf
    ```

    b. Localize as seguintes linhas no arquivo:

    ```
    NameVirtualHost 10.0.0.10:80
    Listen 10.0.0.10:80
    ```

    Note que elas configuram o servidor apache para escutar apenas na interface IPv4. Por padrão, o apache é configurado para escutar todas as interfaces de rede da máquina que está instalado com as linhas (**não altere o arquivo para incluir estas linhas**):

    ```
    NameVirtualHost *:80
    Listen 80
    ```

    Porém, a especificação de endereços IP é bastante comum em servidores de produção com mais de um site configurado. Nesses casos, para que o site seja visto também em IPv6, os novos endereços tem de ser adicionados na configuração.

c. Logo abaixo da configuração do endereço IPv4, adicione as linhas:

```
NameVirtualHost [2001:db8::10]:80
Listen [2001:db8::10]:80
```

***Obs**: Note que o endereço IPv6 deve estar entre colchetes.
Utilize o comando '*Ctrl+X*' seguindo de '*Y*' e '*enter*' para sair e salvar as modificações do arquivo.

d. Abra o arquivo "`/etc/apache2/sites-available/default`" para alterar também a configuração do VirtualHost. Isso pode ser feito com a utilização do editor de texto nano:

```
# nano /etc/apache2/sites-available/default
```

Altere a linha:

```
<VirtualHost 10.0.0.10:80>
```

Para a seguinte:

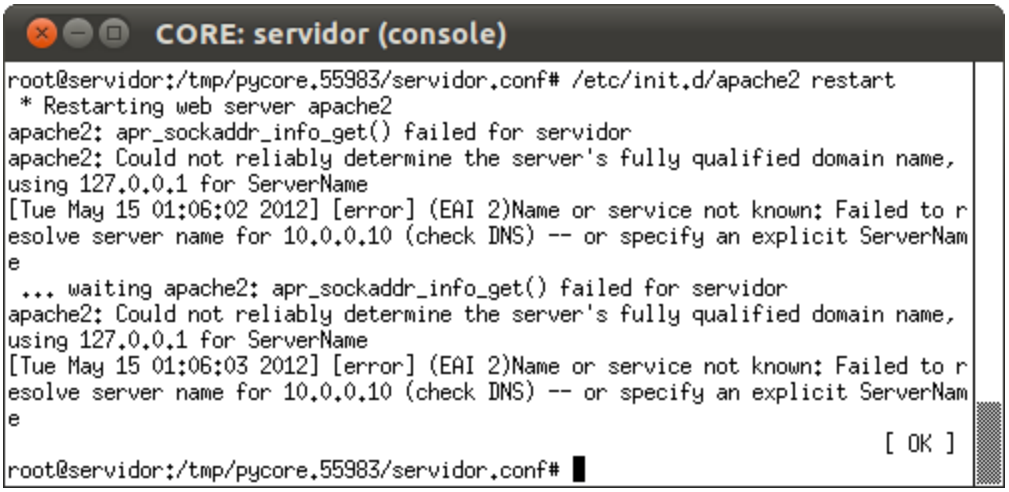
```
<VirtualHost 10.0.0.10:80 [2001:db8::10]:80>
```

Salve o arquivo e feche o editor de texto.

e. Reinicie o serviço apache2 com código a seguir para que ele faça uso das novas configurações:

```
# /etc/init.d/apache2 restart
```

O resultado deve ser:

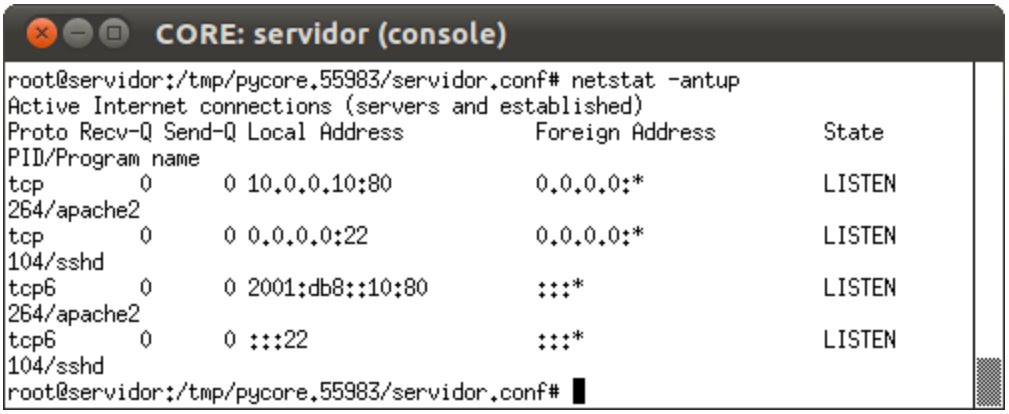


***OBS**: os erros mostrados acontecem porque os nomes de domínio e do servidor não foram configurados para a máquina, porém, eles não afetam o funcionamento do serviço para a experiência.

f. Rode novamente o comando *netstat*, como mostrado a seguir, para verificar que o servidor *apache2* está escutando a porta 80 tanto de seu endereço IPv4 quanto IPv6:

```
# netstat -antup
```

O resultado deve ser:



g. Para confirmar o funcionamento do servidor HTTP, abra um terminal do 'cliente' e utilize os seguintes comandos:

```
# wget http://10.0.0.10/
# wget http://[2001:db8::10]/
```

O resultado deve ser:

6. Encerre a simulação do experimento:
   a. Execute o seguinte comando em um terminal do 'servidor' para parar a execução do serviço apache2:

```
# /etc/init.d/apache2 stop
```

   b. Pare a simulação do CORE :
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Stop.

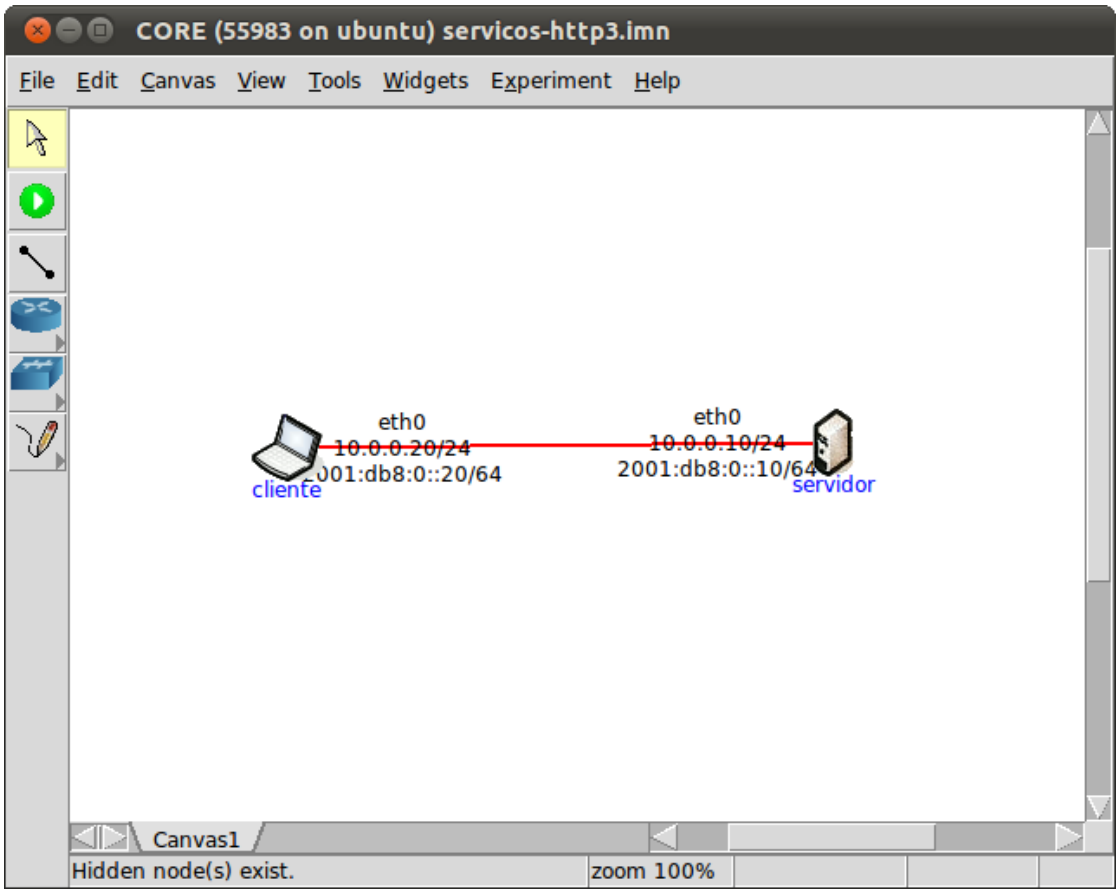
Experiência 5 - Configuração IPv6 do Nginx

1. Caso não esteja utilizando a máquina virtual fornecida pelo NIC.br é preciso, siga o passo a seguir:

    a. O **nginx** é um servidor HTTP de alta performance bastante utilizado na Web. Para instalá-lo, basta rodar o seguinte comando no Terminal:

```
$ sudo apt-get install nginx
```

2. Inicie o CORE e abra o arquivo "**servicos-http3.imn**" localizado no diretório do desktop "Serviços", da máquina virtual do NIC.br. A seguinte topologia deve aparecer:



3. Verifique a configuração dos nós da topologia:
    a. Inicie a simulação realizando um dos seguintes passos:
        i. aperte o botão  ; ou
        ii. utilize o menu Experiment > Start.

4. Inicie o Nginx no ‘servidor’:
    a. Abra um terminal do ‘servidor’ com um duplo-clique.

    b. Utilize o seguinte comando para iniciar o servidor HTTP Nginx:

```
# /etc/init.d/nginx start
```

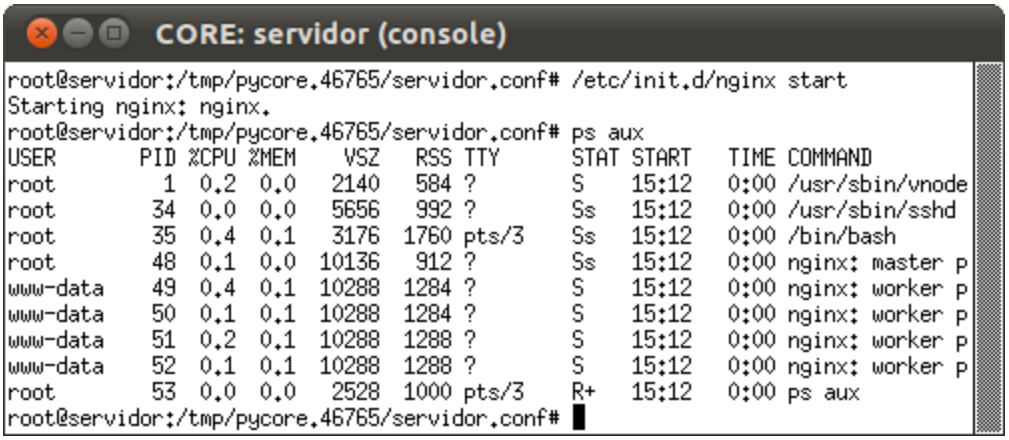
    O resultado deve ser:

5. Verifique que o serviço nginx está ativo e que, ao contrário do Apache2, ele escuta apenas porta 80 do endereço IPv4 da máquina ‘servidor’:
    a. Abra o terminal do ‘servidor’ com um duplo-clique.

    b. Utilize o seguinte comando para verificar que alguns processos do nginx estão ativos:

```
# ps aux
```

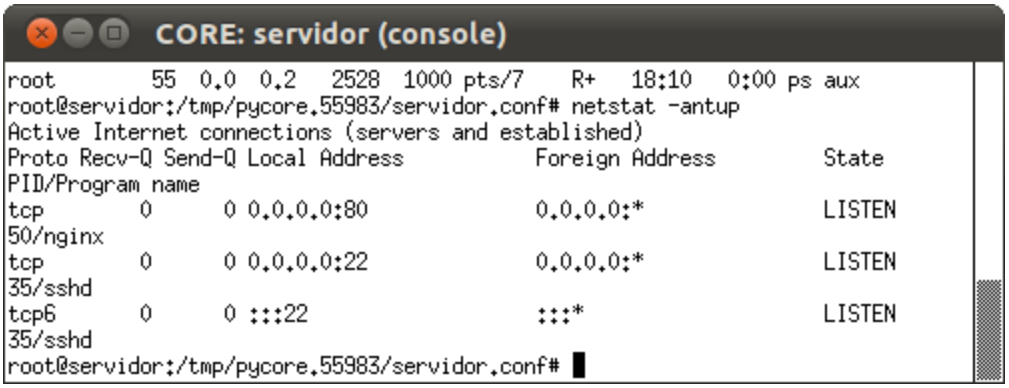
    O resultado deve ser:



    c. Utilize o seguinte comando para verificar que o seviço Nginx está ouvindo a porta 80 apenas de seu endereço IPv4:

```
# netstat -antup
```

O resultado deve ser:



6. Configure o servidor Nginx para funcionar também em IPv6:
   a. Ainda no terminal do servidor, modifique o arquivo de configurações "`/etc/nginx/sites-available/default`". Para isso o editor de texto *nano* pode ser utilizado:

```
# nano /etc/nginx/sites-available/default
```

   b. Nesse arquivo localize as linhas 21 e 22, a seguir mostradas, e descomente-as (remova o caracter '#' do inicio da linha):

```
#listen   80; ## listen for ipv4; this line is default and implied
#listen   [::]:80 default ipv6only=on; ## listen for ipv6
```

   c. Utilize o comando '*Ctrl+X*' seguindo de '*Y*' e '*enter*' para sair e salvar as modificações do arquivo.

   d. Renicie o servidor com o seguinte comando:

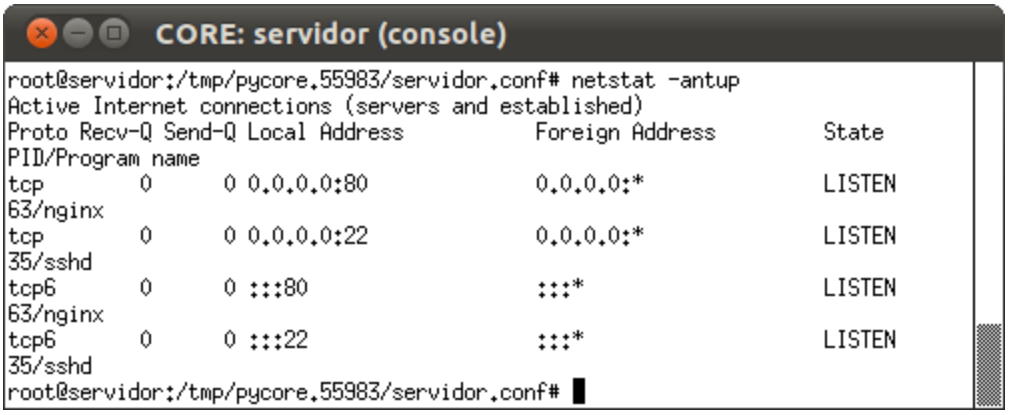
```
# /etc/init.d/nginx restart
```

O resultado deve ser:

e. Verifique que agora o serviço nginx escuta também a porta 80 das interfaces IPv6 do servidor:

```
# netstat -antup
```

O resutado deve ser:



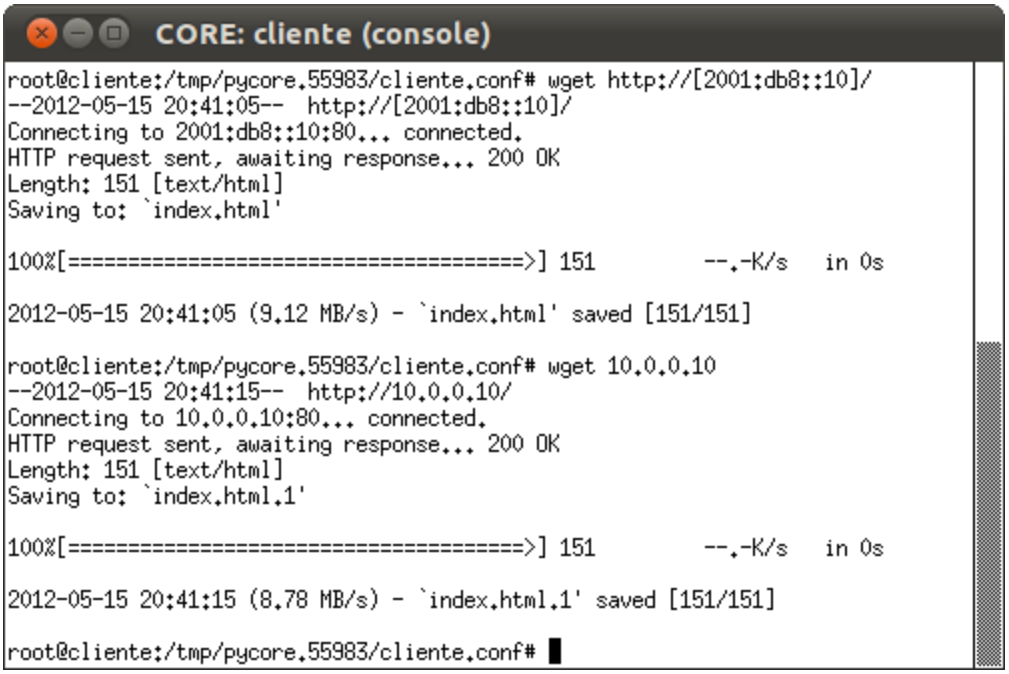
***Obs**: A string ":::80" indica a utilização da porta 80 do endereço IPv6 [::/128] que é utilizado em programação para mostrar que o serviço não está atrelado a nenhum dos endereços IPv6 do dispositivo, ou seja, que ele pode ser acessado em qualquer um dos endereços das interfaces de rede da máquina.

7. Verifique o funcionamento do servidor HTTP a partir do 'cliente':
   a. Abra um terminal do 'cliente' com um duplo-clique.

   b. Realize requisições HTTP GET ao 'servidor' em ambos os endereços do servidor:

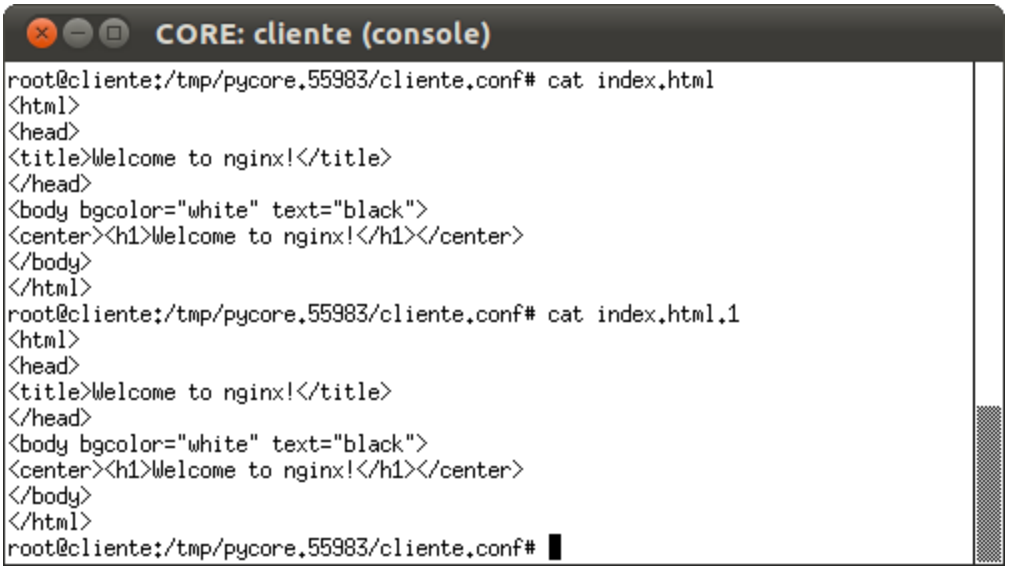
```
# wget http://[2001:db8::10]/
# wget 10.0.0.10
```

O resultado deve ser:



c. Veja que os arquivos foram baixado corretamente:

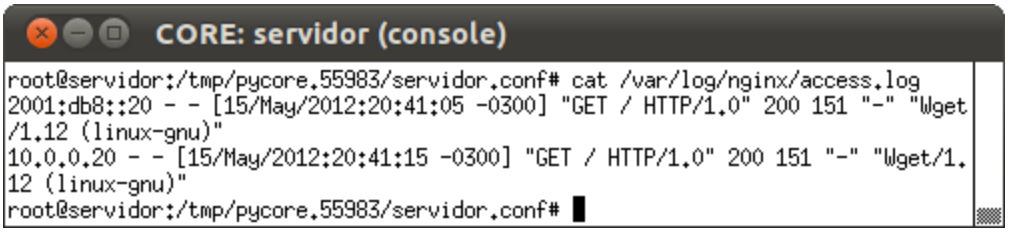
```
# cat index.html
# cat index.html.1
```



***Obs**: Verifique que os arquivos são iguais já que apenas um VirtualServer foi configurado no Nginx.

d. Abra um terminal do servidor e verifique os logs gerados pelo Nginx:

```
# cat /var/log/nginx/access.log
```



***Obs**: Note que, assim como no Apache, o Nginx registra endereços tanto IPv6 quanto IPv4 no arquivo de logs. Isso é importante pois, caso o servidor utilize scripts ou outras ferramentas de análise de logs, essas ferramentas deverão ser capazes de analisar também o formato dos endereços IPv6.

8. Encerre a simulação do experimento:
   a. No terminal do servidor execute o seguinte comando:

```
# /etc/init.d/nginx stop
```

O resultado deve ser:

   b. pare a simulão do CORE

      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Stop.

IPV6 - Serviços IPv6 - Proxy

Objetivo

Esta esperiência tem o objetivo de mostrar a configuração básica de um proxy configurado para permitir o acesso à Internet IPv4 por uma rede unicamente IPv6.

Para a realização do presente exercicio será utilizada a topologia descrita no arquivo: **servicos-proxy1.imn**.

Introdução Teórica

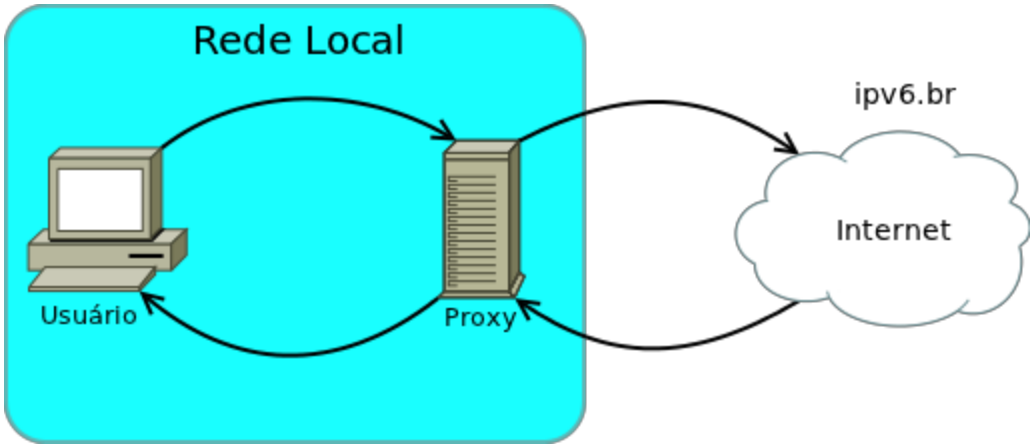
Atualmente a Internet vive um momento de transição do IPv4 para o IPv6 devido à escassez de endereços nesta primeira versão do protocolo. Além das mudanças relacionadas às interfaces físicas dos dispositivos na rede, são necessárias alterações na maneira em que os serviços são providos, uma vez que precisam ser adaptados para suportar os novos formatos de endereços e especificidades dos protocolos de comunicação. Neste sentido, apesar de existirem outras técnicas, o ideal é a utilização de pilha dupla nos equipamentos para possibilitar a execução de ambos os protocolos em paralelo.

Um servidor proxy é um tipo de serviço muito utilizado na Internet e, algumas de suas principais funções são:

- Manter dispositivos anônimos, principalmente por questões de segurança;
- Melhorar o desempenho de aplicações Web com a utilização de caches;
- Filtrar acesso à conteúdos ou serviços, principalmente em redes corporativas;
- Manter registro da utilização da Internet por uma rede interna;
- Verificar a existência de *malware* antes de retornar um conteúdo requisitado.

Com a troca dos protocolos de Internet, servidores proxy podem também ser utilizados como mecanismo de transição, que permitem o acesso Web a partir de uma rede IPv6 para um Web site IPv4 ou vice-versa.

No presente laborátório o proxy de código aberto *squid* será configurado como proxy direto de uma rede local para exemplificar tal funcionalidade. A figura abaixo esquematiza essa situação:



O *squid* foi escolhido para este laboratório por ser um dos servidores proxy mais utilizados para plataforma Linux. Ele suporta IPv6 desde sua versão 2.6 com a utilização de um patch específico e por padrão desde a 3.1. Outros servidores como o Apache, o Nginx ou o MS ISA-Server também podem ser utilizados de maneira semelhante porém para cada um deles existe uma configuração diferente para operar com IPv6.

Roteiro Experimental

Experiência 6 - Forward Proxy em rede IPv6

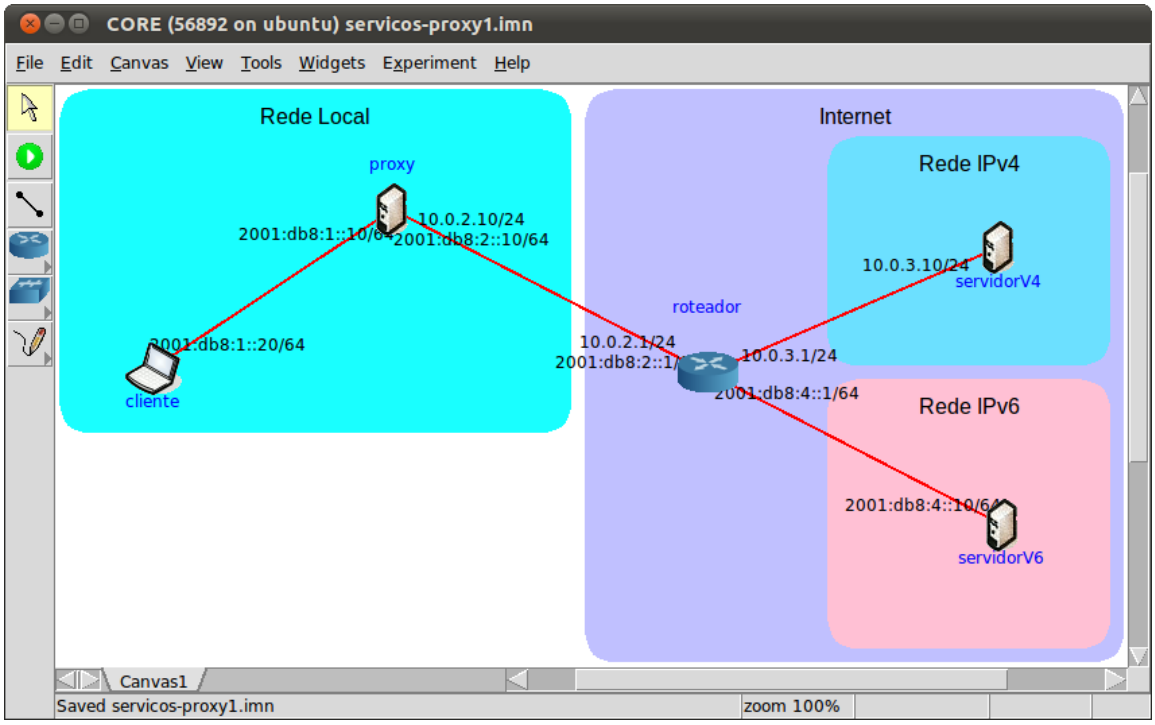
1. Caso não esteja utilizando a máquina virtual fornecida pelo NIC.br, siga os passos a seguir:
    a. O **squid3** é um servidor HTTP bastante utilizado na Web. Para instalá-lo, basta rodar o seguinte comando no Terminal:

```
$ sudo apt-get install squid3
```

    b. O **apache2** é um servidor HTTP bastante utilizado na Web. Para instalá-lo, basta rodar o seguinte comando no Terminal:

```
$ sudo apt-get install apache2
```

2. Inicie o CORE e abra o arquivo "**servicos-proxy1.imn**" localizado no diretório do desktop "servicos/proxy/", da máquina virtual do NIC.br. A seguinte topologia deve aparecer:



Essa topologia representa uma situação em que a rede local de um cliente possui apenas IPv6 e, para acessar conteúdo Web disponível em endereços IPv4, utiliza um proxy. Nela, o roteamento de pacotes foi configurado com a utilização de rotas estática e, a máquina 'proxy' além de suas funções básicas, funciona também como um roteador da rede local para simplificar a topologia.

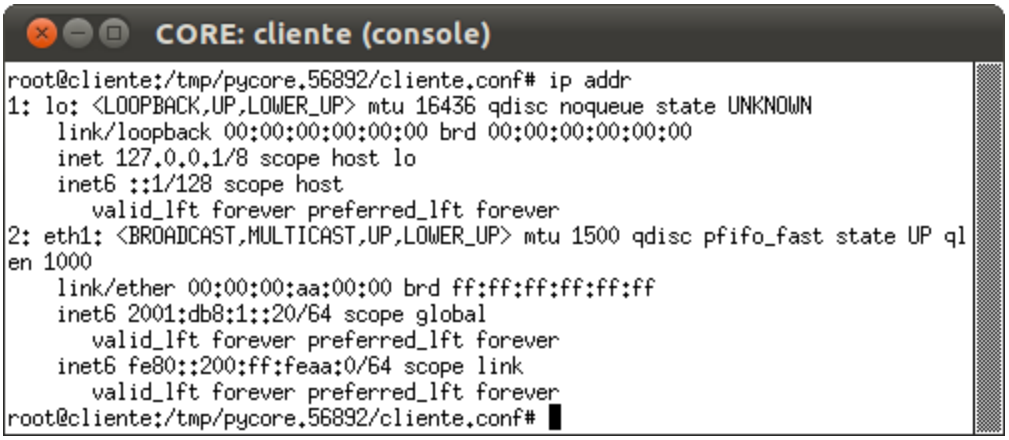
3. Verifique a configuração dos nós da topologia:
   a. Inicie a simulação realizando um dos seguintes passos:
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do 'cliente' com um duplo-clique.

   c. Verifique, com o comando abaixo, a configuração da interface de rede do 'cliente' e note que ele não está configurado com endereço IPv4:

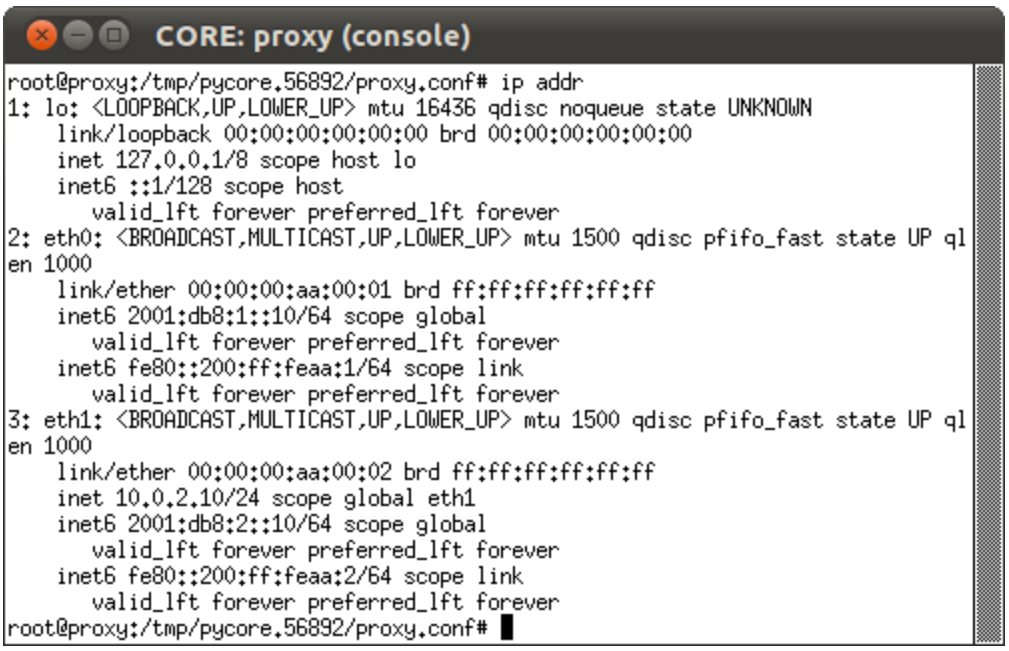
```
# ip addr
```

   O resultado deve ser:



***Obs**: Esse comando possibilita a verificação dos endereços das interfaces de rede da máquina.

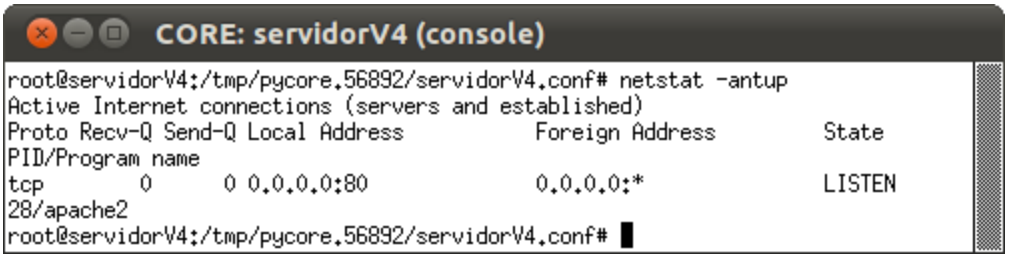
d. Observe a configuração do 'proxy' utilizando o mesmo comando.
O resultado deve ser:

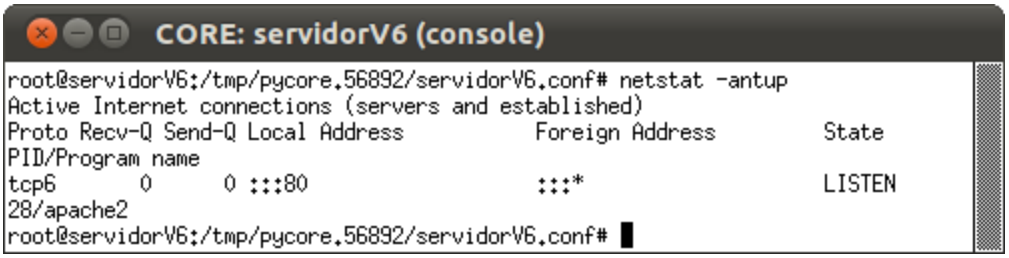


e. Em ambos os servidores, 'servidorV4' e 'servidorV6', utilize o comando abaixo para verificar se eles estão executando o servidores Web (Apache2) corretamente:

```
# netstat -antup
```

Os resutados devem ser semelhantes à:



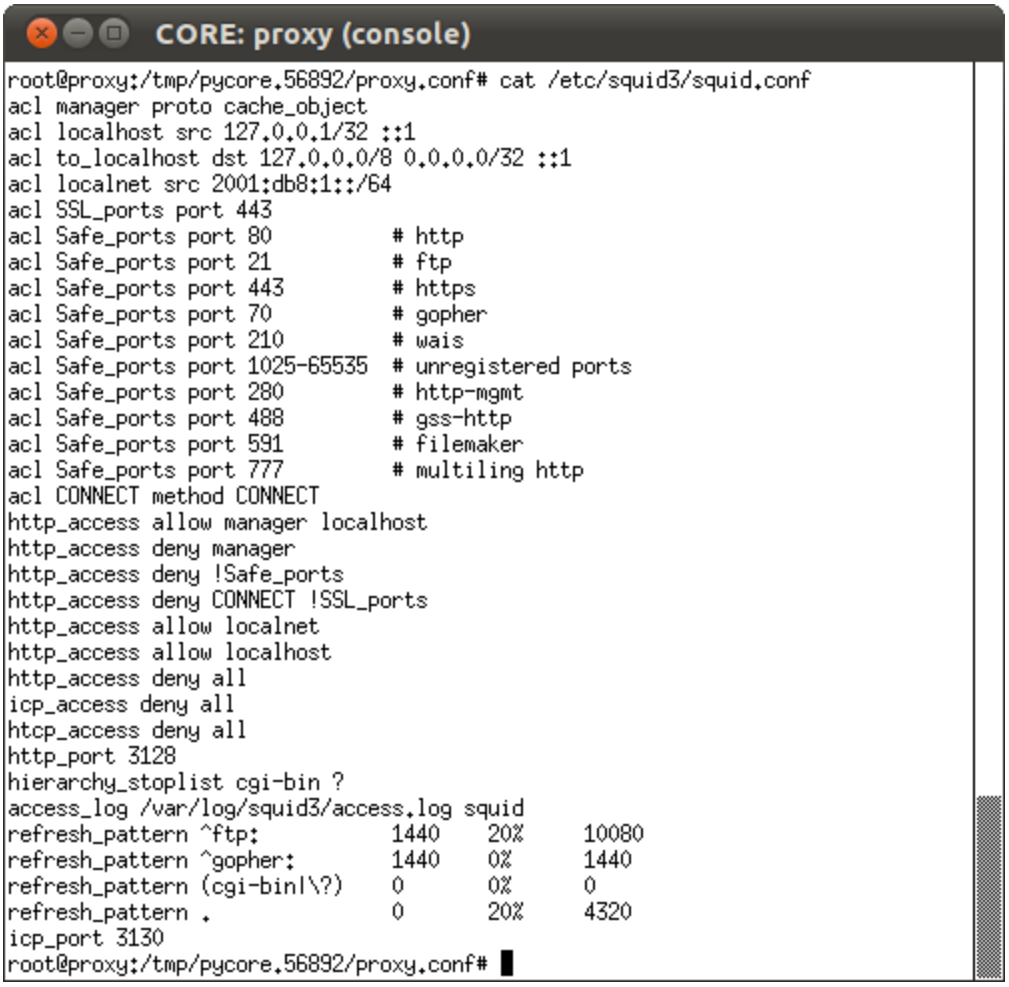


4. Inicie o squid3 no 'proxy':
   a. Abra um terminal do 'proxy' com um duplo-clique.

   b. Utilize o seguinte comando para acessar as configurações do sistema squid3 que será utilizado como proxy Web nesse servidor:

```
# cat /etc/squid3/squid.conf
```

O resultado deve ser:

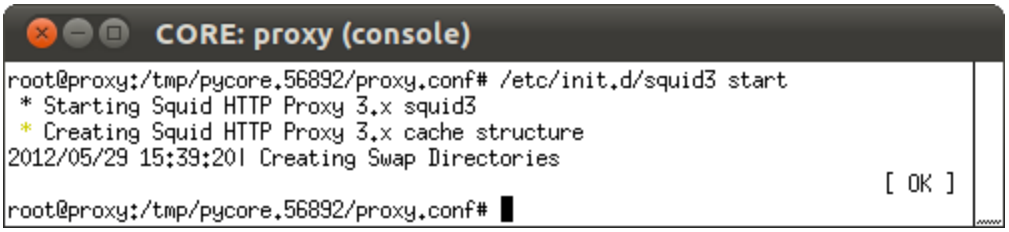


***OBS**: Note que, com o comando "`http_access allow localnet`", o serviço permitirá acesso da rede local que foi definida pela mascara "2001:db8:1::/64" na linha "`acl localnet src 2001:db8:1::/64`". Ainda, na linha "`http_port 3128`" o squid3 é configurado para escutar a porta 3128.

c. Utilize o comando a seguir para iniciar o serviço squid:

```
# /etc/init.d/squid3 start
```

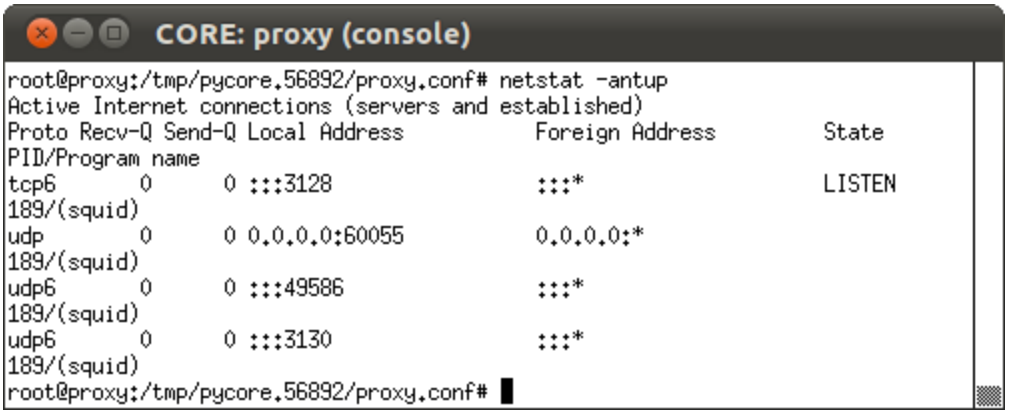
O resultado deve ser:



d. Confirme que o serviço está escutando a porta 3128 com o seguinte comando:

```
# netstat -antup
```

O resultado deve ser similar à:



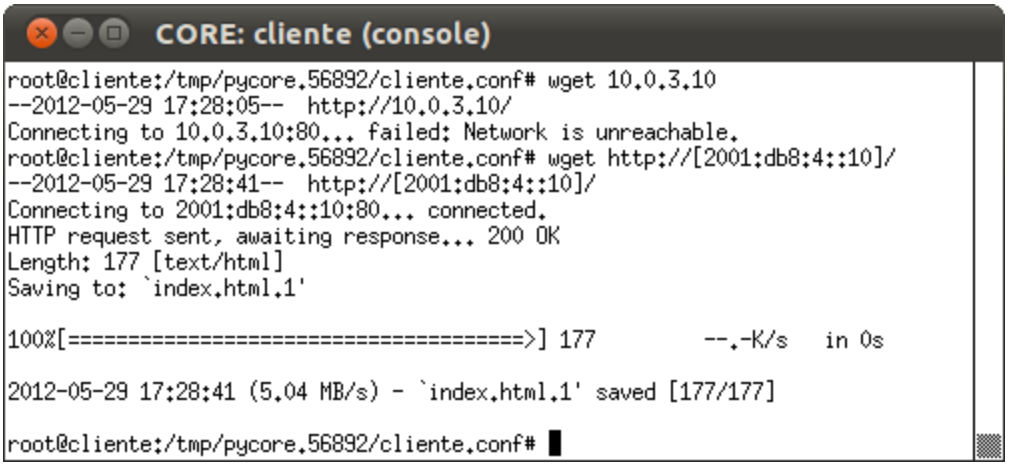
***OBS**: Note que a porta UDP 3130 foi configurada para o serviço ICP (*Internet Cache Protocol*) útil em casos de comunicação entre servidores proxy. E, as outras portas UDP, são escolhidas aleatóriamente por padrão. Na prática, elas são utilizadas com os módulos ICP, HTCP e DNS do squid.

5. Configure o cliente para utilizar o 'proxy' para acessar páginas Web disponíveis apenas em IPv4.
   a. Abra o terminal do 'cliente' com um duplo-clique.

   b. Utilize o seguinte comando para verificar que o apenas o 'servidorV6' está acessível:

```
# wget 10.0.3.10
# wget http://[2001:db8:4::10]/
```

O resultado deve ser:



c. Configure as variáveis de ambiente de forma a que o proxy seja utilizado para acessar páginas Web. Para tanto, utilize os comandos:

```
# export http_proxy="http://[2001:db8:1::10]:3128/"
# export no_proxy=localhost,127.0.0.1
```

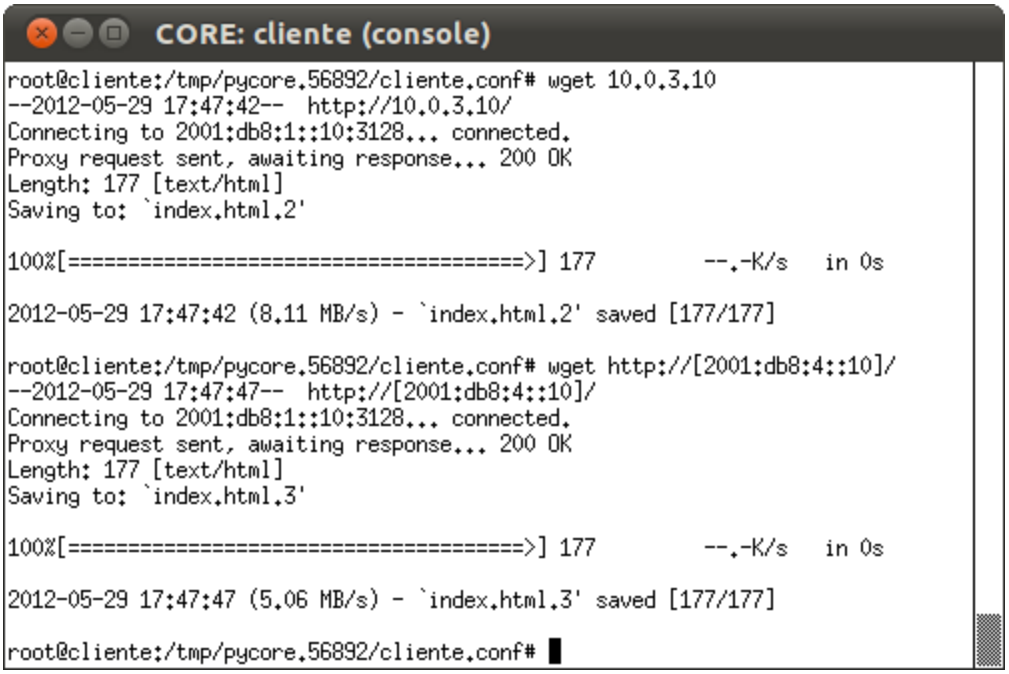
O Resultado deve ser:

***OBS**: essa configuração de proxy é utilizada apenas pela aplicação *wget*. Para configurar outros browsers com servidores proxy, verifique documentação específica.

6. Verifique o funcionamento da solução:
   a. No terminal do 'cliente' acesse novamente ambos os servidores web:

```
# wget 10.0.3.10
# wget http://[2001:db8:4::10]/
```

O resultado deve ser:

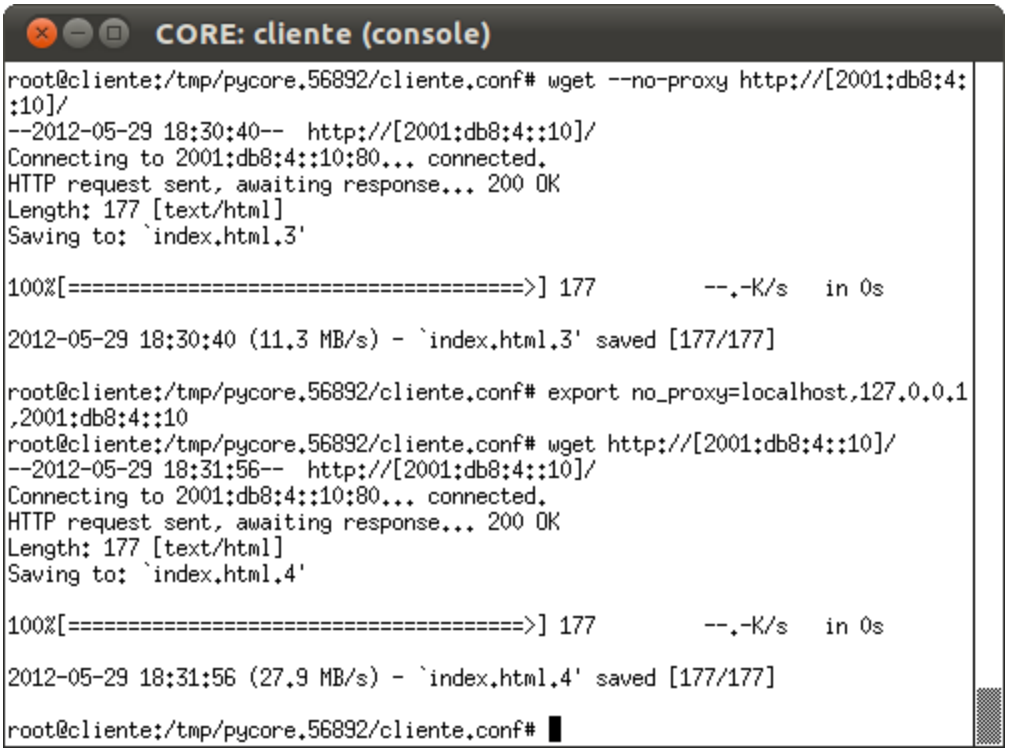


Verifique que em ambos os casos as páginas utilizaram o proxy para acessar o serviço.

b. Como normalmente a verificação do endereço de um site é feita somente no momento da requisição, não é trivial configurar o 'cliente' para utilizar o proxy apenas para acessar sites disponíveis unicamente em IPv4. Porém, endereços específicos podem ser diretamente acessados via IPv6 se adicionados na variável de ambiente `no_proxy` ou com a utilização do parâmetro `--no-proxy` na chamada do comando `wget`. Os comandos abaixo exemplificam a configuração do acesso direto para o 'servidorV6':

```
# wget --no-proxy http://[2001:db8:4::10]/
# export no_proxy=localhost,127.0.0.1,2001:db8:4::10
# wget http://[2001:db8:4::10]/
```

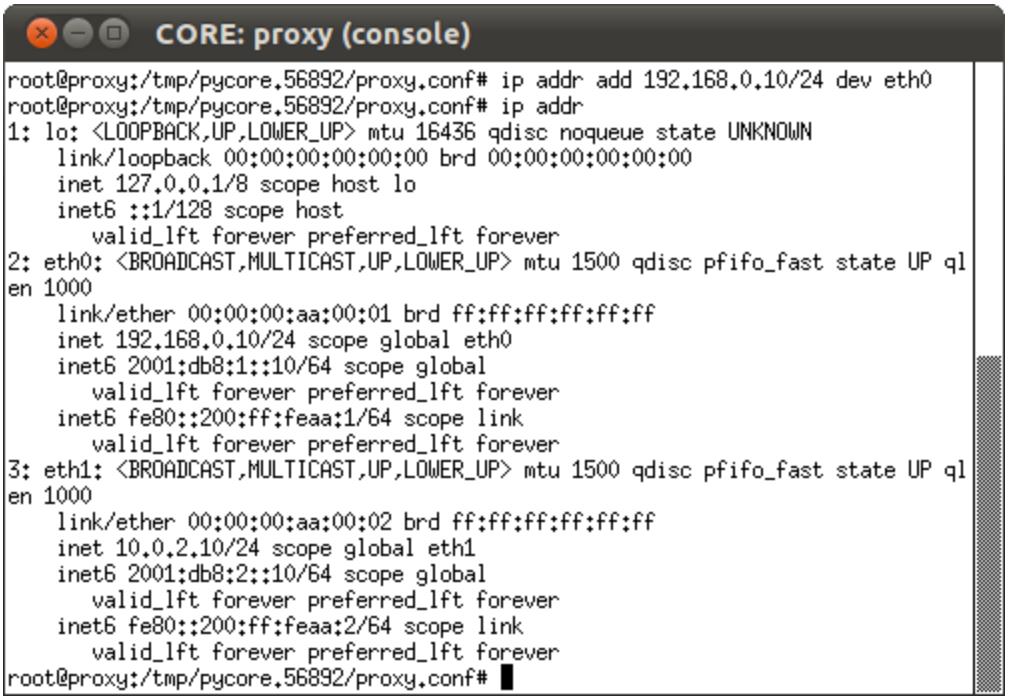
O resultado deve ser:



7. Verifique agora que o inverso também é valido, a partir de uma rede IPv4 pode-se acessar um servidor Web que esteja configurado apenas em IPv6:
    a. Abra um terminal do 'proxy', com um duplo clique, e adicione um endereço IPv4 na interface de rede eth0:

```
# ip addr add 192.168.0.10/24 dev eth0
# ip addr
```

O resultado deve ser:



b. Configure o servidor *squid3* para aceitar requisições da nova rede IPv4 estabelecida. Para isso, utilize o editor de texto *nano* para modificar o arquivo "`/etc/squid3/squid.conf`":

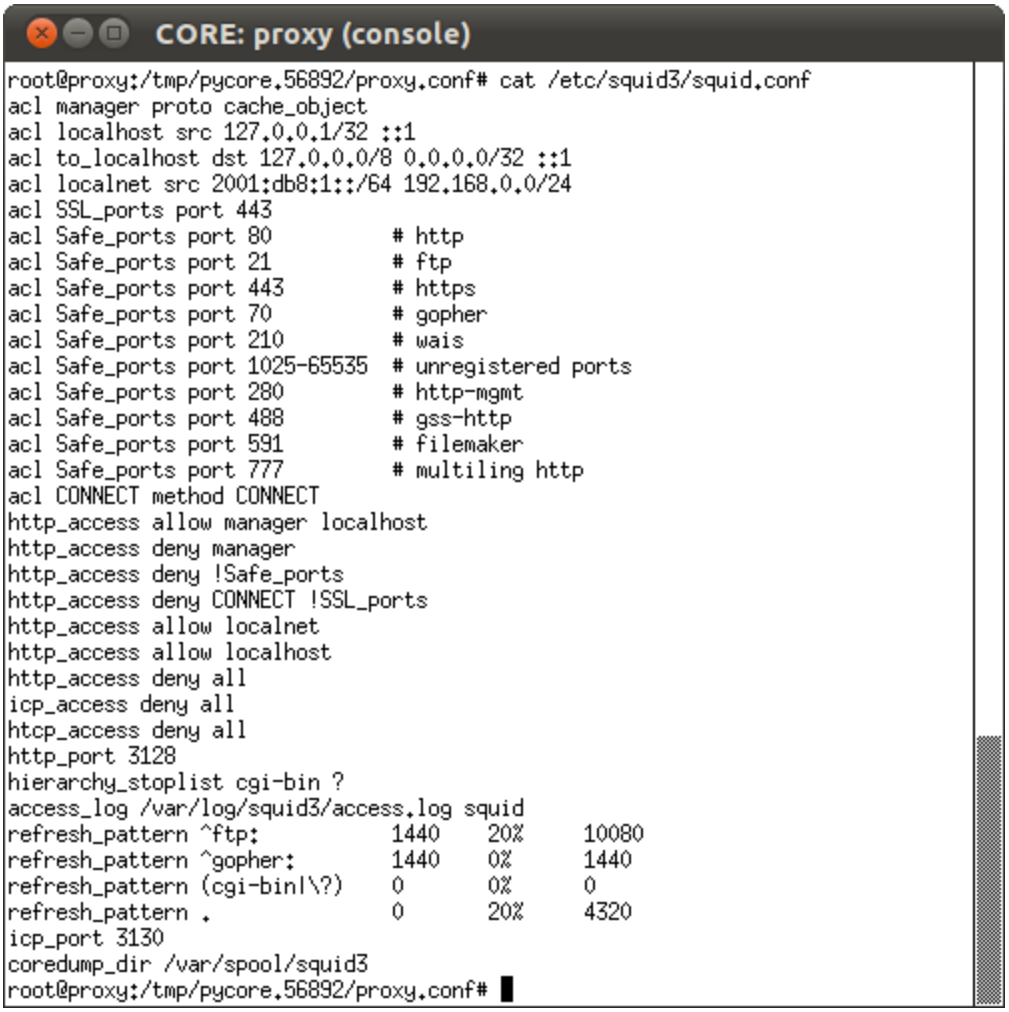
```
# nano /etc/squid3/squid.conf
```

Nesse arquivo altere a linha "`acl localnet src 2001:db8:1::/64`" para que ela fique assim:

```
acl localnet src 2001:db8:1::/64 192.168.0.0/24
```

Save e feche o arquivo com um "Ctrl+x" seguido de Y e enter.

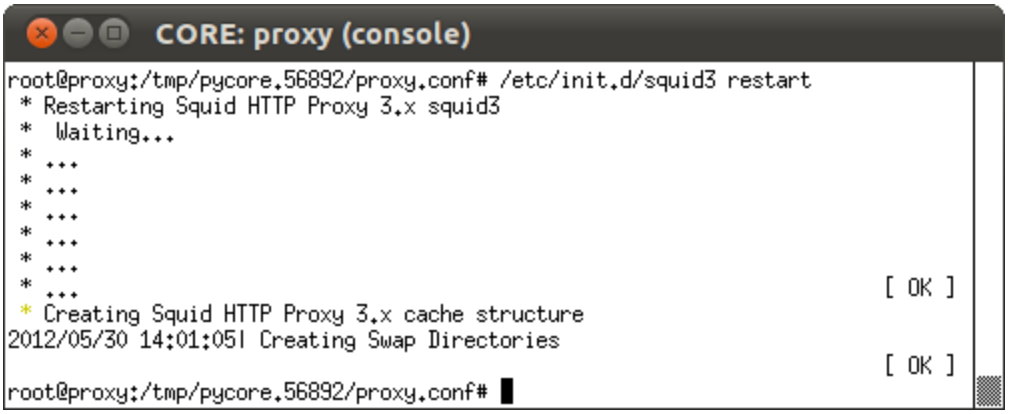
O arquivo deve ficar assim:



c. Reinicie o serviço:

```
# /etc/init.d/squid3 restart
```

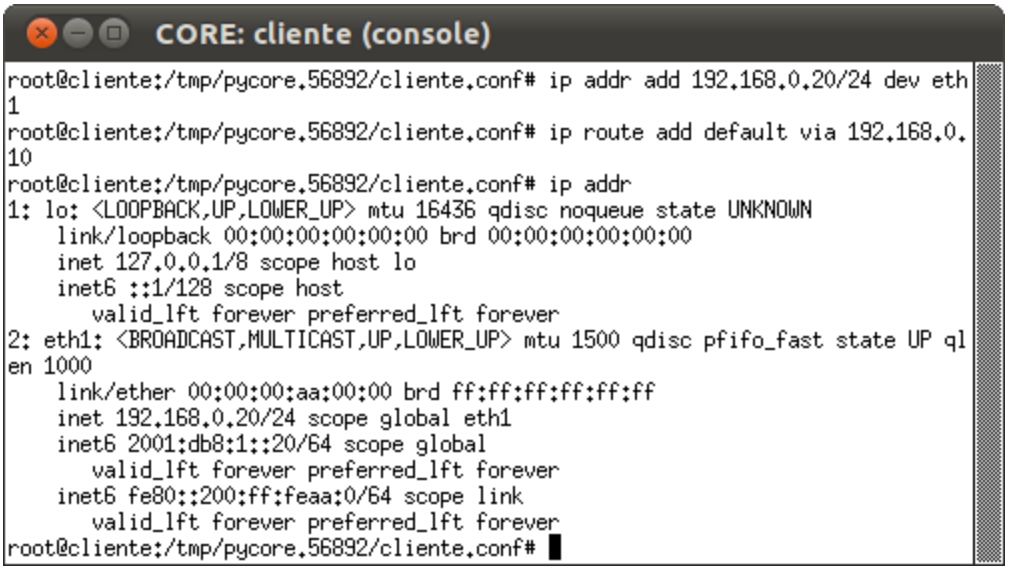
O resultado deve ser parecido com:



d. No terminal do 'cliente', adicione na interface de rede eth1 um endereço IPv4:

```
# ip addr add 192.168.0.20/24 dev eth1
# ip route add default via 192.168.0.10
# ip addr
```

O resultado deve ser:



e. Configure o proxy para utilizar a rede IPv4:

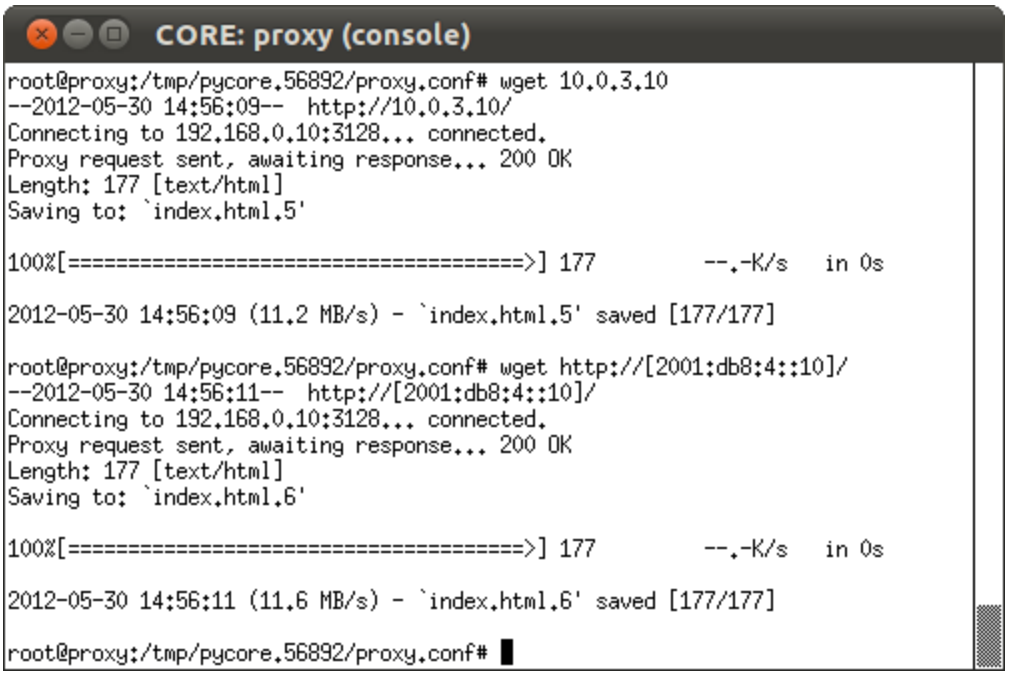
```
# export http_proxy="http://192.168.0.10:3128/"
```

O resultado deve ser:

f. Verifique que ambos os servidores, 'servidorV4' e 'servidorV6', continuam sendo acessíveis via proxy:

```
# wget 10.0.3.10
# wget http://[2001:db8:4::10]/
```

O resultado deve ser:



Note que em ambos os casos o acesso é feito com a utilização do serviço de proxy que se encontra na porta 3128 do endereço IPv4 192.168.0.10.

8. Encerre a simulação do experimento:
   a. Execute o seguinte comando em um terminal do 'proxy' para parar a execução do serviço *squid3*:

```
# /etc/init.d/squid3 stop
```

b. Pare a simulção do CORE :

   i. aperte o botão  ; ou
   ii. utilize o menu Experiment > Stop.

IPV6 - Serviços IPv6 - Proxy Reverso

Objetivo

Esta esperiência tem o objetivo de mostrar a configuração básica de um proxy reverso configurado para permitir o acesso à um Web site alocado em uma rede puramente IPv6 à partir da Internet.

Para a realização do presente exercício será utilizada a topologia descrita no arquivo: **servicos-reverseproxy1.imn**.

Introdução Teórica

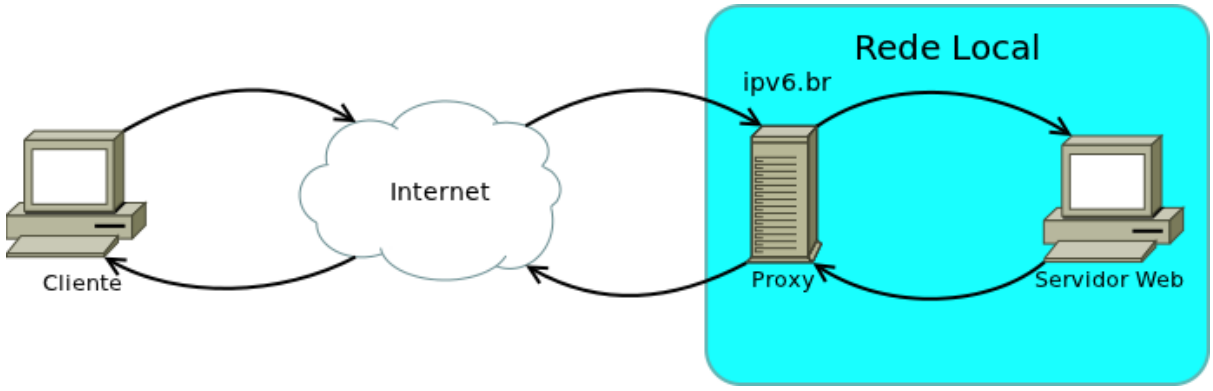
Atualmente a Internet vive um momento de transição do IPv4 para o IPv6 devido à escassez de endereços nessa primeira versão do protocolo. Além das mudanças relacionadas às interfaces físicas dos dispositivos na rede, são necessárias alterações na maneira em que os serviços são providos, uma vez que precisam ser adaptados para suportar os novos formatos de endereços e especificidades dos protocolos de comunicação. Neste sentido, apesar de existirem outras técnicas, o ideal é a utilização de pilha dupla nos equipamentos para possibilitar a execução de ambos os protocolos em paralelo.

Um servidor proxy é um tipo de serviço muito utilizado na Internet e, algumas de suas principais funções são:

- Manter dispositivos anônimos, principalmente por questões de segurança;
- Melhorar o desempenho de aplicações Web com a utilização de caches;
- Filtrar acesso à conteúdos ou serviços, principalmente em redes corporativas;
- Manter registro da utilização da Internet por uma rede interna;
- Verificar a existência de *malware* antes de retornar um conteúdo requisitado.

Com a troca dos protocolos de Internet, servidores proxy podem também ser utilizados como mecanismo de transição, que permitem o acesso Web a partir de uma rede IPv6 para um Web site IPv4 ou vice-versa.

No presente laborátório, o proxy de código aberto *squid* será configurado como proxy reverso de uma rede comercial de um servidor Web para exemplificar como tal funcionalidade pode ser aplicada em um serviço de cache. A figura abaixo esquematiza essa situação:



O *squid* foi escolhido para esse laboratório por ser um dos servidores proxy mais utilizados para plataforma Linux. Ele suporta IPv6 desde sua versão 2.6 com a utilização de um patch específico e por padrão desde a 3.1. Outros servidores como o Apache, o Nginx ou o MS ISA-Server também podem ser utilizados de maneira semelhante porém para cada um deles existe uma configuração diferente para operar com IPv6.

Roteiro Experimental

Experiência 7 - Configuração de proxy Web

1. Caso esteja utilizando a máquina virtual fornecida pelo NIC.br, pule para o passo 2:

    a. O **squid3** é um servidor HTTP bastante utilizado na Web. Para instalá-lo, basta rodar o seguinte comando no Terminal:

    ```
    $ sudo apt-get install squid3
    ```

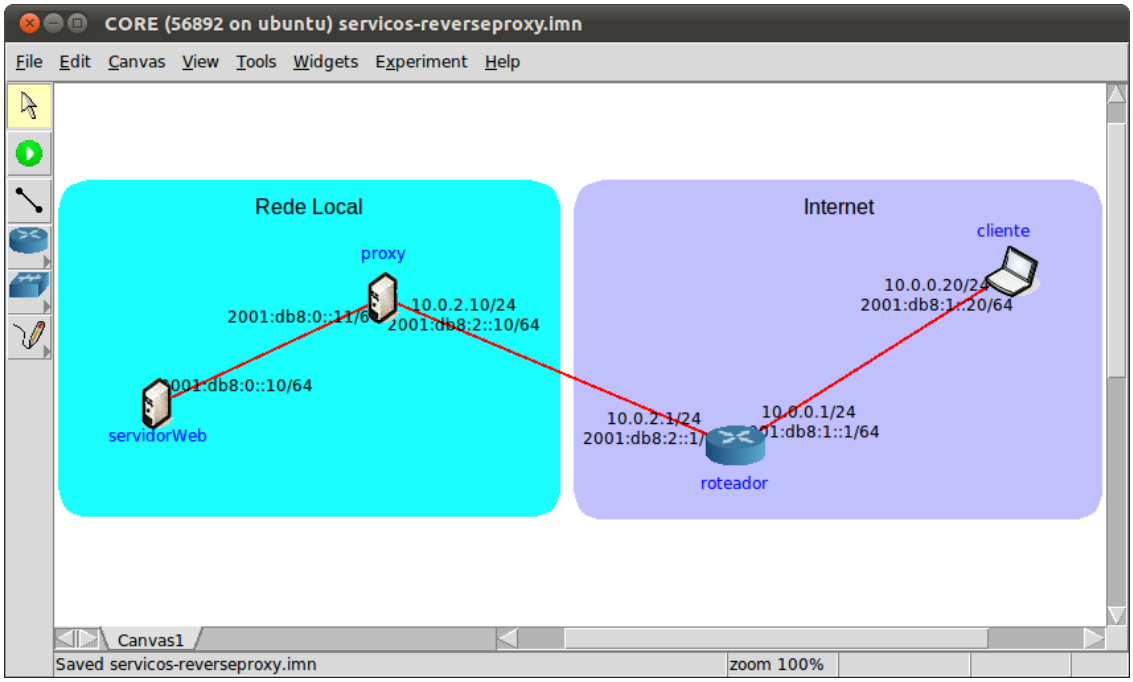
    ***Obs**: Depois do comando ‘`sudo`‘ será solicitada a senha do usuário. Caso esteja utilizando a máquina fornecida, a senha é “core”.

    b. O **apache2** é um servidor HTTP bastante utilizado na Web. Para instalá-lo, basta rodar o seguinte comando no Terminal:

    ```
    $ sudo apt-get install apache2
    ```

    ***Obs**: Depois do comando ‘`sudo`‘ será solicitada a senha do usuário. Caso esteja utilizando a máquina fornecida, a senha é “core”.

2. Inicie o CORE e abra o arquivo “**servicos-reverseproxy.imn**” localizado no diretório do desktop “/home/core/Desktop/servicos/proxyreverso”, da máquina virtual do NIC.br. A seguinte topologia deve aparecer:



Essa topologia representa uma situação em que um proxy reverso é colocado na borda de uma rede IPv6 para funcionar como cache transparente de um servidor Web e, também, para traduzir requisições IPv4 feitas ao servidor Web. O roteamento de pacotes foi configurado com a utilização de rotas estática e, a máquina 'proxy' além de suas funções básicas, funciona também como um roteador da rede local para simplificar a topologia.

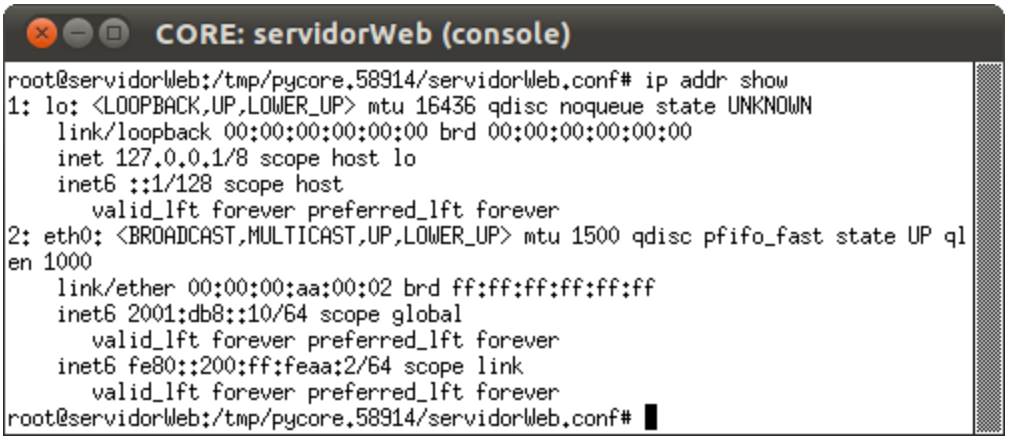
3. Verifique a configuração dos nós da topologia:
   a. Inicie a simulação realizando um dos seguintes passos:
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do 'cliente' com um duplo-clique.

   c. Verifique, com o comando abaixo, a configuração da interface de rede do 'servidorWeb' e note que ele não está configurado com endereço IPv4:

```
# ip addr show
```

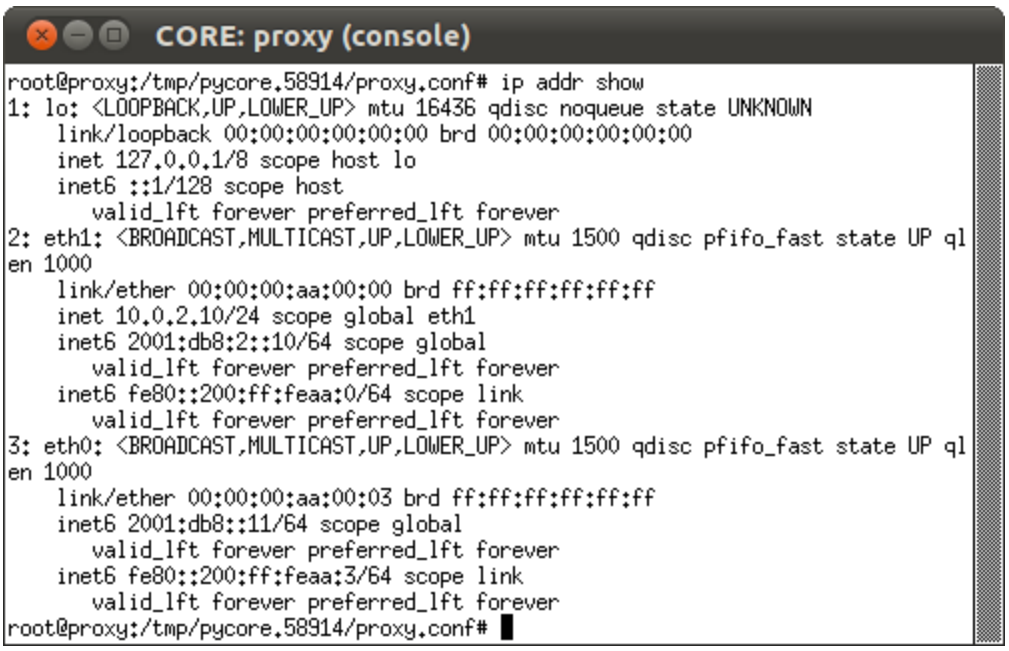
      O resultado deve ser:



***Obs**: Esse comando possibilita a verificação dos endereços das interfaces de rede da máquina.

d. Observe a configuração do 'proxy' utilizando o mesmo comando.

O resultado deve ser:



e. No terminal do 'servidorWeb' utilize o comando abaixo para verificar que o servidor Web (Apache2) está sendo executado:

```
# netstat -antup
```

O resultado deve ser semelhantes à:

4. Inicie o squid3 no 'proxy':
   a. Abra um terminal do 'proxy' com um duplo-clique.

   b. Utilize o seguinte comando para acessar as configurações do sistema squid3 que será utilizado como proxy Web nesse servidor:

```
# cat /etc/squid3/squid.conf
```

Note que o arquivo possui o seguinte conteúdo:

```
http_port 80 accel defaultsite=[2001:db8:0::10]
forwarded_for on

refresh_pattern ^ftp:           1440    20%     10080
refresh_pattern ^gopher:        1440    0%      1440
refresh_pattern .               0       20%     4320

cache_peer  2001:db8:0::10  parent  80  0  no-query  originserver
name=myAccel
acl our_sites dst 2001:db8::10
http_access allow our_sites
cache_peer_access myAccel allow our_sites
cache_peer_access myAccel deny all

acl manager proto cache_object
acl localhost src 127.0.0.1/32 ::1
acl to_localhost dst 127.0.0.0/8 0.0.0.0/32 ::1
acl SSL_ports port 443
acl Safe_ports port 21          # ftp
acl Safe_ports port 70          # gopher
acl Safe_ports port 80          # http
acl Safe_ports port 443         # https
acl Safe_ports port 210         # wais
acl Safe_ports port 1025-65535  # unregistered ports
acl Safe_ports port 280         # http-mgmt
acl Safe_ports port 488         # gss-http
```

```
acl Safe_ports port 591          # filemaker
acl Safe_ports port 777          # multiling http
acl CONNECT method CONNECT
http_access allow manager all
http_access allow manager
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access deny all
access_log /var/log/squid3/access.log squid
coredump_dir /var/spool/squid3
```

Verifique o conteúdo de algumas linhas desse arquivo:

```
http_port 80 accel defaultsite=[2001:db8:0::10]
forwarded_for on
```

Para que o squid funcione como um proxy transparente, ele deve escutar requisições na porta 80 como um servidor Web. Para isso duas linhas são adicionadas. A primeira linha informa a configuração de dominio padrão que o proxy deve reconhecer. No caso, o endereço IPv6 do 'servidorWeb' foi configurado pois o cenário não implementa nenhum servidor DNS. Já segunda linha indica que o endereço do requisitante deve ser repassado ao servidor Web para que este consiga manter um registro correto dos acessos.

```
cache_peer 2001:db8:0::10 parent 80 0 no-query originserver
name=myAccel
```

Essa linha configura o endereço do 'servidorWeb' no cache do squid. Caso existam mais servidores Web na rede local, eles devem ser configurados de forma semelhante no squid.

```
acl our_sites dst 2001:db8::10
http_access allow our_sites
cache_peer_access myAccel allow our_sites
cache_peer_access myAccel deny all
```

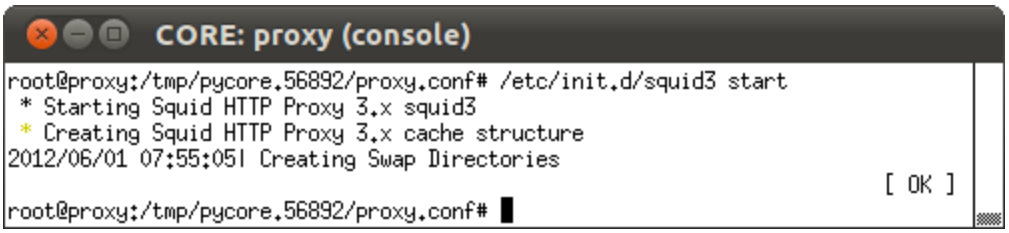
Com esse grupo de comando o squid é pré configurado para aceitar apenas os registros cadastrados, como "`our_sites`". No caso, foi utilizado diretamente os endereços IP do 'servidorWeb', porém normalmente o nome do Web site poderia ser utilizado diretamente com a substituição da opção "`dst`" por "`dstdomain`".

O restante dos comandos faz parte da configuração padrão indicada para servidores squid.

c. Utilize o comando a seguir para iniciar o serviço squid:

```
# /etc/init.d/squid3 start
```

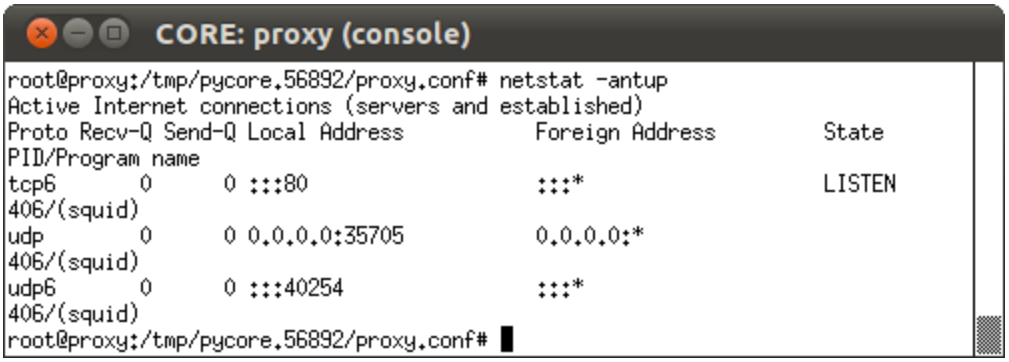
O resultado deve ser:



d. Confirme que o serviço está escutando a porta 80 com o seguinte comando:

```
# netstat -antup
```

O resultado deve ser similar à:

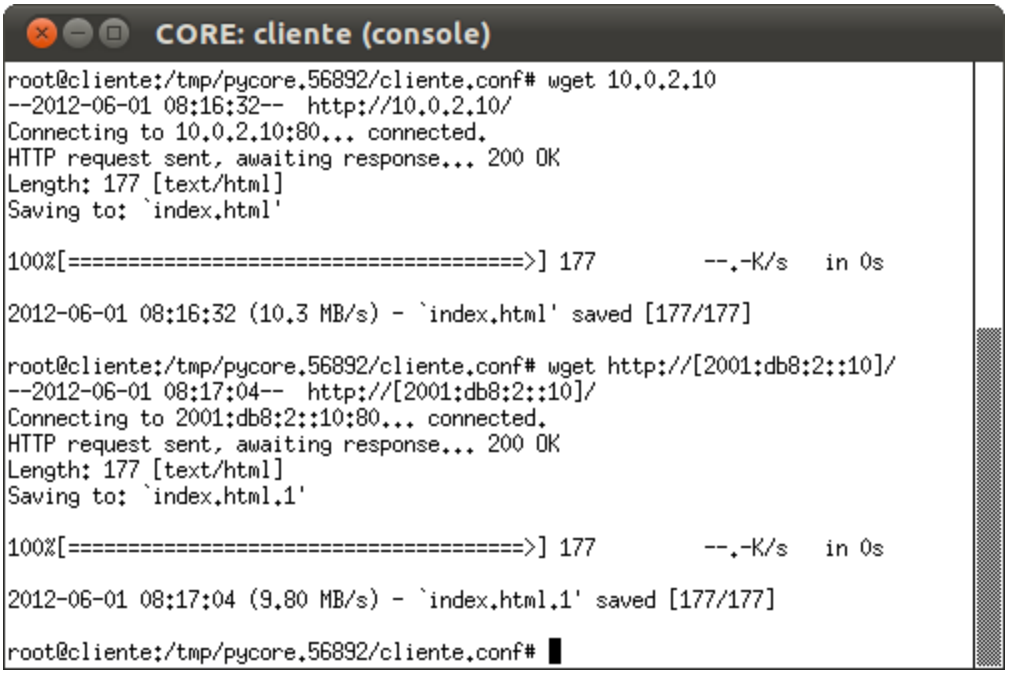


***OBS**: Note que as portas UDP, são escolhidas aleatóriamente por padrão. Na prática, elas são utilizadas com os módulos ICP, HTCP e DNS do squid3.

5. Verifique o funcionamento da solução:
   a. Abra, com um duplo-clique, o terminal do 'cliente' acesse o 'servidorWeb' via proxy tanto em IPv4 quanto em IPv6:

```
# wget 10.0.2.10
# wget http://[2001:db8:2::10]/
```

O resultado deve ser:

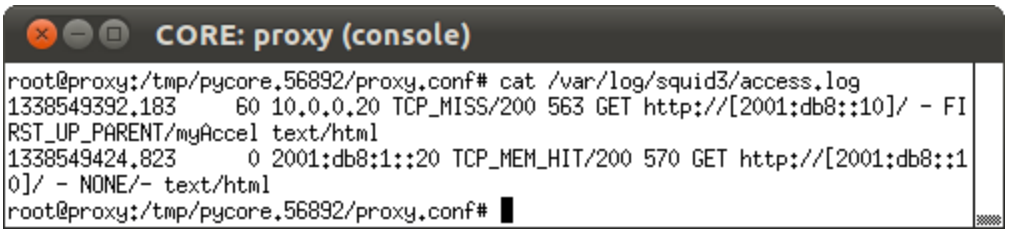


***Obs**: Normalmente, o acesso ao site Web utilizaria o nome de domínio e, o endereço acessado dependeria do registro no DNS. Nesse caso, tal registro deveria apontar para o endereço do proxy ao invés do endereço IP do Web site.

b. No terminal do 'proxy', acesse o arquivo de log do *squid3* e verifique que o acesso ao 'servidorWeb' foi realizado por ele:

```
# cat /var/log/squid3/access.log
```

O resultado deve ser similar à:



***Obs**: Note que esse arquivo apresenta as requisiçoes feitas pelo cliente (`10.0.0.20`, `2001:db8:1::20`) para o servidor (`2001:db8::10`) através do proxy.

c. Abra, com um duplo clique, o terminal do 'servidorWeb' e verifique que apenas um acesso foi realizado, já que o segundo foi respondido diretamente pelo cache:


```
# cat /var/log/apache2/access.log
```

O resultado deve ser similar à:

***Obs**: a primeira requisição enviada para o cliente pode ser visualizada na linha com a expressão “GET / HTTP/1.1”.

6. Encerre a simulação do experimento:
   a. Execute o seguinte comando num terminal do ‘proxy’ para parar a execução do serviço *squid3*:

```
# /etc/init.d/squid3 stop
```

   b. Pare a simulção do CORE :
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Stop.

IPV6 - Serviços IPv6 - Samba

Objetivo

Esta experiência tem o objetivo de mostrar a configuração básica de um serviço samba com compartilhamento arquivos, utilizando como base o protocolo IPv6.

Para a realização do presente exercicio será utilizada a topologia descrita no arquivo: **servicos-samba1.imn**.

Introdução Teórica

Samba é um software livre que permite compartilhar e gerenciar recursos de um servidor por dispositivos dotados de diferentes sistemas operacionais. Dentre os recursos disponibilizados, o sistema de arquivos e o sistema de impressoras são os mais utilizados. Sua arquitetura é cliente-servidor e a comunicação segue os protocolos SMB (*Server Message Block*) e CIFS (*Common Internet File System*).

Este experimento visa configurar um servidor samba que disponibilize o acesso a uma pasta compartilhada à um cliente, utilizando comunicação apenas em IPv6. Desde a versão 3.2.0 do samba existe suporte ao IPv6 tanto nas ferramentas de servidor como nas de cliente. Contudo, os endereços precisam ser configurados corretamente para que não haja problemas na comunicação.

Roteiro Experimental

Experiência 8 - Samba

1. Caso esteja utilizando a máquina virtual fornecida pelo NIC.br, pule para o passo 2:

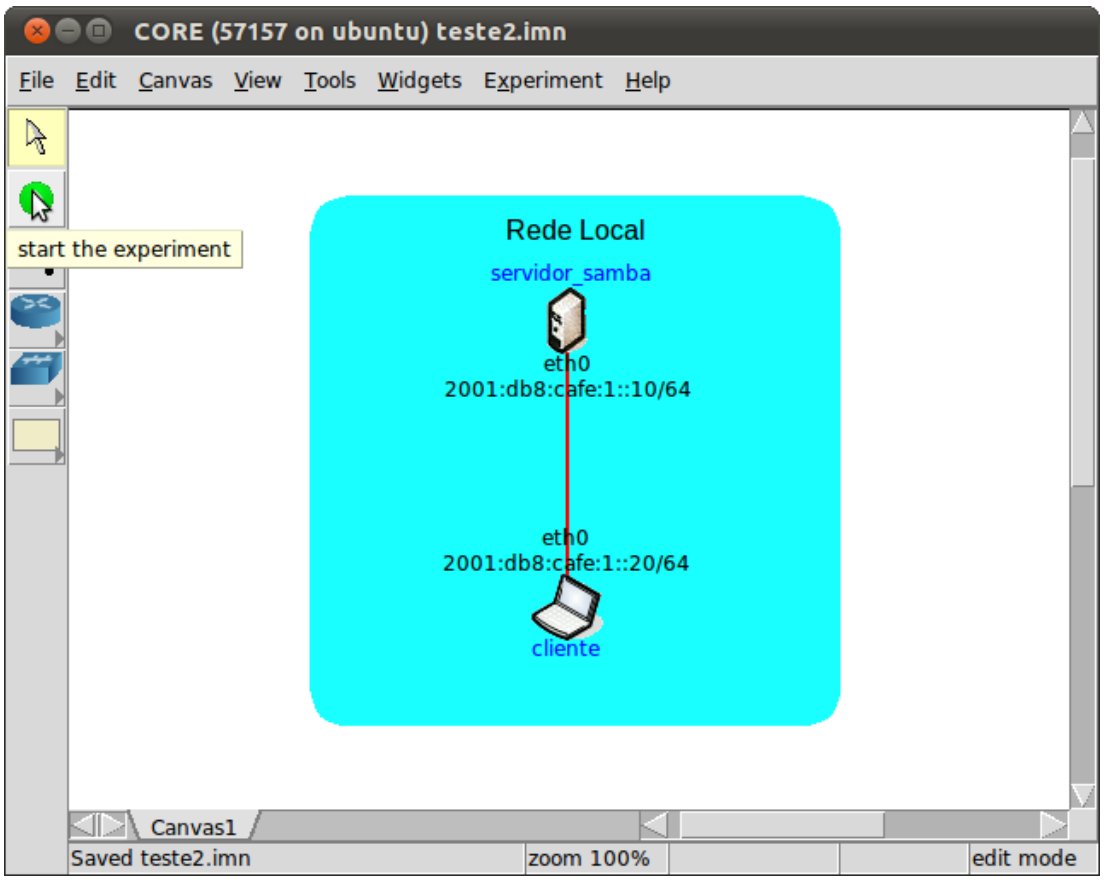
    a. O **smbd** é um servidor que provém serviços SMB/CIFS aos clientes. Para instalá-lo, basta rodar o seguinte comando no Terminal:

    ```
    $ sudo apt-get install samba
    ```

    b. O **cifs-utils** é um pacote que contêm o sistema de arquivos CIFS. Para instalá-lo, basta rodar o seguinte comando no Terminal:

    ```
    $ sudo apt-get install cifs-utils
    ```

2. Inicie o CORE e abra o arquivo “**servicos-samba1.imn**” localizado no diretório /home/core/Desktop/servicos/samba, da máquina virtual do NIC.br. A seguinte topologia deve aparecer:



3. Analise a topologia, observando os endereços, e comece o experimento:
   a. Inicie a simulação utilizando um dos seguintes passos:
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Start.

4. Prepare o ambiente no servidor para rodar o serviço samba adequadamente.
   a. Abra o terminal do ‘servidor_samba’ com um duplo-clique.

   b. Crie um usuário de nome ‘teste’ com a senha ‘tt’ para permitir o acesso de clientes à máquina servidora. Utilize o comando:

```
# useradd -g users -p tt teste
```

O resultado deve ser:

b. Crie a pasta que será acessada pelo cliente, com o nome 'exemplo', no caminho '/home/core/samba', através do seguinte comando:

```
# mkdir /home/core/samba/exemplo
```

O resultado deve ser:

c. Crie, dentro dessa pasta, um arquivo chamado 'samba-teste.txt', utlizando o seguinte comando:

```
# touch /home/core/samba/exemplo/samba-teste.txt
```

O resultado deve ser:

d. Mude as permissões de acesso da pasta e do arquivo criado para permitir que o usuário criado consiga acessa-las sem nenhum problema. Utilize o comando:

```
# chown -R teste:users /home/core/samba/exemplo
```

O resultado deve ser:

5. Configure o servidor samba compartilhar uma pasta na rede com acesso habilitado apenas para o usuário criado.

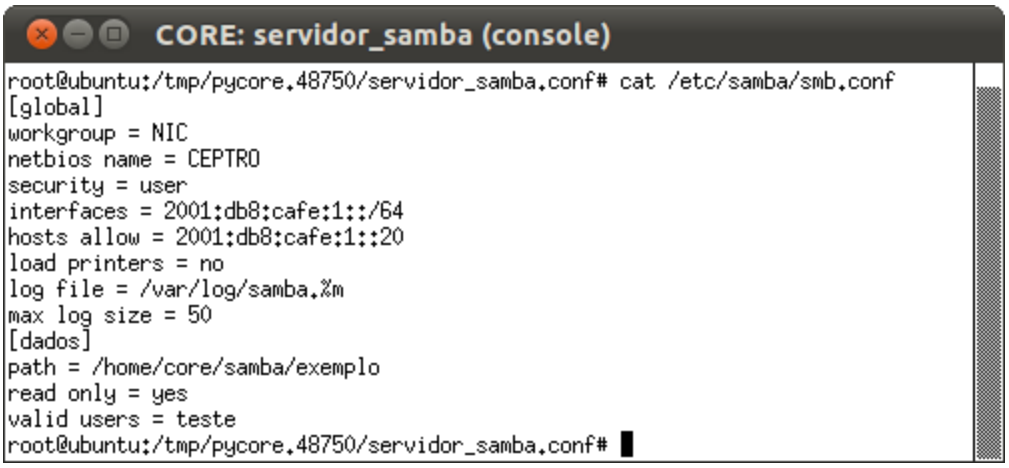
a. Configure o servidor *samba* para aceitar requisições via rede IPv6. Para isso, utilize o editor de texto *nano* para modificar o arquivo "`/etc/samba/smb.conf`":

```
# nano /etc/samba/smb.conf
```

Escreva as seguintes linhas no arquivo:

```
[global]
workgroup = NIC
netbios name = CEPTRO
security = user
interfaces = 2001:db8:cafe:1::/64
hosts allow = 2001:db8:cafe:1::20
load printers = no
log file = /var/log/samba.%m
max log size = 50
[dados]
path = /home/core/samba/exemplo
read only = yes
valid users = teste
```

O resultado deve ser:

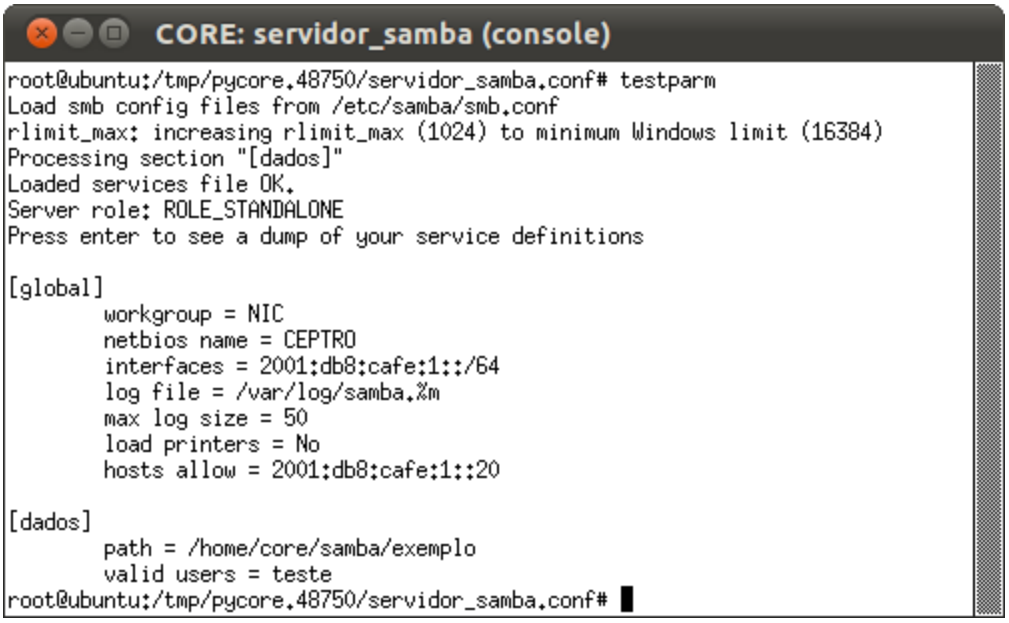


***Obs**: nesse arquivo há duas configurações utilizando endereços IPv6. O campo `interfaces` indica quais serão as interfaces de rede que aceitarão conexões de clientes e, o campo `hosts allow`, lista os endereços com permissão de acesso ao serviço.

b. Para validar e gerar uma versão otimizada das configurações, utiliza-se o comando:

```
# testparm
```

O resultado deve ser:



c. Digite o seguinte comando para que usuário 'teste' criado anteriormente seja adicionado também às configurações do samba (utilize a mesma senha 'tt') :

```
# smbpasswd -a teste
```

O resultado deve ser:

d. Inicie o serviço do servidor samba com o seguinte comando:

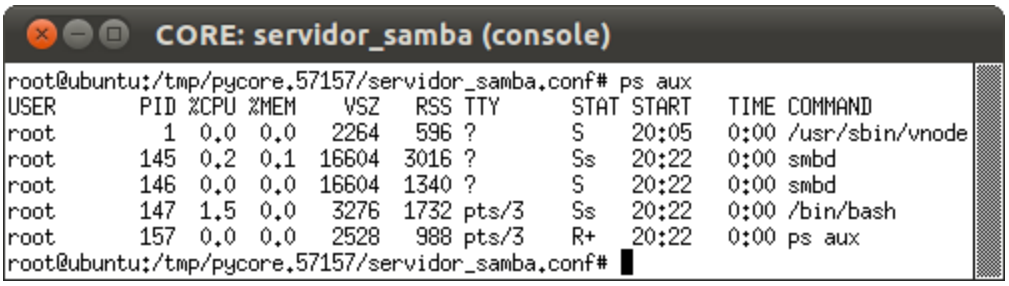
```
# smbd
```

O resultado deve ser:

e. Verifique se o serviço esta rodando com o comando:

```
# ps aux
```

O resultado deve ser parecido com:



***Obs**: Note que as linhas cotendo a palavra smbd, provam que o serviço samba está ativo.

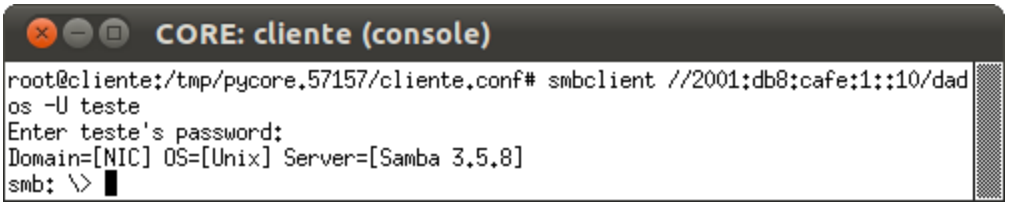
6. Configure o 'cliente' para se conectar ao serviço samba estabelecido:
   a. Abra um terminal do 'cliente' com um duplo-clique.

   b. Inicie o serviço '`smbclient`' para acessar a máquina servidora com o comando:

```
# smbclient //2001:db8:cafe:1::10/dados -U teste
```

***Obs**: Quando a senha do usuário 'teste' for solicitada, digite 'tt'.

O resultado deve ser:

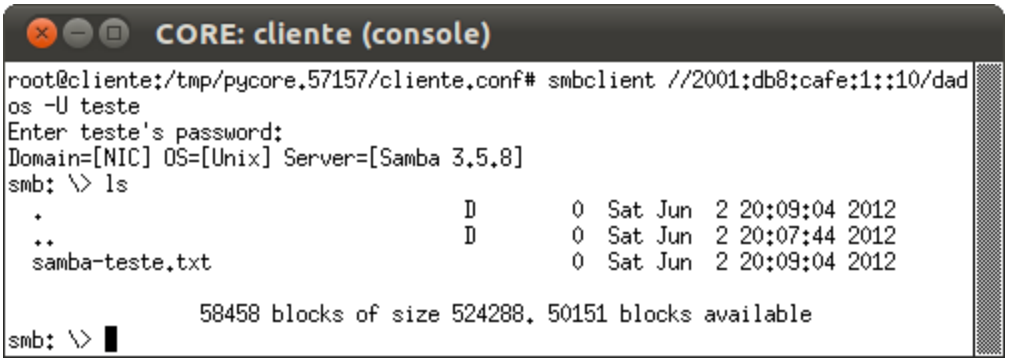


***Obs**: Para conhecer mais comando nesse terminal do samba digite 'help'.

c. Verifique que o arquivo "`samba-teste.txt`" criado anteriormente na máquina 'servidor_samba' pode ser acessado:

```
smb: \> ls
```

O resultado deve ser:



Digite o comando '`exit`' ou Ctrl+D para sair do terminal samba.

d. Outra maneira que pode ser utilizada para acessar o 'servidor_samba' é através de um mapeamento fixo de um drive de rede. Para isso crie uma pasta 'mnt' no cliente com o comando:

```
# mkdir mnt
```

O resultado deve ser:

e. Digite o comando para montar o mapeamento fixo:

```
# mount.cifs //2001:db8:cafe:1::10/dados -o user=teste,password=tt
./mnt/
```

O resultado deve ser:

f. Entre na pasta mnt e observe o conteúdo dela com os comandos:

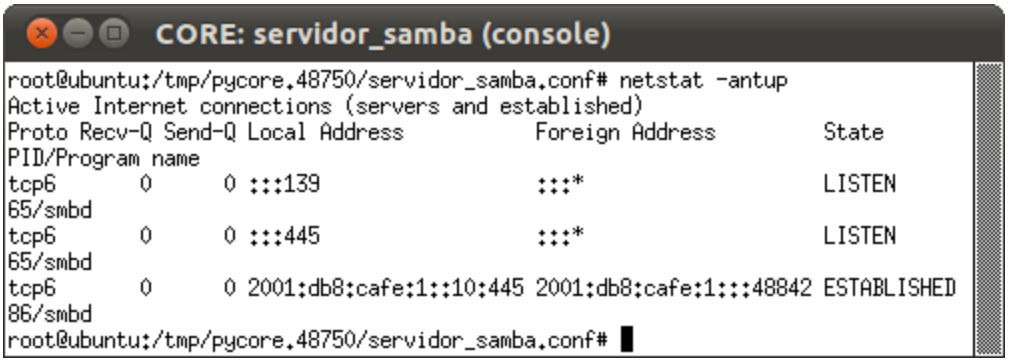
```
# cd mnt/
# ls
```

O resultado deve ser:

g. No terminal do 'servidor_samba' utilize o comando abaixo para verificar que a conexão entre as duas máquinas foi estabelecida através do samba:

```
# netstat -antup
```

O resultado deve ser:



***Obs**: Observe a conexão no endereço IPv6 do servidor (2001:db8:cafe:1::10) na porta do samba (445). A porta utilizada pelo cliente pode ser diferente de 48842.

7. Encerre a simulação do experimento:
   a. Abra um terminal do 'cliente' com um duplo-clique.
   b. Desmonte o mapeamento entre as pastas com o comando:

```
# umount mnt/
```

O resultado deve ser:

c. No terminal do “servidor_samba”, remova o cliente do samba com o comando:

```
# smbpasswd -x teste
```

O resultado deve ser:

d. Remova também o cliente da máquina virtual com o comando:

```
# userdel teste
```

O resultado deve ser:

e. Finalize a simulação com um dos dois comandos:

i. aperte o botão 
ii. utilize o menu Experiment > Stop.

IPv6 - Laboratório de Ataques DoS ao Neighbor Discovery

Objetivo

Esse laboratório tem o objetivo mostrar o funcionamento de ataque DoS (*Denial of Service*) ao protocolo *Neighbor Discovery Protocol* (NDP).

Para o presente exercício serão utilizadas as topologias descrita nos arquivos:
seguranca-dos-na.imn

Introdução Teórica

NDP é um protocolo desenvolvido com a finalidade de gerir dois processos que trabalham com o relacionamento entre nós vizinhos em uma rede.

O primeiro deles é o processo de autoconfiguração das interfaces dos dispositivos pertencentes a uma rede. Para isso é necessário a realização de 3 tarefas:

- *Parameter Discovery*, que atua na descoberta de informações sobre o enlace, como MTU ou hop limit.
- *Address Autoconfiguration*, que trabalha com autoconfiguração de um endereço em uma interface de certo nó.
- *Duplicate Address Detection*, que opera em um nó com o propósito de descobrir se o endereço que se deseja configurar, já está sendo utilizado por um outro nó.

O segundo, é o de transmissão de pacotes por um nó a um determinado destino. Para isso, é necessário a realização de 6 tarefas:

- *Router Discovery*, que trabalha com a descoberta de roteadores pertencentes ao enlace.
- *Prefix Discovery*, que implementa a descoberta de prefixos de redes, necessária para a decisão das rotas de transmissão de pacotes, ou seja, para decidir se eles devem ser enviados a um roteador específico ou direto para outro dispositivo do mesmo enlace.
- *Address Resolution*, que serve para descobrir o endereço físico dos equipamentos através de seus endereços IPv6.
- *Neighbor Unreachability Detection*, que permite aos nós descobrir se um vizinho é alcançavel.
- *Redirect*, que permite ao roteador informar à um nó uma rota melhor à ser utilizada quando este envia pacotes a determinado destino.
- *Next-hop Determination*, um algoritmo para mapear um endereço IP de destino em um endereço IP de um dispositivo vizinho para onde o tráfego deve ser transmitido.

Roteiro Experimental

Experiência 1 - DoS com NA (*Neighbor Advertisement*)

1. Se você estiver utilizando a máquina virtual fornecida pule para o passo 3. Para fazer algumas verificações durante o experimento, também, será necessária a utilização do programa Wireshark que auxilia numa melhor visualização de pacotes transmitidos na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

   Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

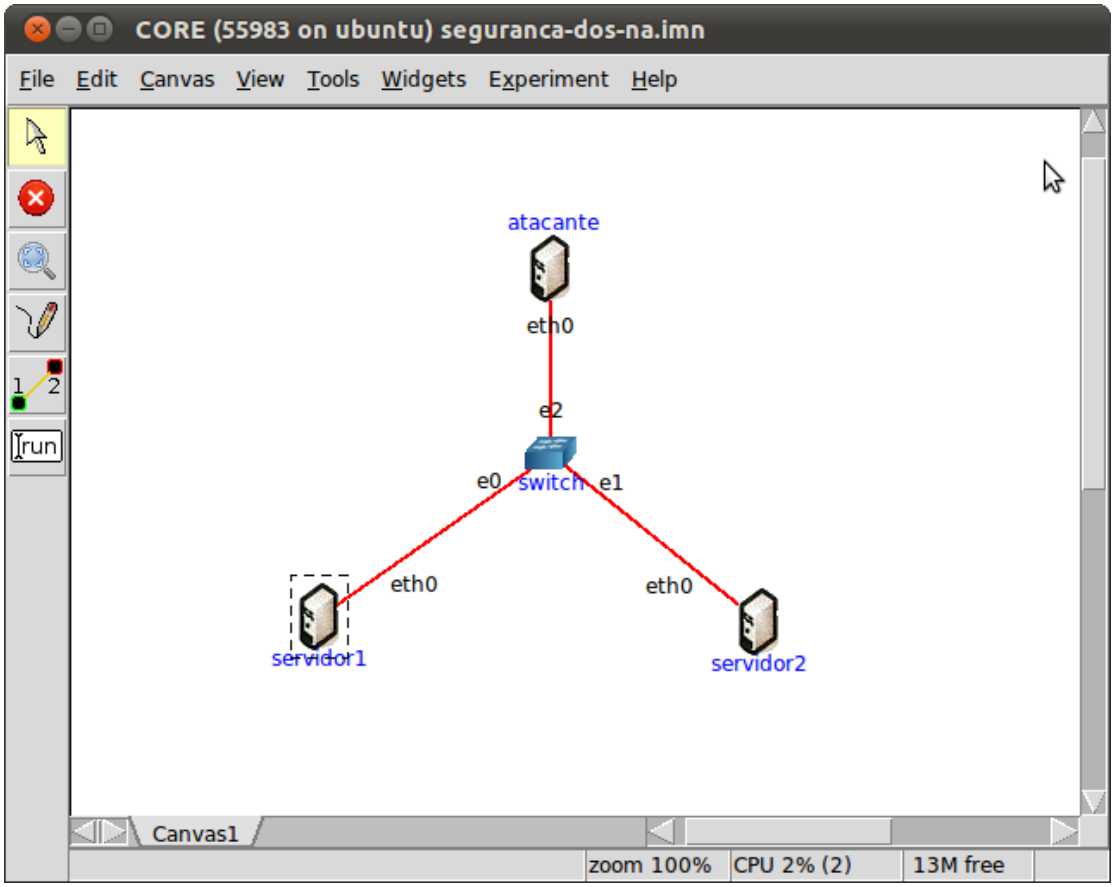
2. Para fazer o ataque será necessário utilizar as ferramentas THC-IPv6, disponíveis em:
   http://www.thc.org/thc-ipv6/
   Descompacte o pacote e execute os comandos abaixo, na pasta descompactada, para instalar. A senha quando solicitada é “core”.

```
$ sudo make
$ sudo make install
```

3. Para fazer a detecção de ataques é necessário instalar o NDPMON, disponível em:
   http://ndpmon.sourceforge.net/download.html
   Descompacte o pacote e execute os comandos abaixo, na pasta descompactada, para instalar. A senha quando solicitada é “core”.

```
$ sudo apt-get install autoconf
$ sudo apt-get install libxml2-dev
$ sudo autoconf
$ sudo ./configure
$ sudo make
$ sudo make install
```

4. Inicie o CORE e abra o arquivo “seguranca-dos-na.imn” localizado no diretório do desktop “Segurança/DoS”, da máquina virtual fornecida pelo NIC.br. A seguinte topologia de rede deve aparecer:



Seu objetivo é representar uma rede local onde um ataque de DoS baseado no envio de falsos de NA (*Neighbor Advertisement*) possa ser simulado. Nessa situação, a máquina 'atacante' responde a todos as requisições NS (*Neighbor Solicitation*) informando ser a dona do endereço IP em questão. Este comportamento impede que novas máquinas que tenham se conectado na rede e enviado pacotes NS consigam utilizar qualquer endereço IP e por consequência, faz com que esses novos dispositivos fiquem sem acesso à Internet.

5. Verifique a configuração dos nós da topologia:

a. Inicie a simulação:

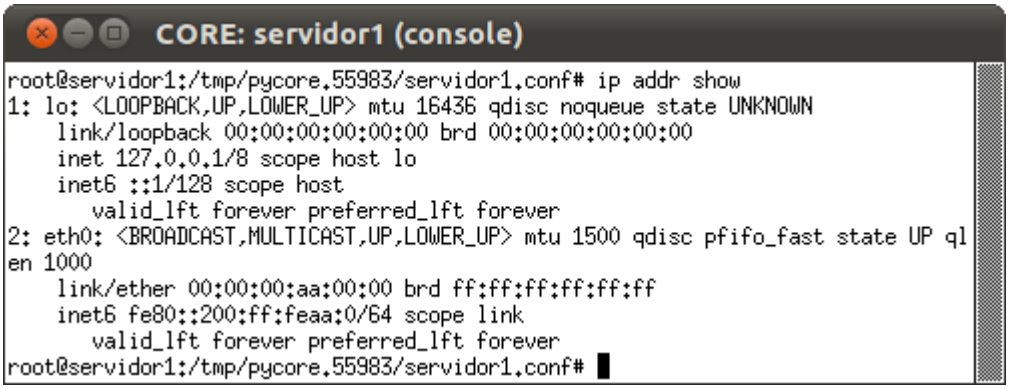
i. aperte o botão  ; ou

ii. utilize o menu Experiment > Start.

b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do 'servidor1', através de um duplo-clique, e utilize o seguinte comando para verificar suas configurações das interfaces de rede:

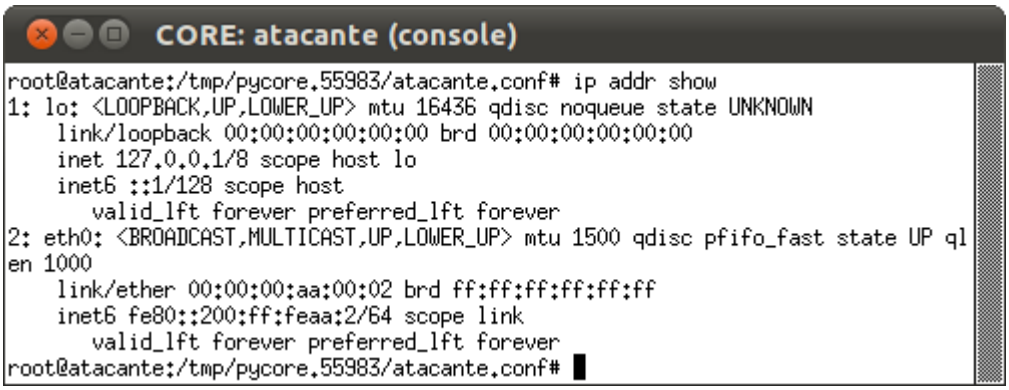
```
# ip addr show
```

O resultado deve ser:



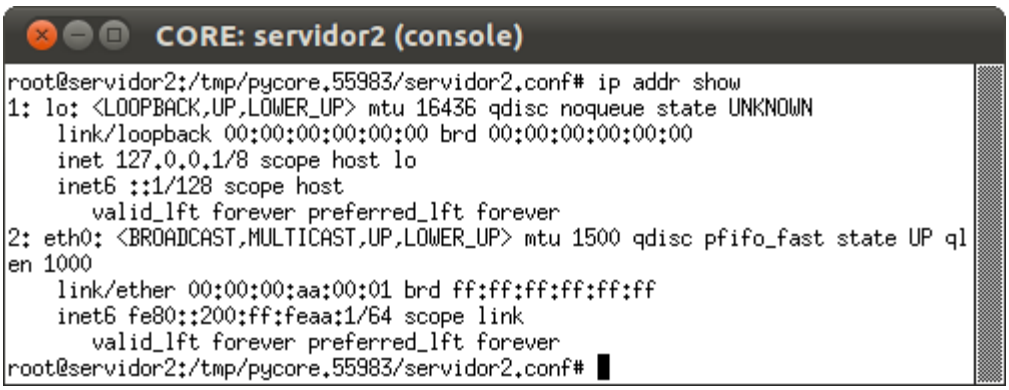
c. Abra o terminal do 'atacante' através de um duplo-clique e utilize o mesmo comando para visualizar as configurações das interfaces de rede.

O resultado deve ser:



d. Abra o terminal do 'servidor2' através de um duplo-clique e utilize o mesmo comando para visualizar as configurações das interfaces de rede.

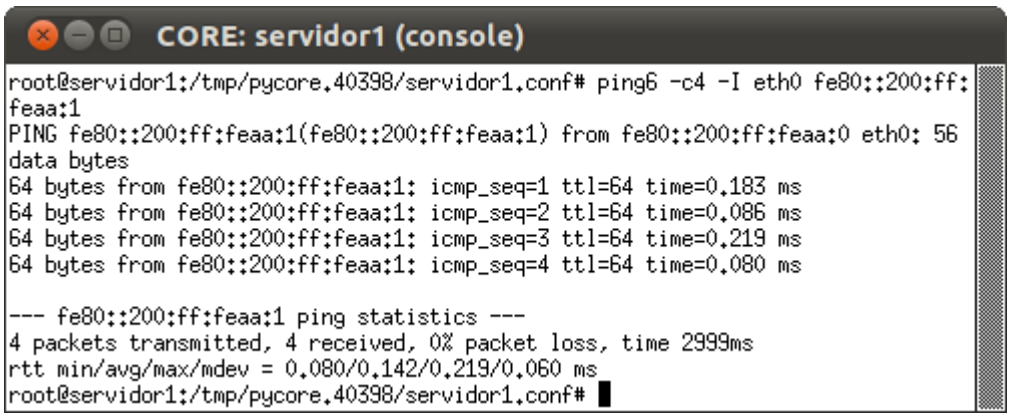
O resultado deve ser:



6. Abra o terminal do ‘servidor1’, através de um duplo-clique, e execute um ping6 para o ‘servidor2’:

```
$ ping6 -c 4 -I eth0 fe80::200:ff:feaa:1
```

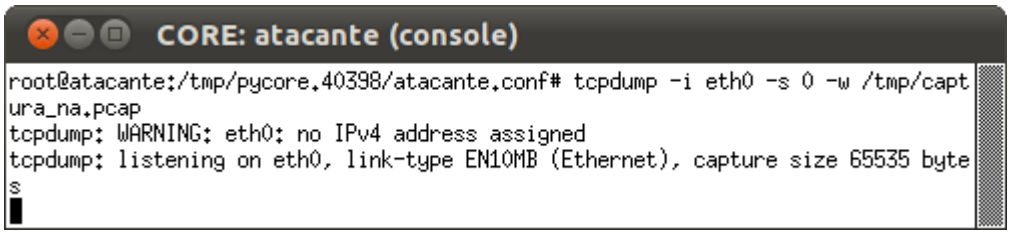
O resultado deve ser:



7. Abra um terminal do ‘atacante’. Utilize o seguinte comando para iniciar uma captura de pacotes na interface eth0:

```
$ tcpdump -i eth0 -s 0 -w /tmp/captura_na.pcap
```

O resultado deve ser:



Essa janela de terminal deve ser mantida aberta.

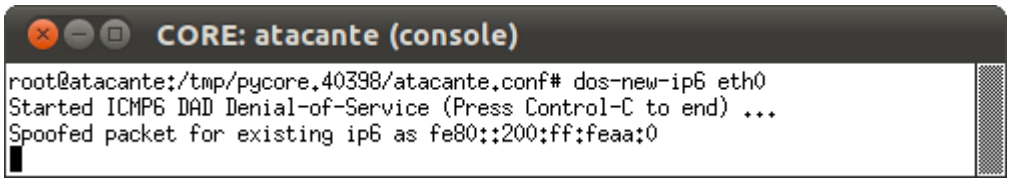
8. Abra um novo terminal no ‘atacante’ sem interromper a captura que está em execução no outro terminal. Execute o programa dos-new-ip6:

```
$ dos-new-ip6 eth0
```

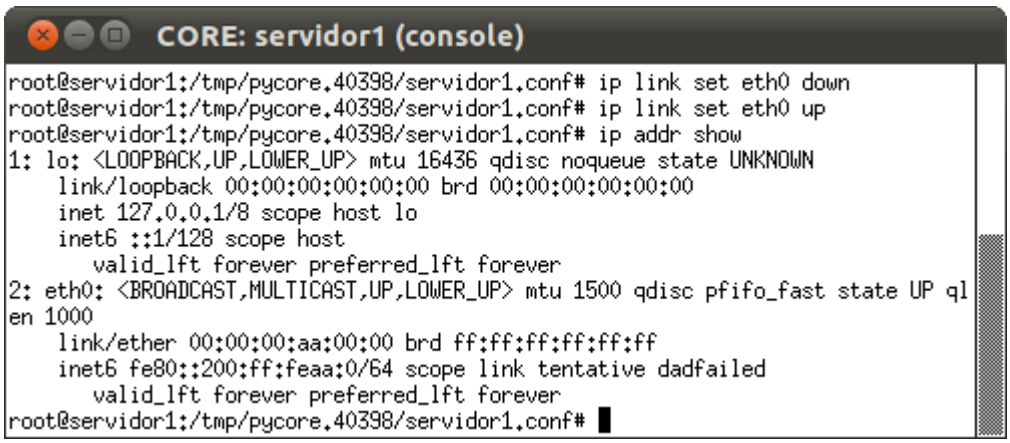
9. No terminal do ‘servidor1’ desative e ative a interface eth0, para que ela realize o processo de obtenção de endereço IPv6 de link local.

```
$ ip link set eth0 down
$ ip link set eth0 up
$ ip addr show
```

O resultado no terminal do 'atacante' rodando o dos-new-ip6 deve ser:



O resultado no terminal 'servidor1', deve mostrar a falha em obter um IPv6 de link local conforme a figura:



10. Para confirmar que não foi possível obter o endereço de link local, repita no 'servidor1' o ping6 do começo da experiência:

```
$ ping6 -c 4 -I eth0 fe80::200:ff:feaa:1
```

O resultado deve ser:

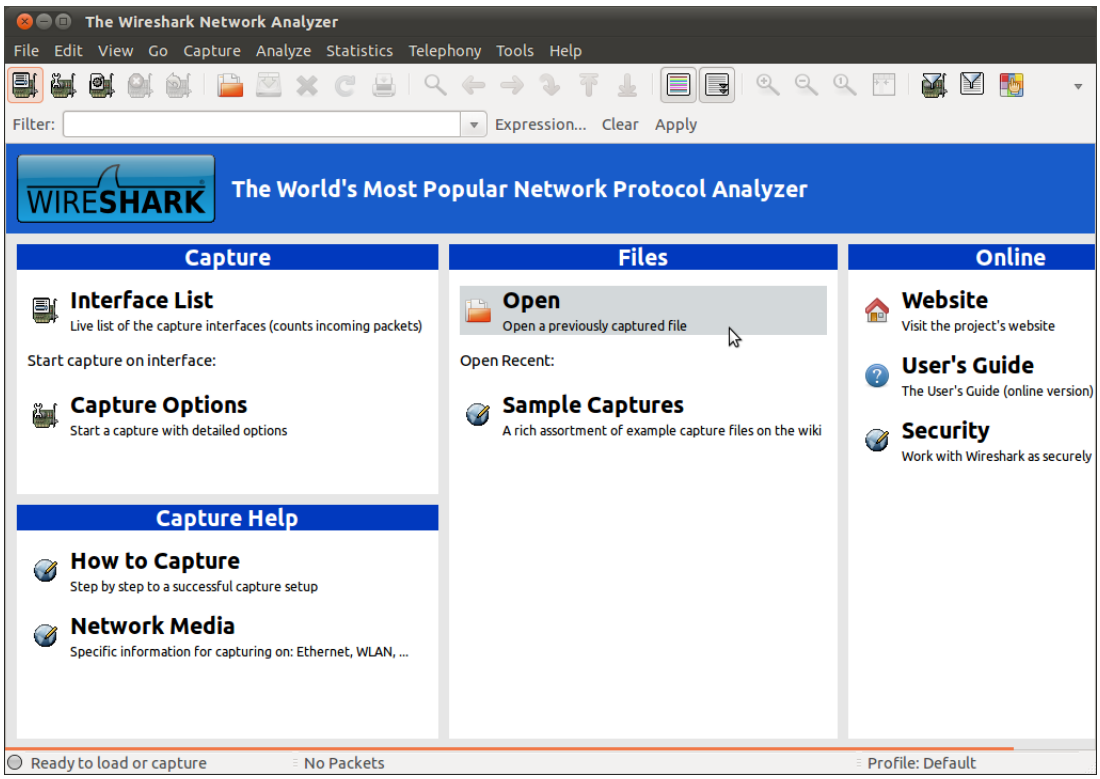
11. Pare a captura de pacotes no primeiro terminal do 'atacante' usando Ctrl+C.

12. Encerre a simulação:

- **a.** aperte o botão  ; ou
- **b.** utilize o menu Experiment > Stop.

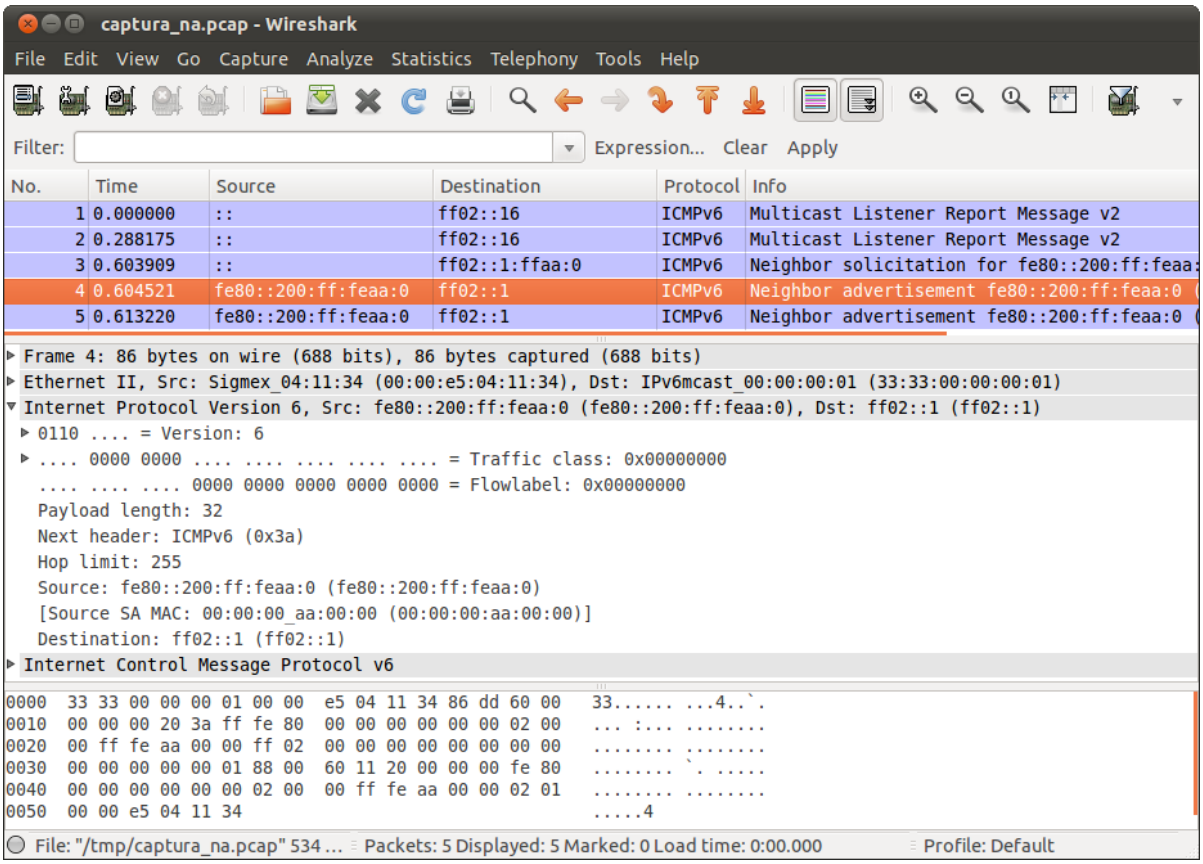
13. Para verificar os pacotes capturados, utilize o programa Wireshark. Uma maneira é através de um terminal na máquina virtual com o comando:

```
$ wireshark
```



Esse programa tem como principais funcionalidades a captura e análise de pacotes transmitidos por uma interface de rede. Através de seu uso, é possível melhor visualizar os pacotes que trafegam pela rede. Verifique o arquivo de captura previamente obtido.

a. Abra o arquivo `/tmp/captura_na.pcap` com o menu File>Open:



b. Analise os pacotes que chegaram e saíram da eth0 do 'atacante' e note que um pacote de *Neighbor Solicitation* foi enviado pelo 'servidor1' recebido pelo 'atacante' que, na sequência, enviou pacotes forjados de *Neighbor Advertisement* para informar ele que já está utilizando o endereço IP (fe80::200:ff:feaa:0), mesmo que o endereço de sua interface seja (fe80::200:ff:feaa:2) como visto no item 4.c.

c. Este comportamento irregular gera um ataque de DoS (*Denial of Service*) pois impede que qualquer novo dispositivo consiga ativar sua interface de comunicação.

14. Este ataque poderia ser evitado com a utilização do SeND (*Secure Neighbor Discovery*), conforme detalhado na apostila teórica, mas não foi possível encontrar uma implementação funcional para Linux. Assim sendo, o próximo passo consiste em executar uma ferramenta que detecte comportamentos estranhos gerando o log destes pacotes, auxiliando na detecção de ataques e na identificação da máquina atacante.

15. Volte a janela do CORE:

a. Inicie a simulação:

i. aperte o botão  ; ou

ii. utilize o menu Experiment > Start.

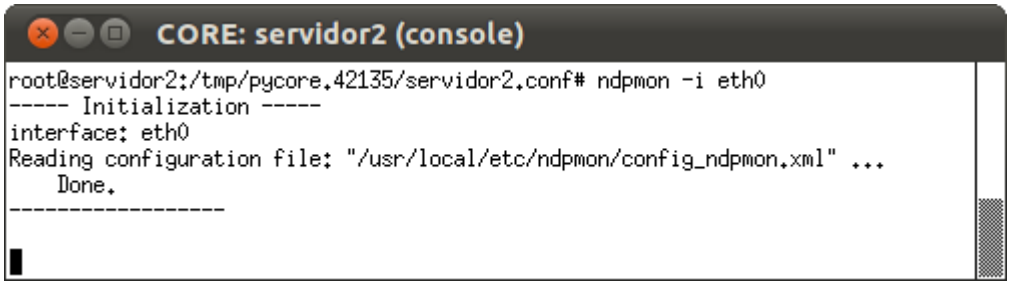
b. Espere até que o CORE termine a inicialização da simulação e abra o terminal um terminal no ‘atacante’. Execute o programa dos-new-ip6:

```
$ dos-new-ip6 eth0
```

16. Abra o terminal do ‘servidor2’ e execute o programa NDPMON.

```
$ ndpmon -i eth0
```

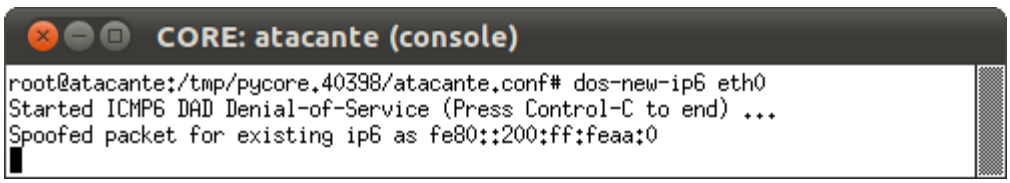
O resultado deve ser:



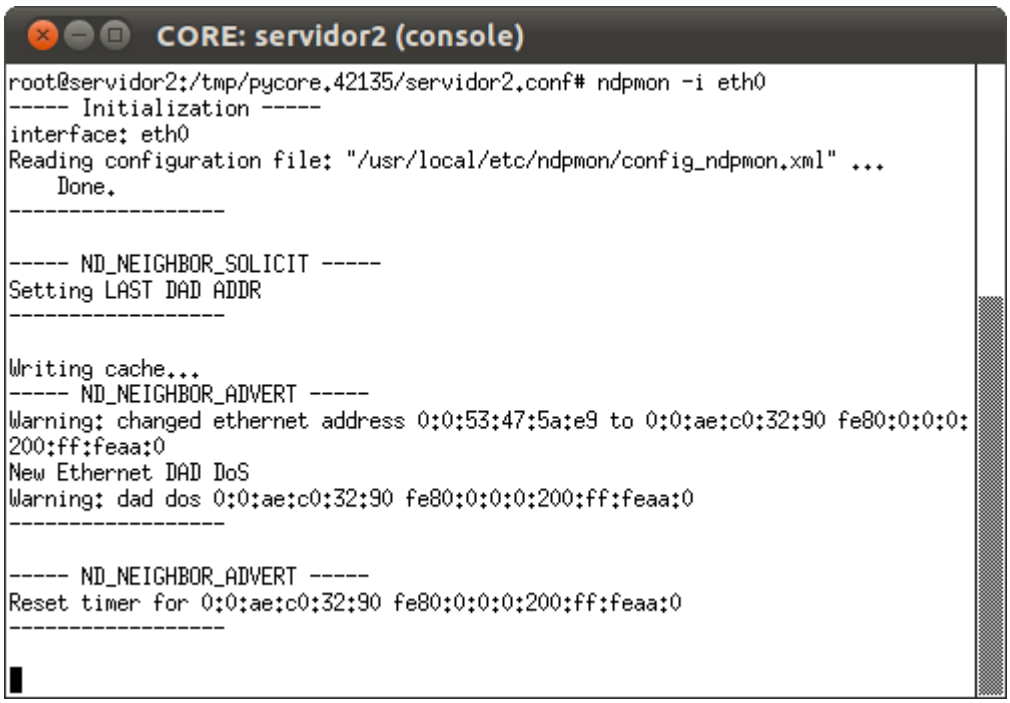
17. Abra o terminal do ‘servidor1’ e desative e ative a interface eth0, para que ela realize o processo de obtenção de endereço IPv6 de link local.

```
$ ip link set eth0 down
$ ip link set eth0 up
$ ip addr show
```

O resultado no terminal do ‘atacante’ rodando o dos-new-ip6 deve ser:



O resultado no terminal do 'servidor2' deve indicar que ocorreu um ataque de negação de serviço no processo de detecção de endereço duplicado (New Ethernet DAD DoS - Warning: dad dos) indicando o endereço IPv6 atacado, conforme a figura abaixo:



18. O NDPMON pode ser configurado para enviar emails no caso de detecção de ataques permitindo uma rápida ação dos administradores de rede para mitigar os problemas causados. O software também faz o log dos eventos ocorridos, sendo este a primeira fonte de informação na tentativa de encontrar o atacante.

19. Encerre o experimento parando a simulação:

- **c.** aperte o botão  ; ou
- **d.** utilize o menu Experiment > Stop.

IPv6 - Laboratório de firewall

Objetivo

Este laboratório tem como objetivo guiar a configuração de dois tipos de firewall IPv6:

1. Firewall para servidor; e
2. Firewall para máquinas no meio do caminho (roteador).

Verificaremos que, ao contrário do IPv4, é necessário responder e permitir a passagem de pacotes ICMPv6, (*Internet Control Message Protocol version 6*), necessários para o correto funcionamento de redes IPv6.

Para o presente exercício será utilizada a topologia descrita no arquivo: **seguranca-firewall.imn**.

Introdução Teórica

O ICMPv6 é um tipo de mensagem que teve como base o protocolo ICMPv4, desenvolvido para utilização em conjunto com o IPv6 como parte substancial de sua arquitetura. O uso do protocolo é obrigatório em todos os nós da rede que utilizam IPv6.

A nova versão de ICMP também executa funções que eram exercidas por outros protocolos no IPv4. Essa mudança possui como principal objetivo reduzir a quantidade de protocolos utilizados e, assim, aumentar a coerência e diminuir o tamanho de suas implementações.

As seguintes funcionalidades são agregadas:

- **ARP (*Address Resolution Protocol*)**, cujo o objetivo é mapear endereços da camada de enlace para a camada IP.
- **RARP (*Reverse Address Resolution Protocol*)**, que realiza o inverso do ARP, mapeando os endereços IP para a camada de enlace.
- **IGMP (*Internet Group Management Protocol*)**, que gerencia membros de grupos multicast.

Ou seja, no IPv6 não existem os protocolos ARP, RARP e IGMP, pois, todas as suas funções foram integradas ao ICMPv6. É importante notar que o ARP e RARP, no IPv4, podem ser descritos como protocolos que operam entre as camadas 2 e 3 do modelo ISO/OSI (*Open Systems Interconnection*) e não dependem de pacotes IP. Já o ICMPv6 é um protocolo de camada de rede que é encapsulado em um pacote IP.

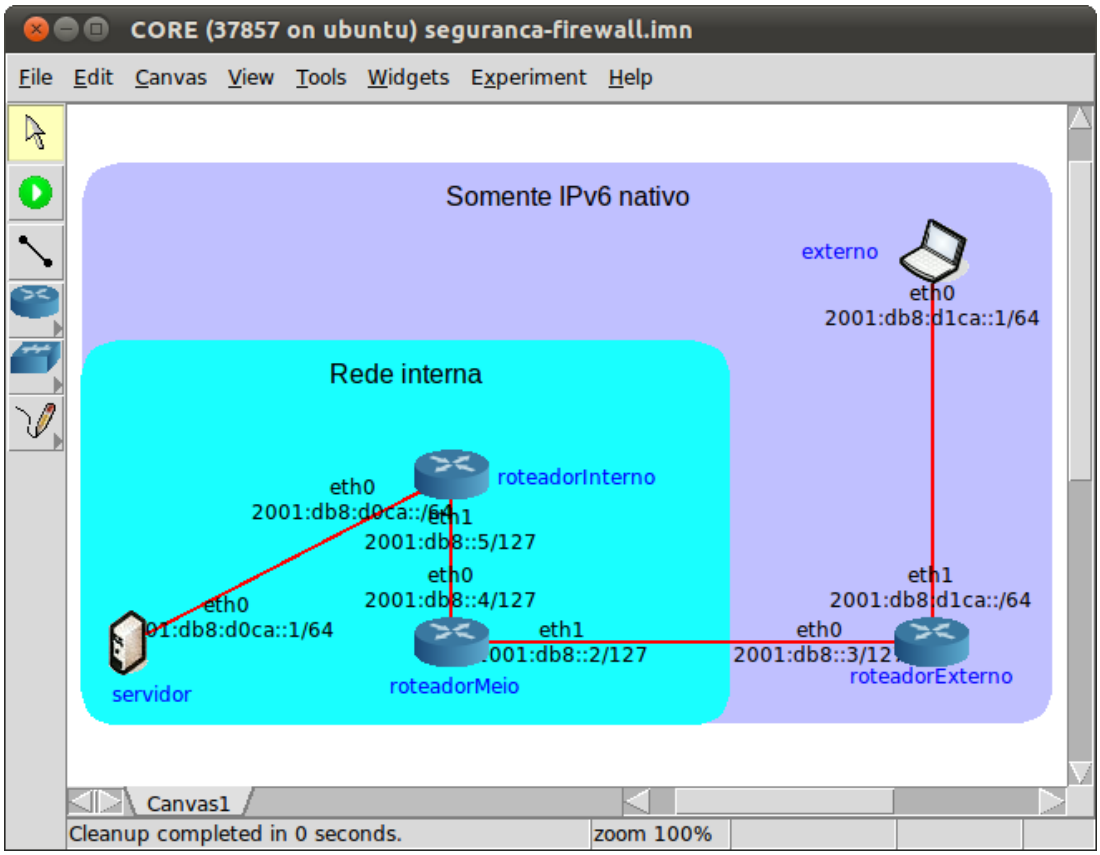
Isso implica que firewalls operando na camada de rede podem, com o IPv6, bloquear funções extremamente básicas como a descoberta de vizinhança e a autoconfiguração de

endereços. Portanto, este laboratório tem a função de ensinar boas práticas para a configuração de firewalls IPv6, evitando que configurações incorretas atrapalhem o funcionamento do IPv6. Para tanto, são seguidas as recomendações da *RFC 4890 -- Recommendations for Filtering ICMPv6 Messages in Firewalls*.

Roteiro Experimental

Experiência 2 - Firewall Stateful

1. Inicie o CORE e abra o arquivo "**seguranca-firewall.imn**" localizado no diretório do desktop "Segurança/Firewall", da máquina virtual fornecida pelo NIC.br. A seguinte topologia deve aparecer:



O objetivo da topologia é representar a estrutura mínima necessária para simular um sistema de firewall. Essa experiência utiliza uma rede interna composta por 'roteador' e 'servidor' e está dividida em duas partes: a configuração de firewall em um servidor ou desktop seguido da configuração de um firewall "no meio do caminho".

A rede foi configurada com rotas estáticas que permitem conexões entre todas as máquinas. Apesar de o 'roteador' transportar apenas pacotes IPv6, a máquina

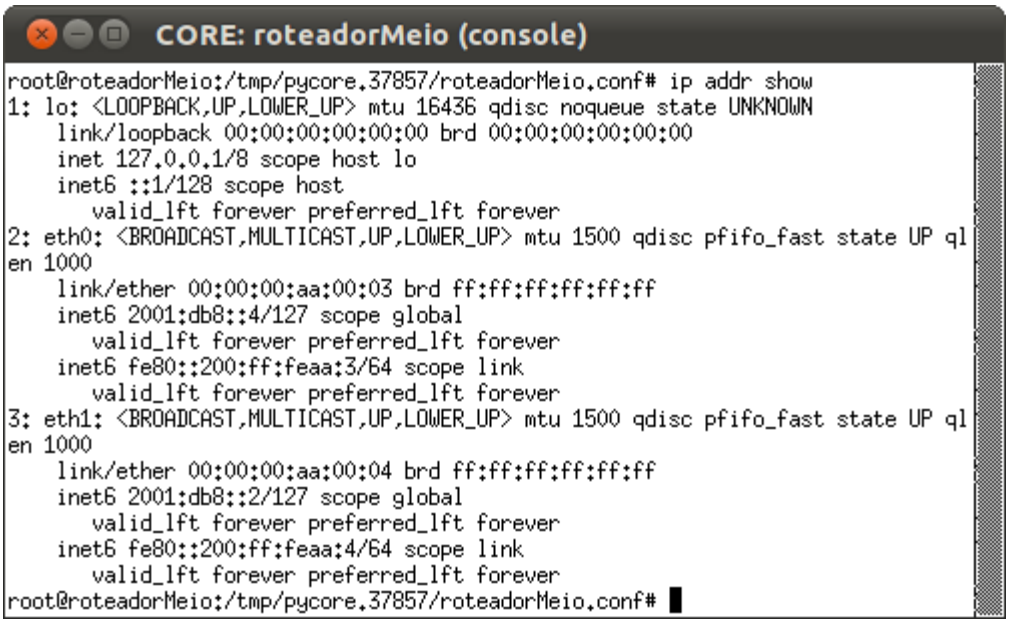
poderia utilizar pilha dupla sem nenhum tipo de modificação na configuração do experimento.

2. Verifique a configuração dos nós da topologia:
   a. Inicie a simulação:
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Start.

   b. Abra o terminal do 'roteadorMeio', através de um duplo-clique, e utilize o seguinte comando para verificar suas configurações das interfaces de rede:

```
# ip addr show
```

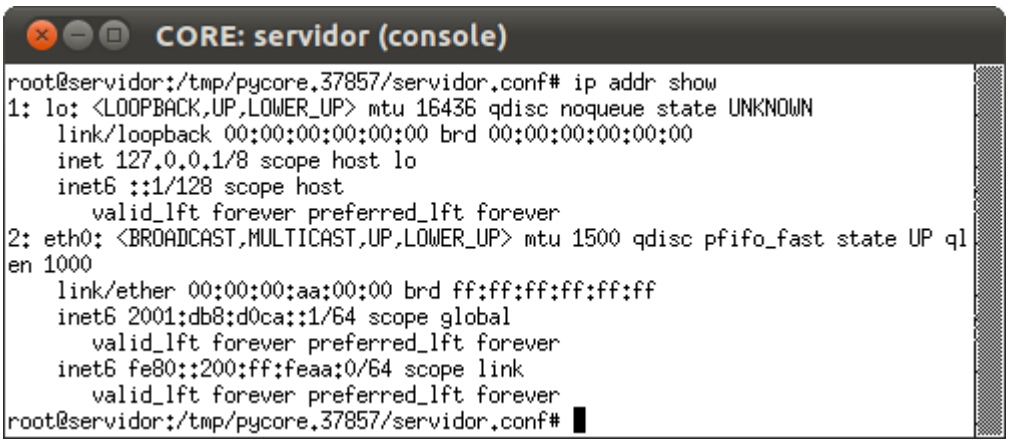
   O resultado deve ser:



   c. Abra o terminal do 'servidor' através de um duplo-clique e utilize o seguinte comando para verificar as configurações de suas interfaces de rede:

```
# ip addr show
```

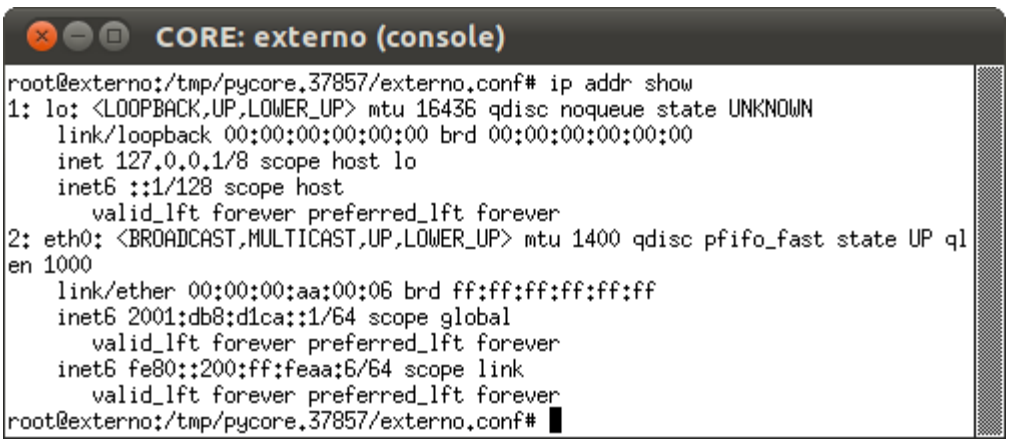
O resultado deve ser:



d. Abra o terminal da máquina ‘externo’ com um duplo-clique e utilize o seguinte comando para verificar as configurações das interfaces de rede:

```
# ip addr show
```

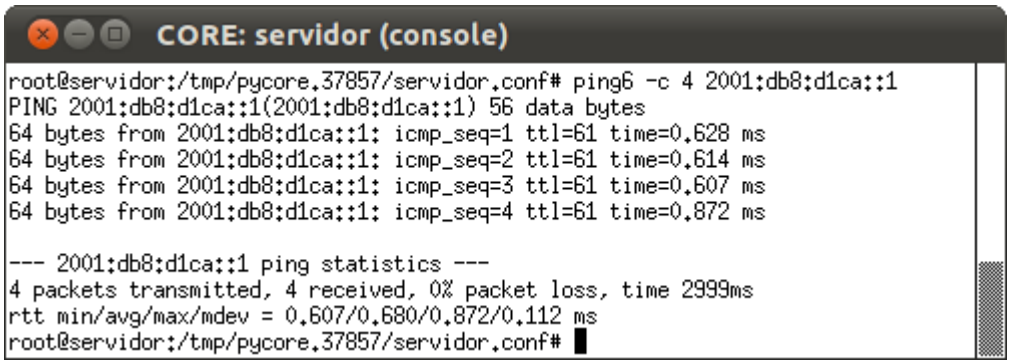
O resultado deve ser:



3. Abra o terminal do ‘servidor’ com um duplo-clique e utilize o seguinte comando para verificar a conectividade:

```
# ping6 -c 4 2001:db8:d1ca::1
```

O resultado deve ser:



4. Configure o firewall no 'servidor'.

   a. Abra o terminal do 'servidor' com um duplo-clique e verifique o conteúdo do arquivo firewall.sh com o seguinte comando:

```
# less firewall.sh
```

   O conteúdo do arquivo é o seguinte:

```
#!/bin/sh

PATH=/sbin:/bin:/usr/sbin:/usr/bin

# caminho do iptables
iptables="/sbin/ip6tables"

# Meus IPs
# IPs destino (todos os IPs locais)
ips_destino="2001:db8:d0ca::1"

start () {
  echo "Iniciando o filtro de pacotes: ip6tables..."

  # A politica padrao eh recusar todos os pacotes
  echo "Configurando a politica padrao para recusar todos os pacotes"
  $iptables -F
  $iptables -P INPUT DROP
  $iptables -P OUTPUT ACCEPT
  $iptables -P FORWARD DROP

  # Permitir trafego ilimitado para o localhost
  echo "Permitindo trafego ilimitado para o localhost"
  $iptables -A INPUT  -i lo -j ACCEPT
  #$iptables -A OUTPUT -o lo -j ACCEPT
```

```
# Desabilita RH0
echo "Desabilita RH0"
$iptables -A INPUT -m rt --rt-type 0 -j DROP

# IPs de documentacao
#echo "Descarta documentacao"
#$iptables -A INPUT -s 2001:0db8::/32 -j DROP

# Stateful firewall
echo "Permitindo pacotes de saida e retorno para todas as conexoes ja estabelecidas"
$iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
#$iptables -A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT

# Da Internet para o IP Unicast Global do host

for ip in $ips_destino
   do
    # Trafego SSH
    echo -n "ssh "
    $iptables -A INPUT -p tcp -s 2000::/3 --sport 513:65535 -d $ip --dport 22 \
      -j ACCEPT

    # Permitindo Traceroute
    echo -n "traceroute "
    $iptables -A INPUT -p udp -m udp -s 2000::/3 -d $ip --dport 33434:33523 \
      -m state --state NEW -j REJECT --reject-with icmp6-port-unreachable

    echo -n "icmp in "
    # ECHO REQUESTS E RESPONSES (Type 128 e 129)
    $iptables -A INPUT -p icmpv6 --icmpv6-type echo-request -d $ip -m limit \
      --limit 10/s -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type echo-reply -d $ip -m limit \
      --limit 10/s -j ACCEPT

    # DESTINATION UNREACHABLE (Type 1)
    $iptables -A INPUT -p icmpv6 --icmpv6-type destination-unreachable -d $ip \
      -j ACCEPT

    # PACKET TOO BIG (Type 2)
    $iptables -A INPUT -p icmpv6 --icmpv6-type packet-too-big -d $ip -j ACCEPT

    # TIME EXCEEDED (Type 3)
    $iptables -A INPUT -p icmpv6 --icmpv6-type ttl-zero-during-transit -d $ip \
      -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type ttl-zero-during-reassembly -d $ip \
      -j ACCEPT

    # PARAMETER PROBLEM (Type 4)
    $iptables -A INPUT -p icmpv6 --icmpv6-type unknown-option -d $ip -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type unknown-header-type -d $ip -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type bad-header -d $ip -j ACCEPT

    # NA (Type 136)
    ## Modifique a linha abaixo -- 1 de 3 ##
```

```
    $iptables -A INPUT -p icmpv6 --icmpv6-type 136 -d $ip -j DROP

done


# para Link local

    # ECHO REQUESTS E RESPONSES (Type 128 e 129)
    $iptables -A INPUT -p icmpv6 --icmpv6-type echo-request -d fe80::/64 -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type echo-reply -d fe80::/64 -j ACCEPT

    # DESTINATION UNREACHABLE (Type 1)
    $iptables -A INPUT -p icmpv6 --icmpv6-type destination-unreachable \
      -d fe80::/64 -j ACCEPT

    # PACKET TOO BIG (Type 2)
    $iptables -A INPUT -p icmpv6 --icmpv6-type packet-too-big -d fe80::/64 -j ACCEPT

    # TIME EXCEEDED (Type 3)
    $iptables -A INPUT -p icmpv6 --icmpv6-type ttl-zero-during-transit \
      -d fe80::/64 -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type ttl-zero-during-reassembly \
      -d fe80::/64 -j ACCEPT

    # PARAMETER PROBLEM (Type 4)
    $iptables -A INPUT -p icmpv6 --icmpv6-type unknown-option -d fe80::/64 -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type unknown-header-type -d fe80::/64 \
      -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type bad-header -d fe80::/64 -j ACCEPT

    # NEIGHBOR DISCOVERY
    # RA (Type 134)
    $iptables -A INPUT -p icmpv6 --icmpv6-type 134 -d fe80::/64 -j ACCEPT
    # NA (Type 136)
    ## Modifique a linha abaixo -- 2 de 3 ##
    $iptables -A INPUT -p icmpv6 --icmpv6-type 136 -d fe80::/64 -j DROP


# Da Internet ou Link Local para Solicited Node

    # NEIGHBOR DISCOVERY
    # NS (Type 135)
    ## Modifique a linha abaixo -- 3 de 3 ##
    $iptables -A INPUT -p icmpv6 --icmpv6-type 135 -d ff02::1:ff00:0/104 -j DROP

# De Link Local para multicast

    # NEIGHBOR DISCOVERY
    # RA (Type 134)
    $iptables -A INPUT -p icmpv6 --icmpv6-type 134 -s fe80::/64 -d ff02::1 -j ACCEPT

    echo .

# Descartando tudo mais
```

```
  echo "Descartando todos os demais pacotes... "
  $iptables -A INPUT -j DROP

}

stop () {
  echo "Parando o filtro de pacotes: ip6tables..."
  $iptables -P INPUT ACCEPT
  $iptables -F INPUT
  $iptables -P OUTPUT ACCEPT
  $iptables -F OUTPUT
  $iptables -P FORWARD ACCEPT
  $iptables -F FORWARD
  echo "Todas as regras e cadeias estao limpas."
  echo "Tome cuidado... Isso eh perigoso!!"
  echo "Execute: ** $0 start ** assim que possivel."
}

status () {
       $iptables --list -v
}

case "$1" in
    start)
  start
    ;;
    stop)
  stop
    ;;
    try|test)
  start
  sleep 10
  stop
    ;;
    restart|reload|force-reload)
  stop
  sleep 2
  start
    ;;
    status)
  status
    ;;
    *)
  echo "Uso: $0 {start|stop|restart|status|try}" >&2
  exit 1
    ;;
esac

exit 0
```

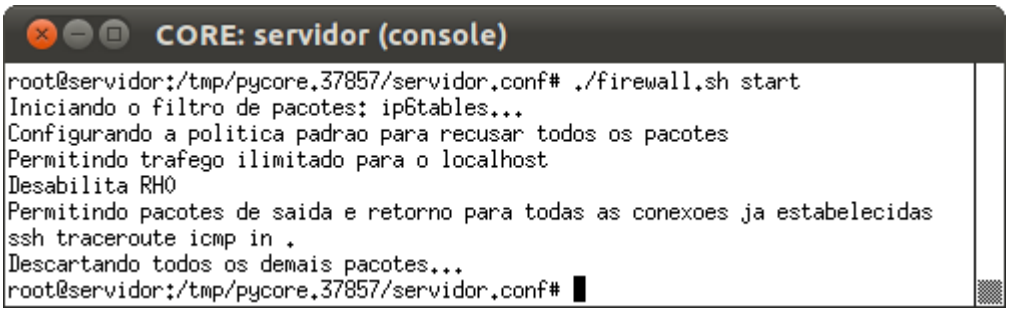
Esse script foi escrito com base na RFC 4890. Note que os três trechos destacados em negrito são regras que rejeitam (DROP) o recebimento de

mensagens ICMPv6 relacionadas a Neighbor Solicitation e Neighbor Advertisement. Nos passos seguintes, será verificado o funcionamento do firewall quando essas messagens são bloqueadas.

b. No mesmo terminal, execute o arquivo de configuração de firewall utilizado do comando:

```
# ./firewall.sh start
```

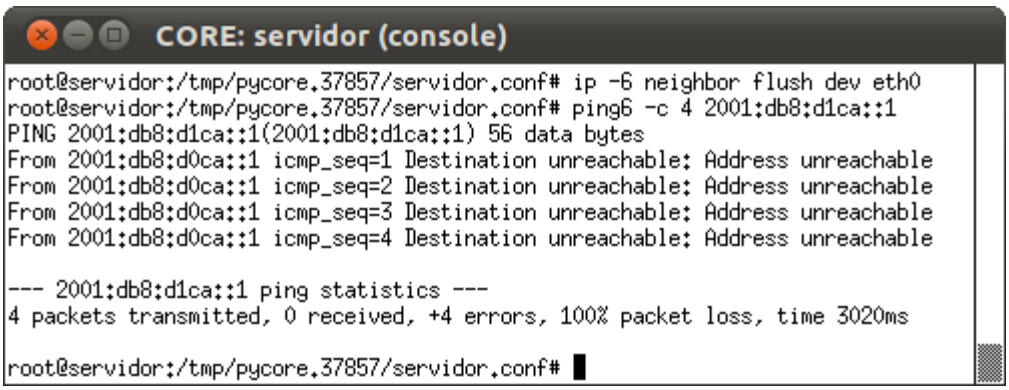
O resultado deve ser:



c. Verifique a conectividade entre a máquina ‘servidor’ e a ‘externo’. Ainda no terminal do ‘servidor’, execute os seguintes comandos:

```
# ip -6 neighbor flush dev eth0
# ping6 -c 4 2001:db8:d1ca::1
```

O resultado deve ser:

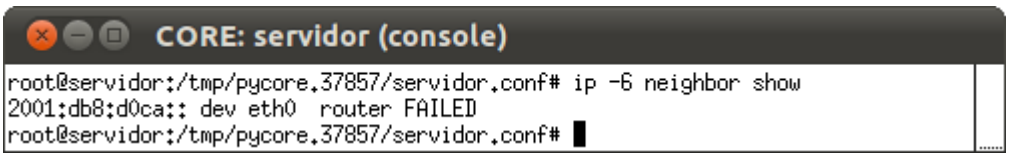


Observe que não há mais conectividade entre o ‘servidor’ e o ‘externo’. A seguir, será verificada a causa disso.

d. Ainda no ‘servidor’, execute o comando:

```
# ip -6 neighbor show
```

O resultado deve ser:

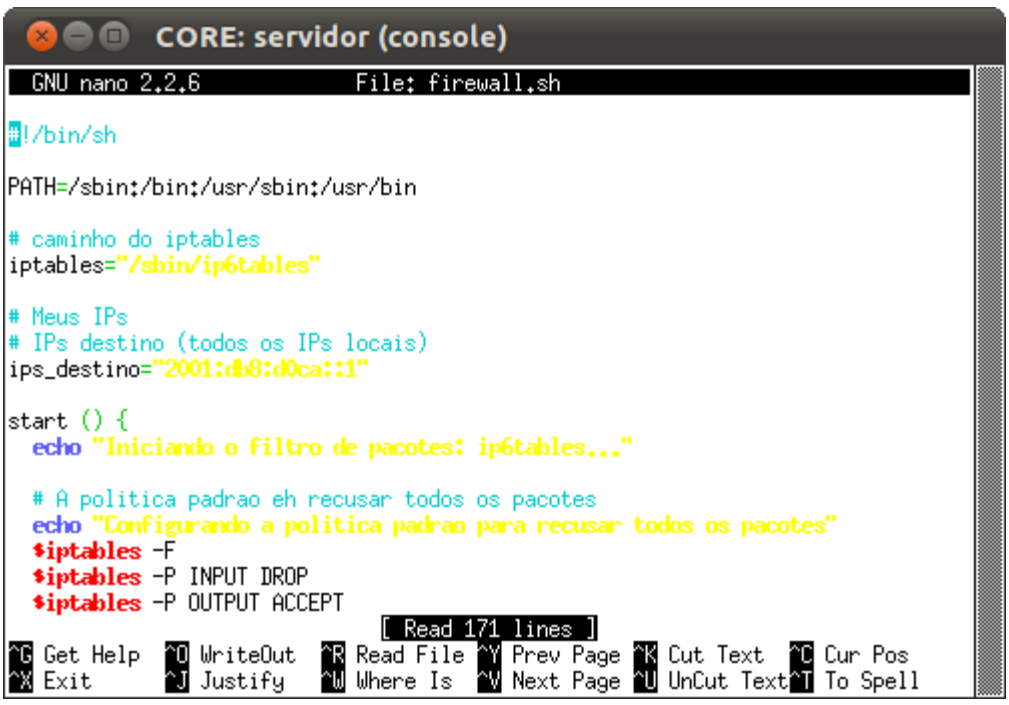


Apesar de o 'servidor' estar diretamente conectado ao 'roteador', não foi possível estabelecer uma comunicação via IPv6, pois o 'servidor' bloqueou as mensagens ICMPv6 de *Neighbor Advertisement* que informam os endereços MAC das máquinas do enlace. Note que, ao contrário da prática usual de se bloquear mensagens ICMP no IPv4, seu bloqueio em IPv6 impossibilita o funcionamento do protocolo, dado seu papel no mapemento de endereços das camadas de rede e de enlace.

e. No mesmo terminal, execute o seguinte comando para editar o script do firewall:

```
# nano firewall.sh
```

O terminal deve ficar assim:



Edite o arquivo modificando as três regras de modo que os **trechos em negrito mostrados no passo 4.a** sejam alterados de DROP para ACCEPT.

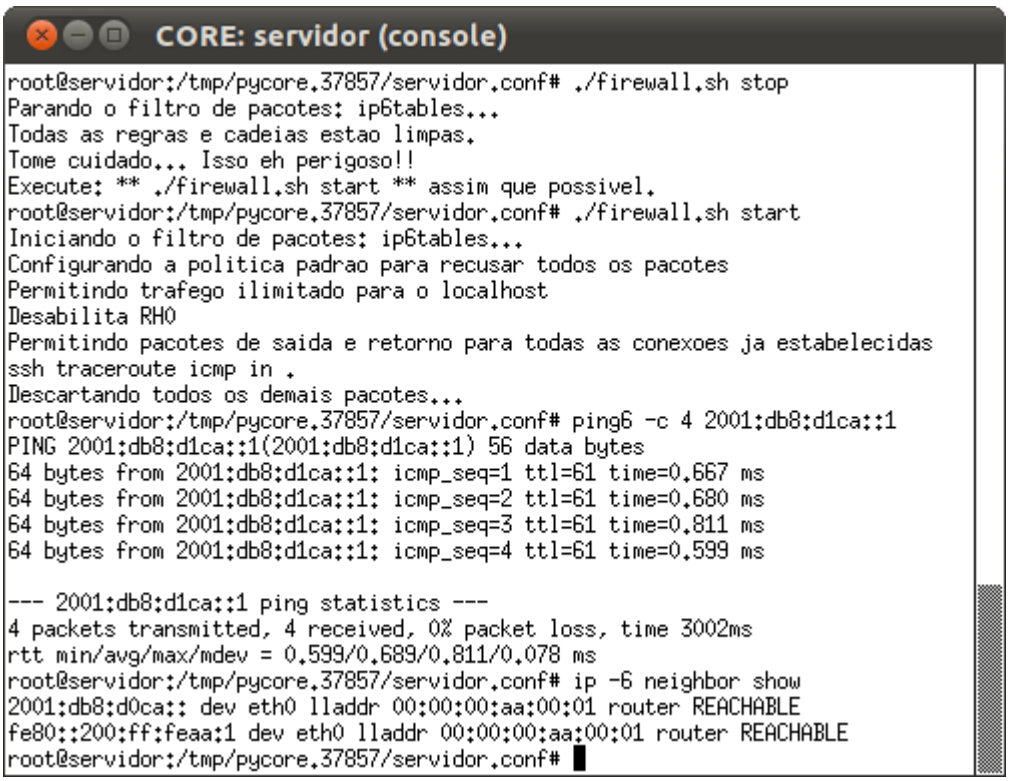
```
…
        # NA (Type 136)
        ## Modifique a linha abaixo -- 1 de 3 ##
        $iptables -A INPUT -p icmpv6 --icmpv6-type 136 -d $ip -j DROP

…

        # NA (Type 136)
        ## Modifique a linha abaixo -- 2 de 3 ##
        $iptables -A INPUT -p icmpv6 --icmpv6-type 136 -d fe80::/64 -j DROP

…

        # NS (Type 135)
        ## Modifique a linha abaixo -- 3 de 3 ##
        $iptables -A INPUT -p icmpv6 --icmpv6-type 135 -d ff02::1:ff00:0/104 -j DROP
…
```

Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

f. A seguir, execute os seguintes comandos:

```
# ./firewall.sh stop
# ./firewall.sh start
# ping6 -c 4 2001:db8:d1ca::1
# ip -6 neighbor show
```

O resultado deve ser:



Note que o ping6 funcionou corretamente. Como exercício adicional, estude as regras habilitadas no firewall e verifique que 'servidor' está apto a receber somente algumas mensagens relacionadas ao ICMPv6, traceroute6 e ssh. Referente à sintaxe do script, busque informações relacionadas ao ip6tables e quanto às regras, estude a RFC 4890.

g. No terminal do 'externo', execute o seguinte comando para verificar o acesso ao serviço ssh do servidor:

```
# ssh core@2001:db8:d0ca::1
```

Como é o primeiro acesso da máquina 'externo' à 'servidor' no serviço *ssh*, será solicitado a inclusão da chave pública RSA do 'servidor'. Aceite-a colocando a resposta "*yes*" no terminal. Quando solicitada, a senha de acesso é *core*.

O resultado deve ser:

h. Após verificar a conectividade via ssh, encerre a sessão utilizando o comando:

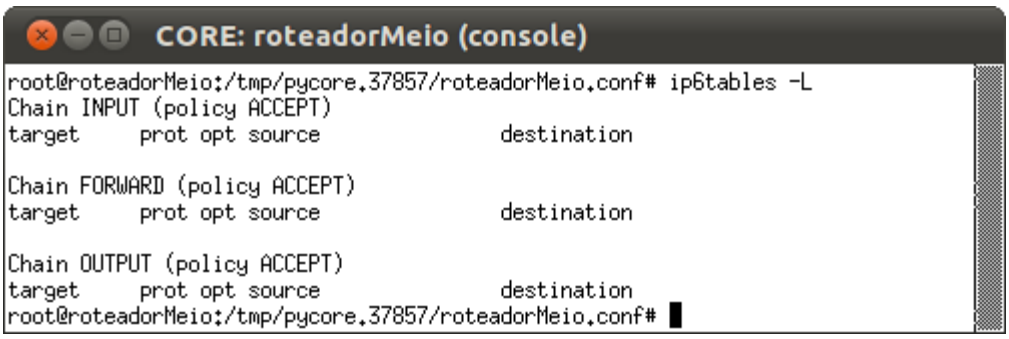
```
# exit
```

5. Configure o firewall no 'roteadorMeio'.

a. Abra o terminal do 'roteadorMeio' através de um duplo-clique e verifique as regras iniciais do firewall:

```
# ip6tables -L
```

O resultado deve ser:



Note que não há nenhum tipo de restrição, uma vez que as políticas de recebimento (INPUT), envio (OUTPUT) e encaminhamento (FORWARD) estão configuradas para aceite (ACCEPT).

b. Ainda nesse terminal , verifique o conteúdo do arquivo firewall.sh com o seguinte comando:

```
# cat firewall.sh
```

O conteúdo do arquivo é o seguinte:

```
#!/bin/sh

PATH=/sbin:/bin:/usr/sbin:/usr/bin

# caminho do iptables
iptables="/sbin/ip6tables"

# Meus IPs
# IPs destino (todos os IPs locais)
ips_destino="2001:db8::2/128 2001:db8::4/128"

# IPs de rede interna
ips_interno="2001:db8:d0ca::/48 2001:db8::4/127"

start () {
  echo "Iniciando o filtro de pacotes: ip6tables..."

  # A politica padrao eh recusar todos os pacotes
  echo "Configurando a politica padrao para recusar todos os pacotes"
  $iptables -F
  $iptables -P INPUT DROP
  $iptables -P OUTPUT ACCEPT
  $iptables -P FORWARD DROP

  # Permitir trafego ilimitado para o localhost
  echo "Permitindo trafego ilimitado para o localhost"
  $iptables -A INPUT  -i lo -j ACCEPT
  #$iptables -A OUTPUT -o lo -j ACCEPT

  # Desabilita RH0
  echo "Desabilita RH0"
  $iptables -A INPUT -m rt --rt-type 0 -j DROP
  $iptables -A FORWARD -m rt --rt-type 0 -j DROP

  # IPs de documentacao
  #echo "Descarta documentacao"
  #$iptables -A INPUT -s 2001:0db8::/32 -j DROP
  #$iptables -A FORWARD -s 2001:0db8::/32 -j DROP

  # Stateful firewall
  echo "Permitindo pacotes de saida e retorno para todas as conexoes ja estabelecidas"
  $iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
  #$iptables -A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT

  # Da Internet para o IP Unicast Global do host

  for ip in $ips_destino
```

```
   do
    # Permitindo Traceroute
    echo -n "traceroute "
    $iptables -A INPUT -p udp -m udp -s 2000::/3 -d $ip --dport 33434:33523 \
      -m state --state NEW -j REJECT --reject-with icmp6-port-unreachable

    echo -n "icmp in "
    # ECHO REQUESTS E RESPONSES (Type 128 e 129)
    $iptables -A INPUT -p icmpv6 --icmpv6-type echo-request -d $ip -m limit \
      --limit 10/s -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type echo-reply -d $ip -m limit \
      --limit 10/s -j ACCEPT

    # DESTINATION UNREACHABLE (Type 1)
    $iptables -A INPUT -p icmpv6 --icmpv6-type destination-unreachable -d $ip \
      -j ACCEPT

    # PACKET TOO BIG (Type 2)
    $iptables -A INPUT -p icmpv6 --icmpv6-type packet-too-big -d $ip -j ACCEPT

    # TIME EXCEEDED (Type 3)
    $iptables -A INPUT -p icmpv6 --icmpv6-type ttl-zero-during-transit -d $ip \
      -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type ttl-zero-during-reassembly -d $ip \
      -j ACCEPT

    # PARAMETER PROBLEM (Type 4)
    $iptables -A INPUT -p icmpv6 --icmpv6-type unknown-option -d $ip -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type unknown-header-type -d $ip -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type bad-header -d $ip -j ACCEPT

    # NA (Type 136)
    $iptables -A INPUT -p icmpv6 --icmpv6-type 136 -d $ip -j ACCEPT

done

# Da Internet para a rede interna do roteador

for ip in $ips_interno
   do

    # Stateful firewall
    echo -n "Permitindo pacotes de saida e retorno para todas as conexoes ja"
    echo " estabelecidas"
    $iptables -A FORWARD -m state -s 2000::/3 -d $ip --state RELATED,ESTABLISHED \
      -j ACCEPT
    $iptables -A FORWARD -m state -s $ip -d 2000::/3 \
      --state NEW,RELATED,ESTABLISHED -j ACCEPT

    # Permitindo Traceroute
    echo -n "traceroute "
    $iptables -A FORWARD -p udp -m udp -s 2000::/3 -d $ip --dport 33434:33523 \
      -m state --state NEW -j ACCEPT
```

```
    echo -n "icmp in "
    # ECHO REQUESTS E RESPONSES (Type 128 e 129)
    $iptables -A FORWARD -p icmpv6 --icmpv6-type echo-request -m limit \
      --limit 10/s -j ACCEPT
    $iptables -A FORWARD -p icmpv6 --icmpv6-type echo-reply -d $ip -m limit \
      --limit 10/s -j ACCEPT

    # DESTINATION UNREACHABLE (Type 1)
    $iptables -A FORWARD -p icmpv6 --icmpv6-type destination-unreachable -d $ip \
      -j ACCEPT

    # PACKET TOO BIG (Type 2)
    $iptables -A FORWARD -p icmpv6 --icmpv6-type packet-too-big -d $ip -j ACCEPT

    # TIME EXCEEDED (Type 3)
    $iptables -A FORWARD -p icmpv6 --icmpv6-type ttl-zero-during-transit -d $ip \
      -j ACCEPT
    $iptables -A FORWARD -p icmpv6 --icmpv6-type ttl-zero-during-reassembly -d $ip \
      -j ACCEPT

    # PARAMETER PROBLEM (Type 4)
    $iptables -A FORWARD -p icmpv6 --icmpv6-type unknown-option -d $ip -j ACCEPT
    $iptables -A FORWARD -p icmpv6 --icmpv6-type unknown-header-type -d $ip \
      -j ACCEPT
    $iptables -A FORWARD -p icmpv6 --icmpv6-type bad-header -d $ip -j ACCEPT

    # NA (Type 136)
    $iptables -A FORWARD -p icmpv6 --icmpv6-type 136 -d $ip -j ACCEPT

done


# para Link local

    # ECHO REQUESTS E RESPONSES (Type 128 e 129)
    $iptables -A INPUT -p icmpv6 --icmpv6-type echo-request -d fe80::/64 -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type echo-reply -d fe80::/64 -j ACCEPT

    # DESTINATION UNREACHABLE (Type 1)
    $iptables -A INPUT -p icmpv6 --icmpv6-type destination-unreachable \
      -d fe80::/64 -j ACCEPT

    # PACKET TOO BIG (Type 2)
    $iptables -A INPUT -p icmpv6 --icmpv6-type packet-too-big -d fe80::/64 -j ACCEPT

    # TIME EXCEEDED (Type 3)
    $iptables -A INPUT -p icmpv6 --icmpv6-type ttl-zero-during-transit \
      -d fe80::/64 -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type ttl-zero-during-reassembly \
      -d fe80::/64 -j ACCEPT

    # PARAMETER PROBLEM (Type 4)
    $iptables -A INPUT -p icmpv6 --icmpv6-type unknown-option -d fe80::/64 -j ACCEPT
    $iptables -A INPUT -p icmpv6 --icmpv6-type unknown-header-type -d fe80::/64 \
```

```
        -j ACCEPT
      $iptables -A INPUT -p icmpv6 --icmpv6-type bad-header -d fe80::/64 -j ACCEPT

      # NEIGHBOR DISCOVERY
      # RA (Type 134)
      $iptables -A INPUT -p icmpv6 --icmpv6-type 134 -d fe80::/64 -j ACCEPT
      # NS (Type 135)
      $iptables -A INPUT -p icmpv6 --icmpv6-type 135 -d fe80::/64 -j ACCEPT
      # NA (Type 136)
      $iptables -A INPUT -p icmpv6 --icmpv6-type 136 -d fe80::/64 -j ACCEPT


  # Da Internet ou Link Local para Solicited Node

      # NEIGHBOR DISCOVERY
      # NS (Type 135)
      $iptables -A INPUT -p icmpv6 --icmpv6-type 135 -d ff02::1:ff00:0/104 -j ACCEPT

  # De Link Local para multicast

      # NEIGHBOR DISCOVERY
      # RA (Type 134)
      $iptables -A INPUT -p icmpv6 --icmpv6-type 134 -s fe80::/64 -d ff02::1 -j ACCEPT

      echo .

  # Descartando tudo mais
  echo "Descartando todos os demais pacotes... "
  $iptables -A INPUT -j DROP
  $iptables -A FORWARD -j DROP

}

stop () {
  echo "Parando o filtro de pacotes: ip6tables..."
  $iptables -P INPUT ACCEPT
  $iptables -F INPUT
  $iptables -P OUTPUT ACCEPT
  $iptables -F OUTPUT
  $iptables -P FORWARD ACCEPT
  $iptables -F FORWARD
  echo "Todas as regras e cadeias estao limpas."
  echo "Tome cuidado... Isso eh perigoso!!"
  echo "Execute: ** $0 start ** assim que possivel."
}

status () {
       $iptables --list -v
}

case "$1" in
    start)
  start
    ;;
```

```
    stop)
  stop
    ;;
    try|test)
  start
  sleep 10
  stop
    ;;
    restart|reload|force-reload)
  stop
  sleep 2
  start
    ;;
    status)
  status
    ;;
    *)
  echo "Uso: $0 {start|stop|restart|status|try}" >&2
  exit 1
    ;;
esac

exit 0
```

Esse script também foi escrito com base na RFC 4890. Quando comparado ao mostrado no passo 4.a, destacam-se a adição de regras referentes ao ICMPv6 e a duplicação das regras referentes aos endereços IP globais atreladas ao encaminhamento (FORWARD) dos pacotes permitidos na rede interna, além dos dois endereços IPv6 na variável ips_destino (2001:db8:d0ca:: e 2001:db8:d1ca::).

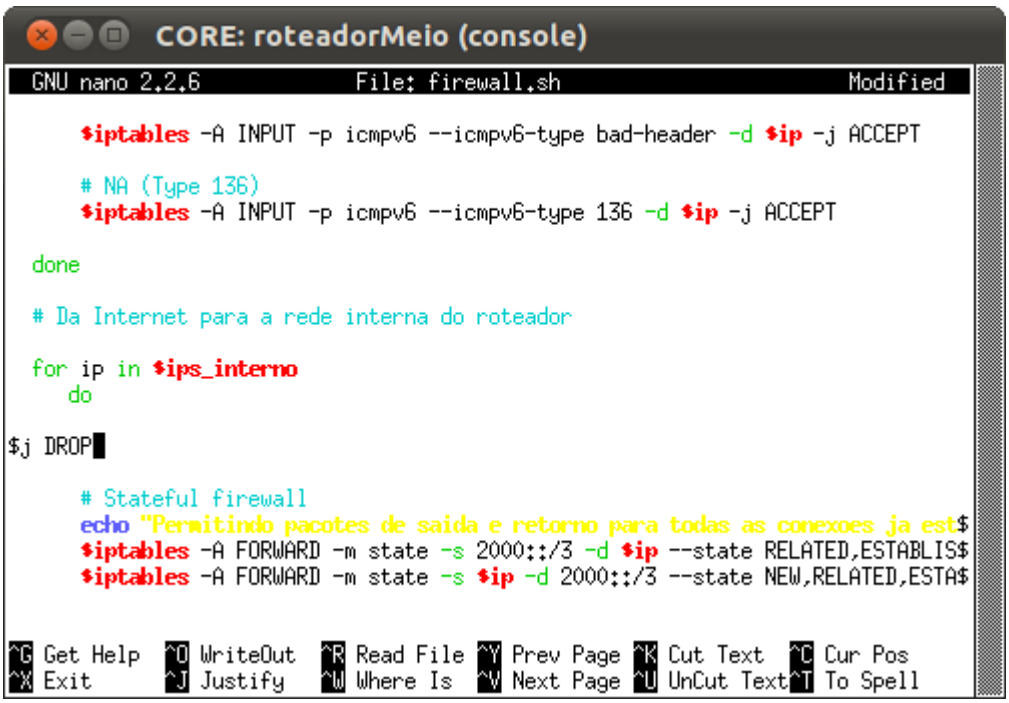
c. No mesmo terminal, execute o seguinte comando para editar o script do firewall:

```
# nano firewall.sh
```

Insira a linha destacada em negrito na seguinte posição do arquivo:

```
# Da Internet para a rede interna do roteador

  for ip in $ips_interno
     do

     $iptables -A FORWARD -p icmpv6 --icmpv6-type packet-too-big -d $ip -j DROP

      # Stateful firewall
```

Após a edição, a tela esperada é:

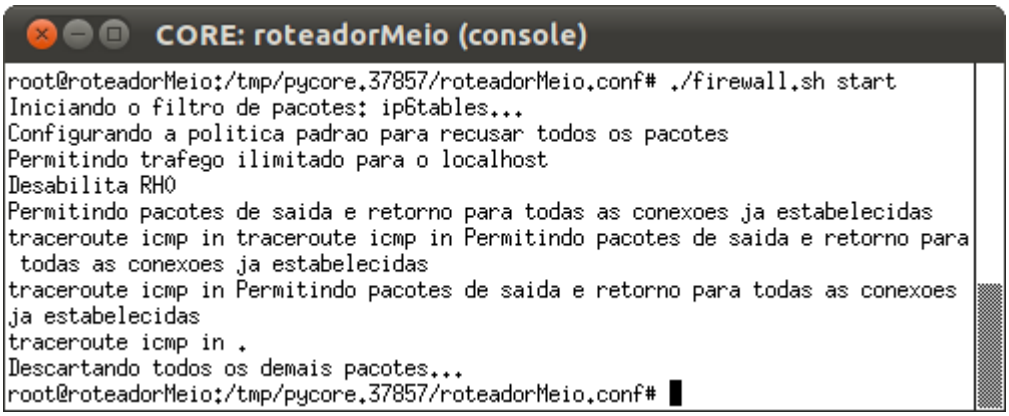


Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

d. Ainda no terminal do 'roteadorMeio', execute o arquivo de configuração do firewall:

```
# ./firewall.sh start
```

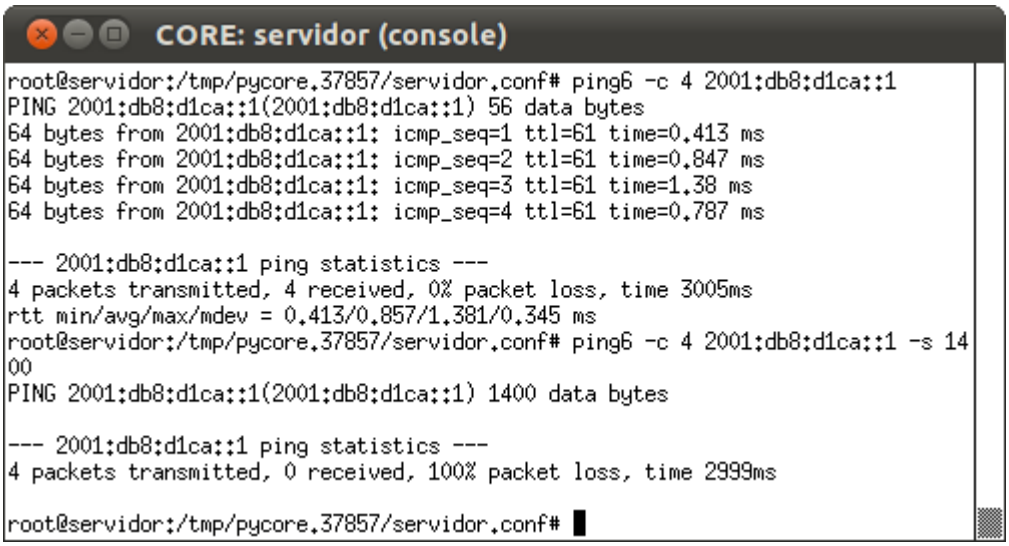
O resultado deve ser:



e. No terminal do ‘servidor’, execute os seguintes comandos:

```
# ping6 -c 4 2001:db8:d1ca::1
# ping6 -c 4 2001:db8:d1ca::1 -s 1400
```

O resultado deve ser:

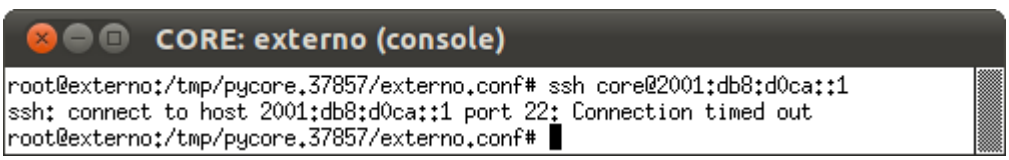


Note que há conectividade entre ‘servidor’ e ‘externo’, como podemos verificar no resultado do primeiro ‘ping’. Já o segundo ‘ping’ foi mal sucedido pois inserimos no passo 5.c a regra de bloqueio do encaminhamento de pacotes ICMPv6 do tipo “*packet too big*” destinados à rede interna.

f. No terminal do ‘externo’, execute o seguinte comando:

```
# ssh core@2001:db8:d0ca::1
```

O resultado deve ser:



Verificando as regras atribuídas ao ip6tables, podemos verificar que o acesso à porta 22 de ‘servidor’ (2001:db8:d0ca::1) não está liberado. Nos próximos passos, editaremos o firewall “no meio do caminho” para resolver a questão.

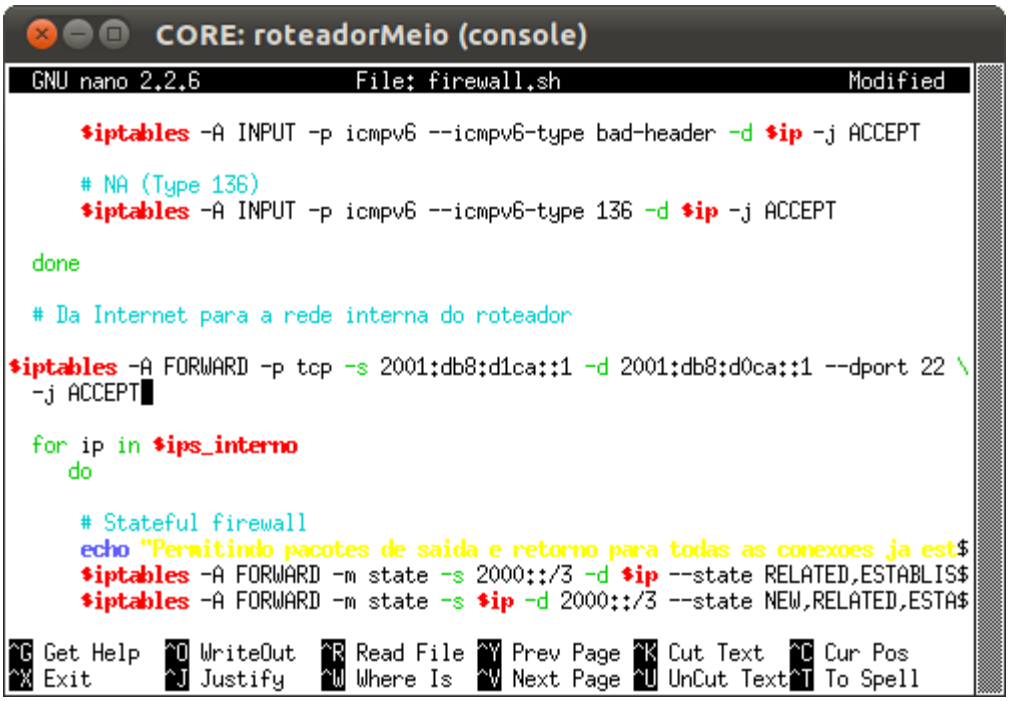
g. No terminal do 'roteadorMeio', execute o seguinte comando:

```
# nano firewall.sh
```

Edite o arquivo conforme o seguinte trecho do arquivo:

```
# Da Internet para a rede interna do roteador
$iptables -A FORWARD -p tcp -s 2001:db8:d1ca::1 -d 2001:db8:d0ca::1 --dport 22 \
  -j ACCEPT

  for ip in $ips_interno
     do

     $iptables -A FORWARD -p icmpv6 --icmpv6-type packet-too-big -d $ip -j DROP

      # Stateful firewall
```

Após a edição, a tela esperada é:

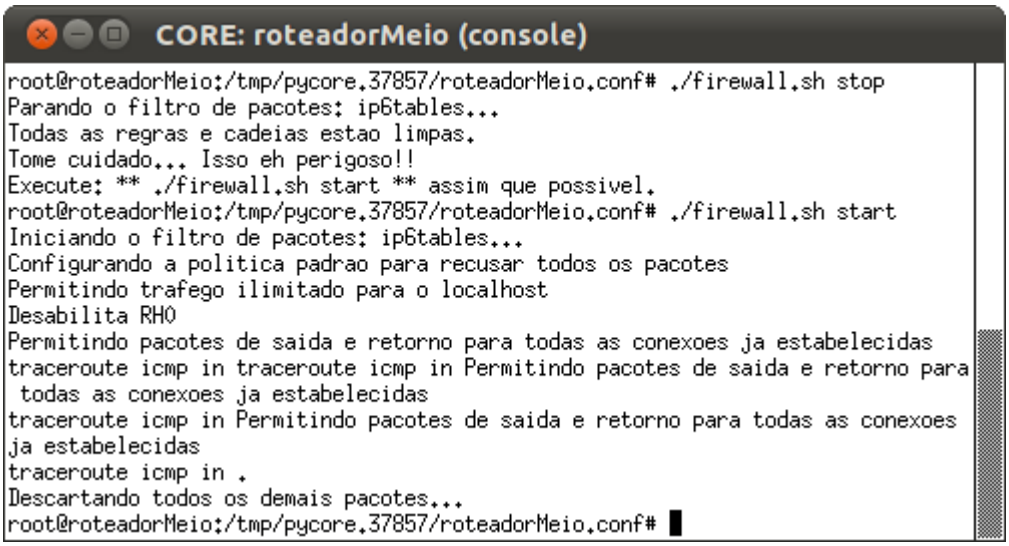


Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

h. Ainda no terminal do 'roteador', execute os seguintes comandos:

```
# ./firewall.sh stop
# ./firewall.sh start
```

O resultado deve ser:



i. No terminal de externo, execute o seguinte comando novamente:

```
# ssh core@2001:db8:d0ca::1
```

Lembrando que a senha de acesso é *core*.

O resultado deve ser parecido com:

j. Após verificar a conectividade via ssh, encerre a sessão através do comando:

```
# exit
```

6. Encerre a simulação:
   a. aperte o botão  ; ou
   b. utilize o menu Experiment > Stop.

IPv6 - Laboratório de IPsec

Objetivo

Esse laboratório tem o objetivo de demonstrar a configuração de uma rede para a utilização de IPsec. Para isto ele é dividido em três partes:

1. IPsec em modo de transporte;
2. IPsec em modo túnel;
3. Configuração automática através do Racoon

Para o presente exercício serão utilizadas as topologias descrita nos arquivos:
seguranca-ipsec-transporte.imn
seguranca-ipsec-tunel.imn
seguranca-ipsec-racoon.imn

Introdução Teórica

Quando o protocolo IPv4 foi concebido, definiu-se que os dados enviados em um determinado pacote IP não receberiam qualquer tipo de ofuscamento ou criptografia na camada de internet. Caso esta proteção fosse necessária, a responsabilidade seria da camada de aplicação. Outro ponto, também não previsto na concepção do protocolo IP, foi a verificação da autenticidade do pacote, o que impossibilitava que a máquina de destino validasse o endereço de origem do pacote (isto é, verificasse se o IP de origem apresentado no pacote era realmente o IP de origem do mesmo). Por causa disso, era possível alterar ou falsificar este endereço IP de origem sem que isto fosse descoberto.

O IPsec foi criado para suprir esta deficiência. Ele é uma suite de protocolos de extensão do protocolo IP e oferece serviços de segurança para prover autenticidade, integridade e confidencialidade aos pacotes IP. Os serviços são providos na camada de rede e portanto também oferecem proteção às camadas superiores. A sua arquitetura foi originalmente especificada na **RFC2401** em 1998 e posteriormente atualizada pela **RFC4301** em 2005.

Há dois modos de operação no IPsec: o Modo de Transporte e o Modo Túnel. O primeiro é usado para proteger a conexão entre apenas duas máquinas, enquanto que o segundo pode ser implementado entre roteadores de borda em redes diferentes, protegendo assim todo o tráfego entre estas duas redes. O IPsec possui dois protocolos: o AH (*Authentication Header* - Cabeçalho de Autenticação), que provê autenticação dos pacotes, e o ESP (*Encapsulated Security Payload* - Dados Encapsulados com Segurança), que provê criptografia.

No Linux, as configurações do IPsec são feitas via o comando `setkey`, que manipula dois bancos de dados: o SAD (Security Association Database) e o SPD (Security Policy

Database). O SAD contém as SA's da máquina (Security Associations), que são configurações de segurança da conexão entre essa máquina e outra. No IPsec, existem dois processos de segurança: autenticação e criptografia. Uma única SA só pode definir a configuração de um dos dois processos, nunca ambos. Por isso, caso a comunicação entre duas máquinas precise ter tanto autenticação quanto criptografia, deverá configurar-se SA's separadas para cada processo. É importante ressaltar também que cada SA configura a comunicação em apenas um sentido da conexão, pois uma SA define a máquina de origem e a de destino dos pacotes. Por isso, para configurar tanto a autenticação quanto a criptografia numa conexão entre duas máquinas, e isso nos dois sentidos, é necessário criar duas SA's para cada processo, num total de quatro (notar que a configuração deve ser feita nas duas máquinas, logo serão quatro SA's numa máquina e quatro na outra). Observe que ainda assim é possível criar apenas uma SA (seja de autenticação ou de criptografia) e desta maneira configurar uma conexão segura apenas num sentido de transmissão. Porém, como o usual é autenticar e criptografar a comunicação nos dois sentidos, as SA's costumam ser geradas aos pares. O segundo banco de dados (SPD), por outro lado, contém as políticas de segurança da máquina, que definem se as configurações serão executadas na comunicação entre as máquinas, juntamente com mais alguns detalhes de como isto é feito.

A manipulação destes bancos de dados é feita através do comando `setkey`, através de seus subcomandos. Estes subcomandos podem ser executados de duas maneiras: 1) inserindo em tempo de execução, um a um, ou 2) carregados a partir de um arquivo de configuração (com a extensão ".conf"). Por default, o `setkey` carrega o arquivo de configuração "/etc/ipsec-tools.conf" na inicialização do sistema. Este arquivo se apresenta totalmente comentado (portanto "vazio") logo após a instalação do ipsec-tools, o que significa que o IPsec não começará a operar automaticamente após sua instalação. Se este arquivo permanecer totalmente comentado, cada vez que o sistema for reinicializado o IPsec perderá qualquer configuração que tenha sido aplicada na última sessão do sistema. Portanto, para que uma determinada configuração do IPsec precise ser permanente e continue válida mesmo após a reinicialização do sistema, as configurações devem ser escritas neste arquivo "ipsec-tools.conf". Caso a configuração seja experimental, ou não deva ser utilizada mais que uma vez, recomenda-se criar outro arquivo ".conf", escrever nele as configurações do IPsec e carregá-las com o `setkey`, ou então executar os subcomandos do `setkey` em tempo de execução.

Embora a configuração manual do IPsec seja viável para um pequeno número de nós, conforme a rede aumenta, torna-se trabalhoso (e propenso a erros) gerar e administrar as chaves de autenticação/criptografia de todas as duplas de nós. Para contornar esta situação, foram criadas formas de configuração automática do IPsec. No Linux, o daemon que cumpre este papel é o Racoon. Para configurá-lo e utilizá-lo, o usuário deverá primeiro criar as políticas de segurança da máquina manualmente. Porém, não será necessário gerar as SA's entre os nós: tanto as SA's quanto suas chaves de autenticação e criptografia correspondentes serão geradas pelo Racoon. Para que o Racoon faça isso, o usuário deverá configurá-lo através do seu arquivo ".conf", que descreve a forma como o Racoon criará e trocará as chaves de segurança entre os nós. O método mais simples e comum do Racoon fazer isso é através das "chaves pré-compartilhadas". Ele cria uma conexão

segura temporária entre os nós para uso exclusivo do Racoon, e então troca as chaves entre os nós de maneira criptografada. Para abrir esta conexão segura, o Racoon verificará em cada nó se ele possui a “chave pré-compartilhada” do seu par para comunicar-se com ele e trocar as chaves do IPsec. Estas chaves pré-compartilhadas (que ao contrário das chaves do IPsec não precisarão ter um formato específico) deverão ser colocadas em cada nó previamente pelo usuário, junto com o arquivo de configuração “.conf” do Raccon. Embora num primeiro momento o processo aparente ser totalmente manual, uma vez que o arquivo de configuração do Racoon e as chaves pré-compartilhadas sejam colocados em todos os nós e o Racoon seja inicializado, o processo de gerenciamento e substituição periódica das chaves do IPsec se tornará totalmente automático, sob o comando do Racoon.

Roteiro Experimental

Experiência 3 - IPsec em Modo de Transporte

1. Se você estiver utilizando a máquina virtual fornecida, vá para o passo 4. Para fazer algumas verificações durante o experimento será necessária a utilização do programa Wireshark, que captura os pacotes enviados através da rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

   Antes da instalação será solicitada a senha do usuário “core”, que também será “core”. Digite a senha para prosseguir com a instalação.

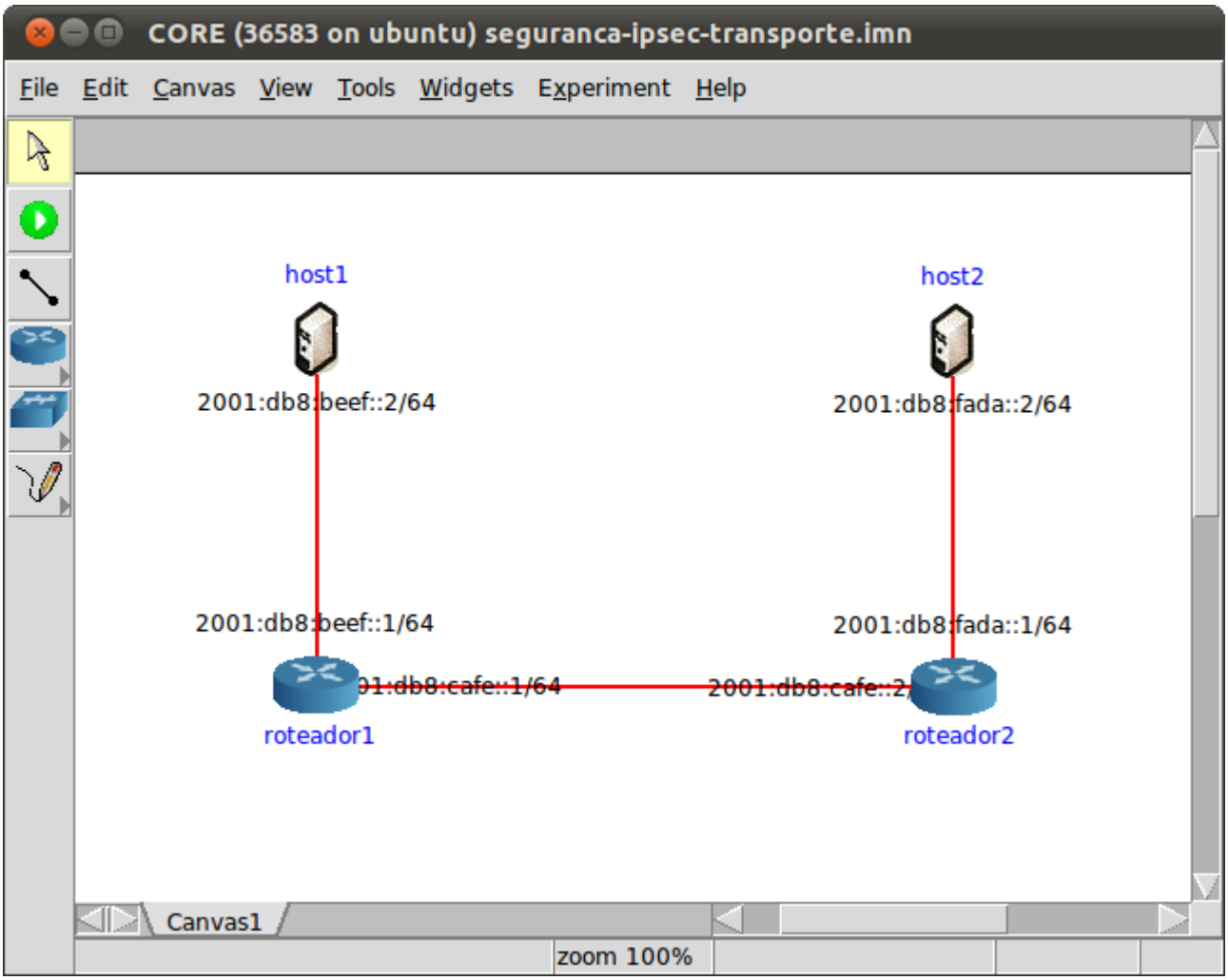
2. Caso as ferramentas de IPsec não estejam instaladas em sua máquina, execute o comando abaixo. A senha da máquina virtual, quando solicitada, é “core”.

```
$ sudo apt-get install ipsec-tools
```

3. Para fazer o envio de pacotes forjados será necessário utilizar as ferramentas THC-IPv6, disponíveis em:
   http://www.thc.org/thc-ipv6/
   Descompacte o pacote e execute os comandos abaixo, na pasta descompactada, para instalar. A senha, quando solicitada, será “core”.

```
$ sudo make
$ sudo make install
```

4. Inicie o CORE e abra o arquivo **"seguranca-ipsec-transporte.imn"** localizado no diretório do desktop "Segurança/IPsec". A seguinte topologia de rede deve aparecer:



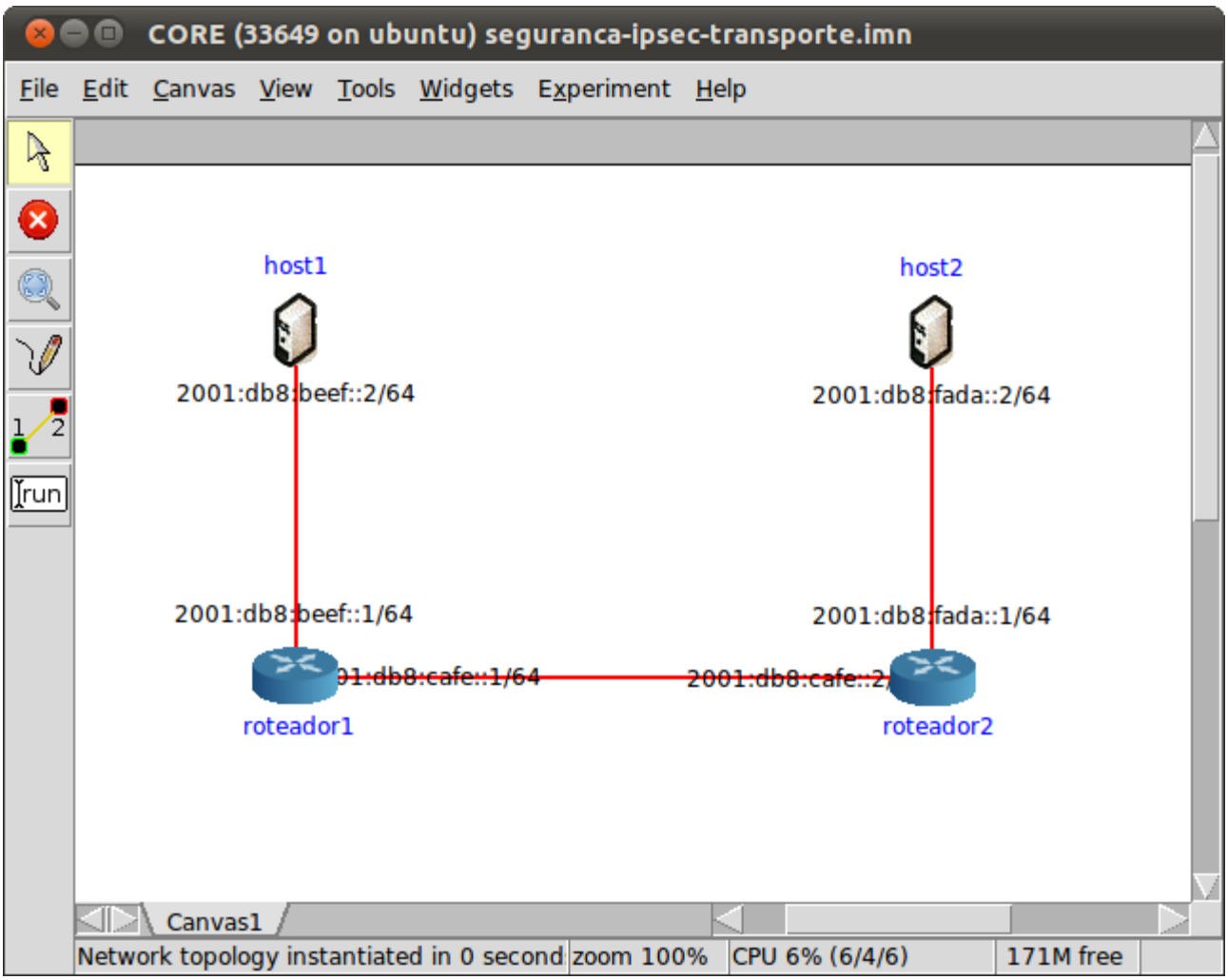
O objetivo dessa topologia de rede é demonstrar que a implementação do IPsec em modo de transporte independe da topologia de rede, isto é, que o IPsec consegue autenticar e criptografar pacotes que são roteados por diversos nós até chegar ao destino. Neste exemplo, implementaremos o IPsec entre o host1 e o host2. Note que há dois roteadores intermediando a comunicação entre os hosts.

Primeiro enviaremos pacotes de ping sem o IPsec estar configurado. A seguir, configuraremos uma conexão com somente autenticação entre os dois hosts. Depois disso, configuraremos uma conexão com autenticação e criptografia, capturando pacotes a cada alteração da configuração. Ao final, analisaremos os pacotes capturados nas três situações.

5. Inicie a simulação:

   a. aperte o botão  ;
   ou utilize o menu Experiment > Start.

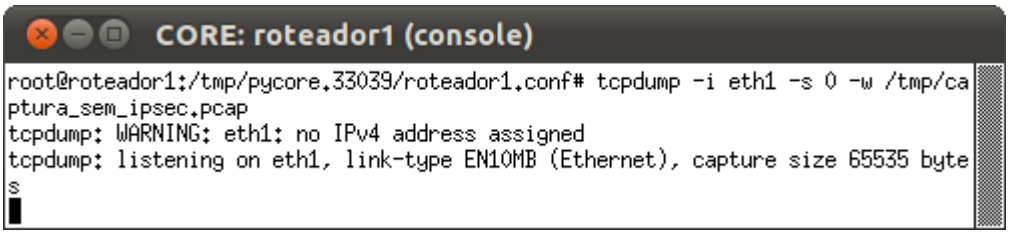
   b. Espere até que o CORE termine a inicialização da simulação. A tela deve ficar como a imagem abaixo. Então, abra um terminal para o roteador1, um para o host1 e um para o host2, através do duplo-clique no símbolo de cada um dos três nós.



6. Faça a captura de pacotes transmitidos entre os hosts sem a configuração do IPsec
   a. Vá para o terminal do roteador1 e utilize o seguinte comando para iniciar a captura de pacotes do roteador:

```
$ tcpdump -i eth1 -s 0 -w /tmp/captura_sem_ipsec.pcap
```

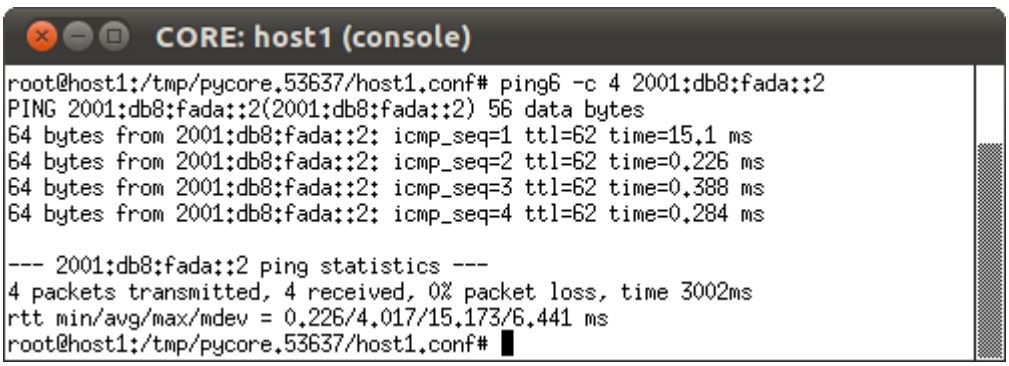
O resultado deve ser:



b. Vá para o terminal do host1 e execute um ping6 para o host2:

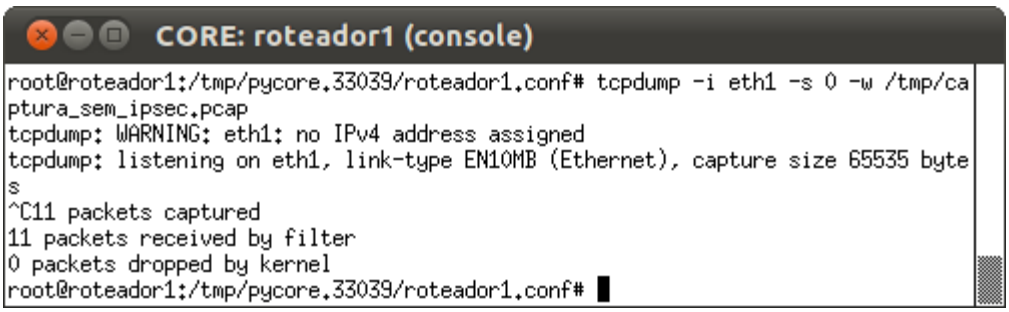
```
$ ping6 -c 4 2001:db8:fada::2
```

O resultado deve ser:



c. No terminal do roteador1, encerre a captura de pacotes através da sequência Ctrl+C.

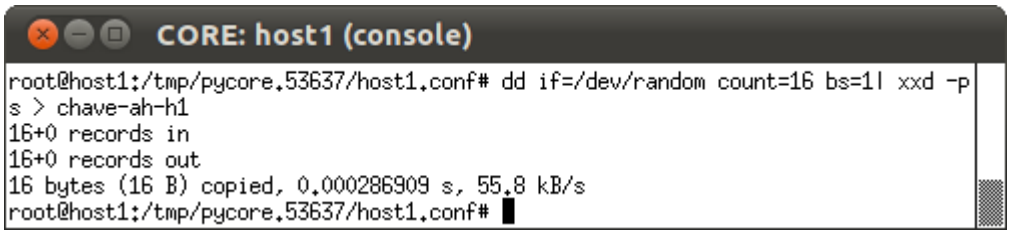
O resultado deve ser similar ao abaixo, já que o número de pacotes capturados pode variar:



7. Vá para o terminal do host1 e gere a chave de autenticação AH executando o comando abaixo. Este comando gerará a chave AH, criará o arquivo `chave-ah-h1` e armazenará nele a chave gerada.

```
$ dd if=/dev/random count=16 bs=1| xxd -ps > chave-ah-h1
```

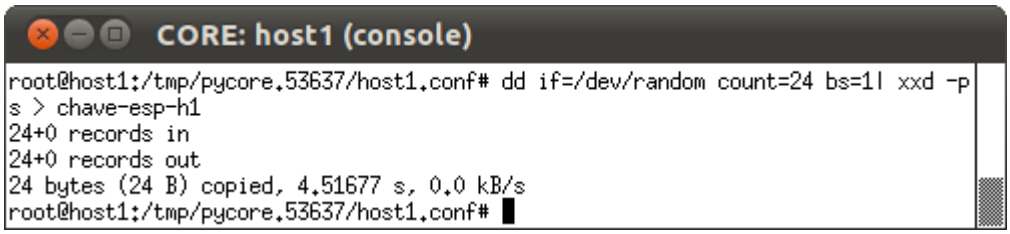
O comando imprimirá a tela abaixo:



8. Gere a chave de criptografia ESP para o mesmo host, executando o comando abaixo no terminal. Este comando gerará a chave ESP, criará o arquivo `chave-esp-h1` e armazenará nele a chave gerada.

```
$ dd if=/dev/random count=24 bs=1| xxd -ps > chave-esp-h1
```

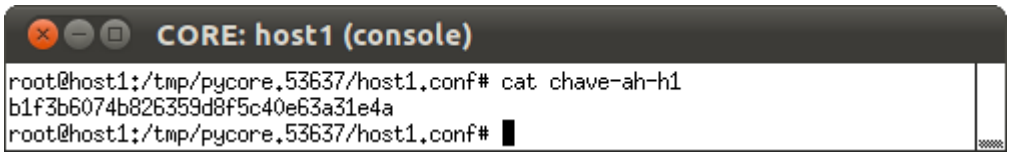
O comando imprimirá a tela abaixo:



Após a execução dos comandos acima, imprima na tela o conteúdo do arquivo `chave-ah-h1` através do comando abaixo e anote o valor apresentado. Execute o comando novamente caso precise anotar a chave de novo. Caso o arquivo seja deletado, execute novamente o comando do passo 7 para gerar uma nova chave (lembrando de trocá-la nos dois arquivos de configuração do IPsec, isto é, tanto no arquivo ".conf" do host1, que iremos começar a escrever agora, quanto no arquivo do host2, que escreveremos depois). **Anote** o valor apresentado:

```
$ cat chave-ah-h1
```

Este comando imprimirá uma tela parecida com a tela abaixo. Note a extensão da chave AH, e os caracteres hexadecimais que a compõem. A chave gerada em sua máquina deve conter a mesma quantidade de caracteres, mas será diferente da chave que aparece na imagem abaixo:



9. Vá para o terminal do host2 e gere a chave de autenticação AH executando o comando abaixo:

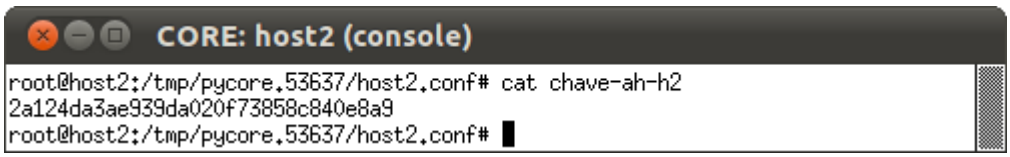
```
$ dd if=/dev/random count=16 bs=1| xxd -ps > chave-ah-h2
```

10. Gere a chave de criptografia ESP para o mesmo host, executando o comando abaixo no terminal:

```
$ dd if=/dev/random count=24 bs=1| xxd -ps > chave-esp-h2
```

Após a execução do comando acima, execute o comando abaixo para imprimir na tela o conteúdo do arquivo `chave-ah-h2`. **Anote** o valor apresentado:

```
$ cat chave-ah-h2
```

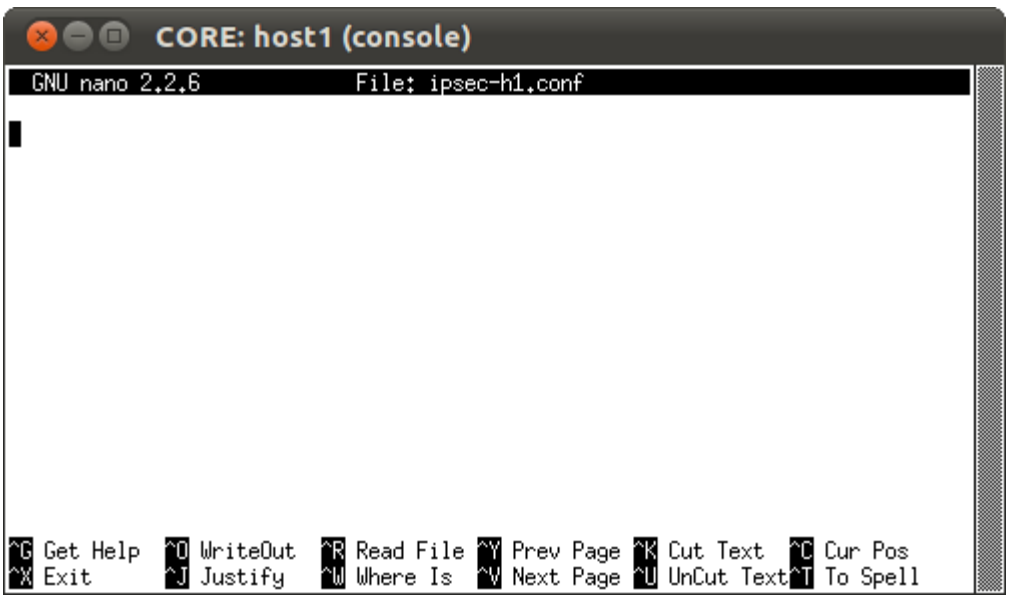


11. Escreva o arquivo de configuração do IPsec para o host1.

a. Vá para o terminal do host1. Crie um arquivo ".conf" com o comando abaixo:

```
$ nano ipsec-h1.conf
```

Com este comando, será aberta a tela do editor de textos `nano`, apresentada a seguir:



Se este arquivo `ipsec-h1.conf` já existisse, o seu conteúdo seria exibido. Porém, como ele não existe, o `nano` abrirá uma tela vazia para o novo arquivo. Após escrever o conteúdo desejado, deveremos salvar as alterações, o que fará com que o `nano` termine de criar o arquivo

`ipsec-h1.conf` e salve nele o conteúdo digitado.

b. Primeiramente, digite as seguintes linhas na tela do `nano`:

```
#!/usr/sbin/setkey -f
flush;
spdflush;
```

A primeira linha define o `setkey` como o interpretador dos comandos que se seguirão. Já o comando `flush` limpa todas as entradas existentes no SAD (Security Association Database), enquanto que o `spdflush` limpa todas as entradas do SPD (Security Policy Database).

c. Insira no arquivo duas linhas de comando de criação de SA, uma linha para cada uma das duas SA's de autenticação do host1. Lembre-se que as SA's são unidirecionais, isto é, cada SA é definida para apenas um sentido da comunicação entre dois pontos de rede, o que torna necessário criar duas SA's para cada comunicação bidirecionalmente segura. As SA's definem também qual o protocolo a ser usado (nos nossos exemplos, ou AH ou ESP), um índice único maior que 255 (o índice também pode ser escrito em hexadecimal, usando o prefixo "0x"), o algoritmo usado na criação da chave (os grupos de algoritmos possíveis são diferentes para autenticação e para criptografia) e a chave AH da máquina com o IP de origem. A estrutura do comando que cria uma SA é apresentada abaixo:

```
add [ip_origem] [ip_destino] [protocolo] [índice] [algoritmo]
[chave];
```

Nesta experiência, haverá apenas a autenticação entre os hosts 1 e 2, portanto será necessário definir apenas duas SA's: uma para a autenticação dos pacotes enviados desta máquina para o host2 e outra para a autenticação dos pacotes enviados do host2 para esta máquina. Os parâmetros usados serão:

[ip_origem]: `2001:db8:beef::2`
[ip_destino]: `2001:db8:fada::2`
[protocolo]: `ah`
[índice]: `0x300` (o valor foi escolhido aleatoriamente)
[algoritmo]: `-A hmac-md5`
[chave]: a chave AH presente no arquivo `chave-ah-h1`, gerado anteriormente. Como este valor é hexadecimal, é necessário colocar o prefixo "0x" antes do valor.

Portanto, o comando assumirá a seguinte forma:

```
add 2001:db8:beef::2 2001:db8:fada::2 ah 0x300 -A hmac-md5
0x[chave-ah-h1];
```

A linha de comando para gerar a segunda SA, no sentido oposto de comunicação, será praticamente igual à linha anterior, porém com as seguintes diferenças: 1º) os IPs serão trocados de posição (o IP "`fada::2`" será a origem e o IP "`beef::2`" será o destino), 2º) o índice terá que ser diferente (por exemplo, `0x301`) e 3º) a chave AH será igualmente diferente (neste exemplo, a chave será o conteúdo do arquivo "`chave-ah-h2`"). Não esquecer de colocar o prefixo "0x" antes da chave. O comando assumirá a seguinte forma:

```
add 2001:db8:fada::2 2001:db8:beef::2 ah 0x301 -A hmac-md5
0x[chave-ah-h2];
```

No `nano`, escreva as duas linhas acima após a linha que contém o comando `spdflush`.

d. A seguir, definiremos as políticas de segurança desta máquina. Neste exemplo, usaremos o seguinte formato simplificado para a política:

```
spdadd [ips_origem] [ips_destino] [protocolo_camada_superior]
[política];
```

As políticas também são unidirecionais e precisam ser definidas aos pares se desejarmos uma segurança IPsec nos dois sentidos de comunicação. Os dois primeiros parâmetros são iguais aos das SA's, porém aqui eles podem definir tanto um único IP quanto um intervalo de IPs (por exemplo, pode-se definir como IPs de origem um prefixo como "2001:db8::/64"). O terceiro parâmetro, `[protocolo_camada_superior]`, define em quais protocolos das camadas superiores do TCP/IP o IPsec será aplicado (no nosso caso usaremos `any`, o que significa que esta política será aplicada aos pacotes de todos os protocolos cabíveis). O último parâmetro - `[política]` - define o que será feito com os pacotes enviados de `[ips_origem]` para `[ips_destino]` e que possuem o protocolo definido em `[protocolo_camada_superior]`. O campo começa com o parâmetro "-P", depois define a direção da comunicação (as opções são `out`, `in` e `fwd`, que significa forward), a seguir define as opções de tratamento dos pacotes, que são `discard`, `none` e `ipsec` (usaremos esta última) e por último define a regra pela qual os pacotes serão processados. No nosso exemplo, a regra será `ah/transport//require`. Para maiores informações sobre as opções disponíveis, consulte o manual do comando `setkey`, digitando o comando

abaixo num dos terminais desta simulação:

```
$ man setkey
```

Assim, os parâmetros usados para a primeira política serão:

[ips_origem]: 2001:db8:beef::2
[ips_destino]: 2001:db8:fada::2
[protocolo_camada_superior]: any
[política]: -P in ipsec ah/transport//require

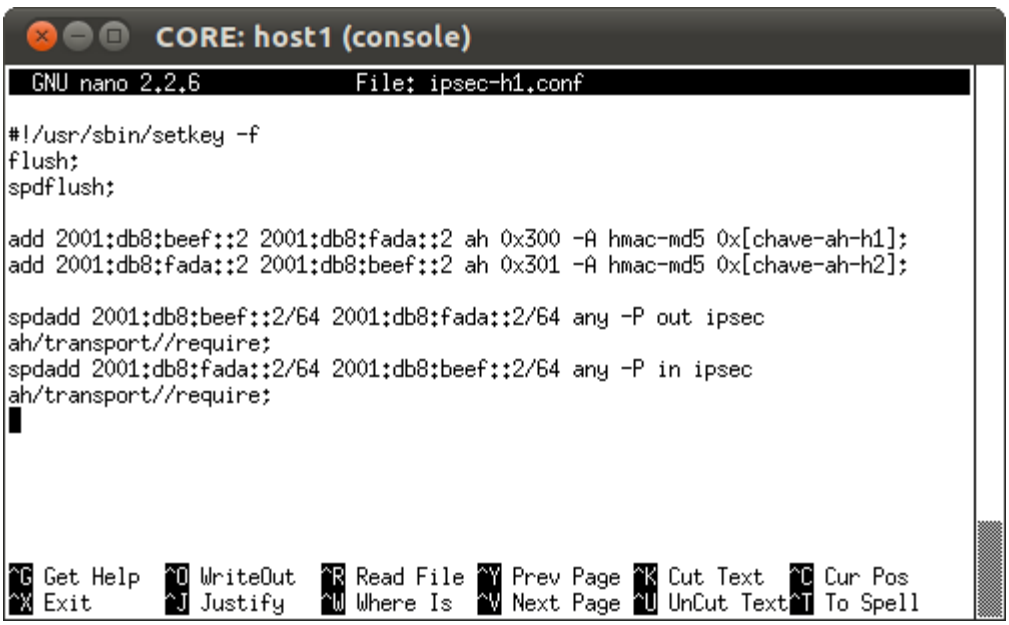
Portanto, o comando assumirá a seguinte forma:

```
spdadd 2001:db8:beef::2/64 2001:db8:fada::2/64 any -P out ipsec
ah/transport//require;
```

A linha de comando para gerar a segunda política, no sentido oposto de comunicação, será praticamente igual à linha anterior, porém com as seguintes diferenças: 1º) os IPs serão trocados de posição (o IP `fada::2` será a origem e o IP `beef::2` será o destino) e 2º) o campo `[política]` terá o sentido `out` no lugar de `in`. O comando assumirá a seguinte forma:

```
spdadd 2001:db8:fada::2/64 2001:db8:beef::2/64 any -P in ipsec
ah/transport//require;
```

No `nano`, escreva as duas linhas acima após a segunda linha de comando `add`. A tela final do `nano` deverá ser igual à imagem abaixo:



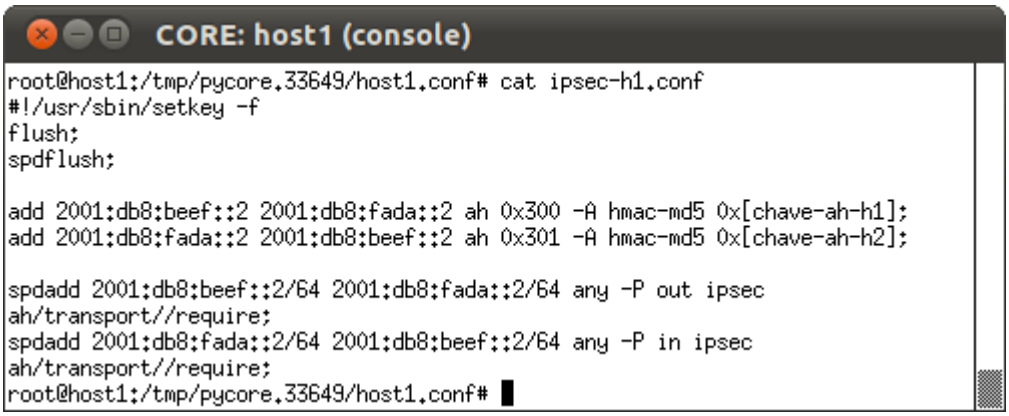
ATENÇÃO: trocar a parte `[chave-ah-h1]` e a parte `[chave-ah-h2]` pelas chaves reais, que estão armazenadas nos arquivos `chave-ah-h1` e `chave-ah-h2`!

e. Salve o arquivo e saia do `nano`, pressionando primeiro CTRL + O, depois ENTER e em seguida CTRL + X.

f. Imprima o conteúdo do arquivo:

```
$ cat ipsec-h1.conf
```

O conteúdo apresentado na tela deve ser igual à imagem abaixo:



12. Escreva o arquivo de configuração do IPsec para o host2.

a. Vá para o terminal do host2. Crie um arquivo ".conf" com o comando abaixo:

```
$ nano ipsec-h2.conf
```

Com este comando, será aberta a tela do editor de textos `nano`.

b. Digite as seguintes linhas na tela do `nano`:

```
#!/usr/sbin/setkey -f
flush;
spdflush;
```

c. Insira no arquivo duas linhas de comando de criação de SA, uma linha para cada uma das duas SA's de autenticação do host2, seguindo o modelo abaixo:

```
add [ip_origem] [ip_destino] [protocolo] [índice] [algoritmo]
[chave];
```

Observando os parâmetros acima, perceba como as SA's do host2 serão exatamente iguais às SA's do host1. De fato, a SA de saída de pacotes do host1 é exatamente igual à SA de entrada de pacotes do host2, e a de entrada de pacotes do host1 é igual à de saída de pacotes do host2. Porém, como a ordem de declaração das SA's não importa e as SA's não deixam explícito se o sentido é de entrada ou de saída, basta copiar as SA's do host1 para o arquivo ".conf" do host2. Assim, as duas linhas de comando de criação de SA's são apresentadas abaixo:

```
add 2001:db8:beef::2 2001:db8:fada::2 ah 0x300 -A hmac-md5
0x[chave-ah-h1];
add 2001:db8:fada::2 2001:db8:beef::2 ah 0x301 -A hmac-md5
0x[chave-ah-h2];
```

Escreva estas duas linhas abaixo do comando `spdflush`.

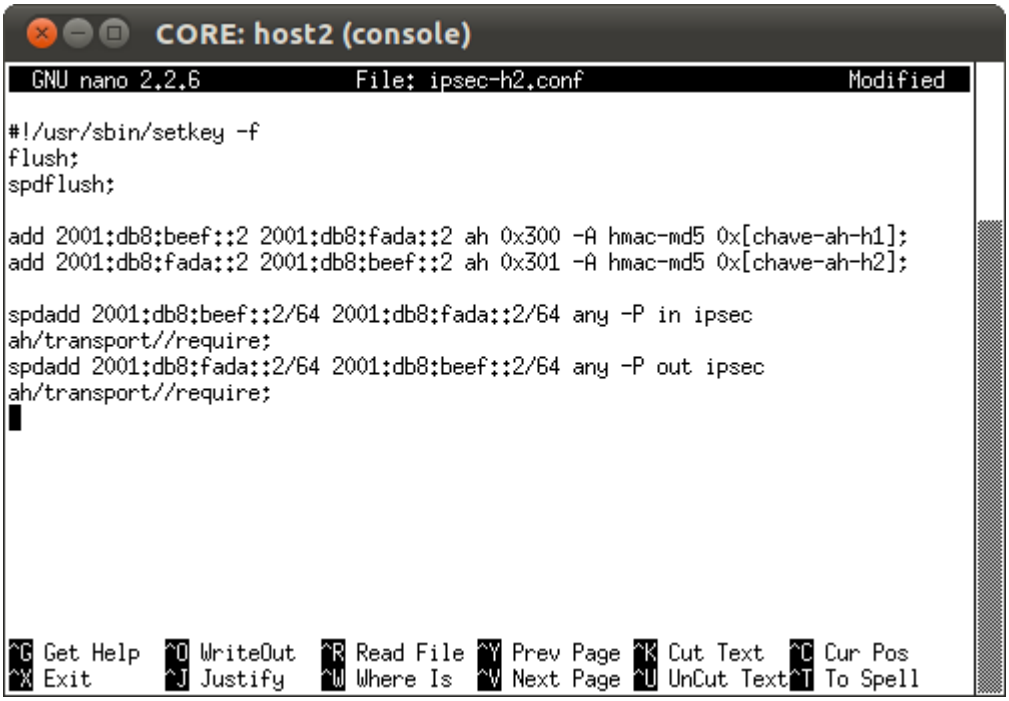
d. Insira no arquivo duas linhas de comando de criação de política de segurança, uma linha para cada uma das duas políticas de autenticação do host2, seguindo o modelo abaixo:

```
spdadd [ips_origem] [ips_destino] [protocolo_camada_superior]
[política];
```

Observe que, ao contrário das SA's, as políticas de segurança dizem explicitamente se o sentido de comunicação é de saída (`out`) ou de entrada (`in`). Tal declaração está dentro do parâmetro `[política]`. Observe também que, com exceção deste detalhe, as duas políticas do host2 serão iguais às duas políticas do host1. A diferença estará exatamente neste sentido de comunicação, que deve ser trocado nas duas políticas do host2. Portanto, para definir as políticas do host2 neste exemplo, basta copiar as políticas do host1 e inverter o sentido de comunicação (mudar para `out` onde for `in` e mudar para `in` onde for `out`). Assim, as duas linhas de comando de criação de políticas são apresentadas abaixo:

```
spdadd 2001:db8:beef::2/64 2001:db8:fada::2/64 any -P in ipsec
ah/transport//require;
spdadd 2001:db8:fada::2/64 2001:db8:beef::2/64 any -P out ipsec
ah/transport//require;
```

No `nano`, escreva as duas linhas acima após a segunda linha de comando `add`. A tela final do `nano` deverá ser igual à imagem abaixo:

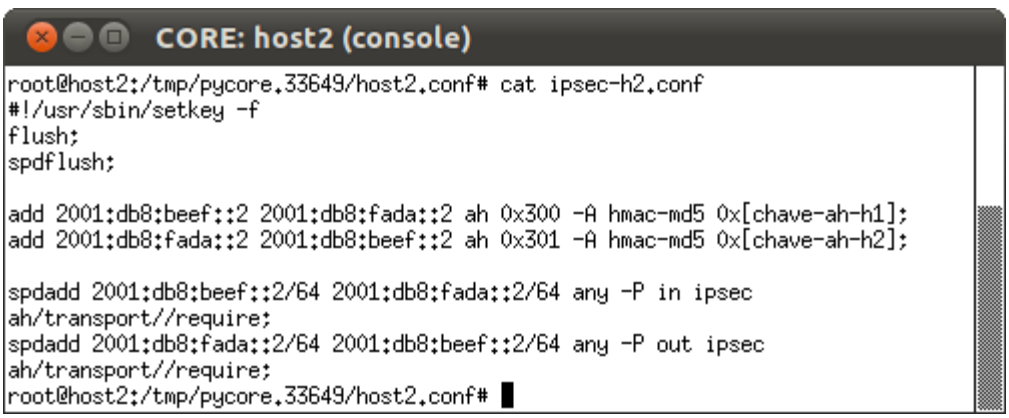


ATENÇÃO: trocar a parte `[chave-ah-h1]` e a parte `[chave-ah-h2]` pelas chaves reais, que estão armazenadas nos arquivos `chave-ah-h1` e `chave-ah-h2`!

e. Salve o arquivo e saia do `nano`, pressionando primeiro CTRL + O, depois ENTER e em seguida CTRL + X.

f. Imprima o conteúdo do arquivo:

```
$ cat ipsec-h2.conf
```



Preste atenção para as diferenças entre este arquivo e o arquivo `ipsec-h1.conf`. As únicas diferenças são o sentido de comunicação `in/out` explicitado nas políticas de comunicação. Isto significa que o processo de escrita dos dois arquivos ".conf" se limita a escrever apenas um deles, copiá-lo para a outra máquina e alterar o `in/out` nas políticas `spdadd`.

13. Vá para o terminal do host1 e carregue as configurações do arquivo `ipsec-h1.conf` com o comando abaixo:

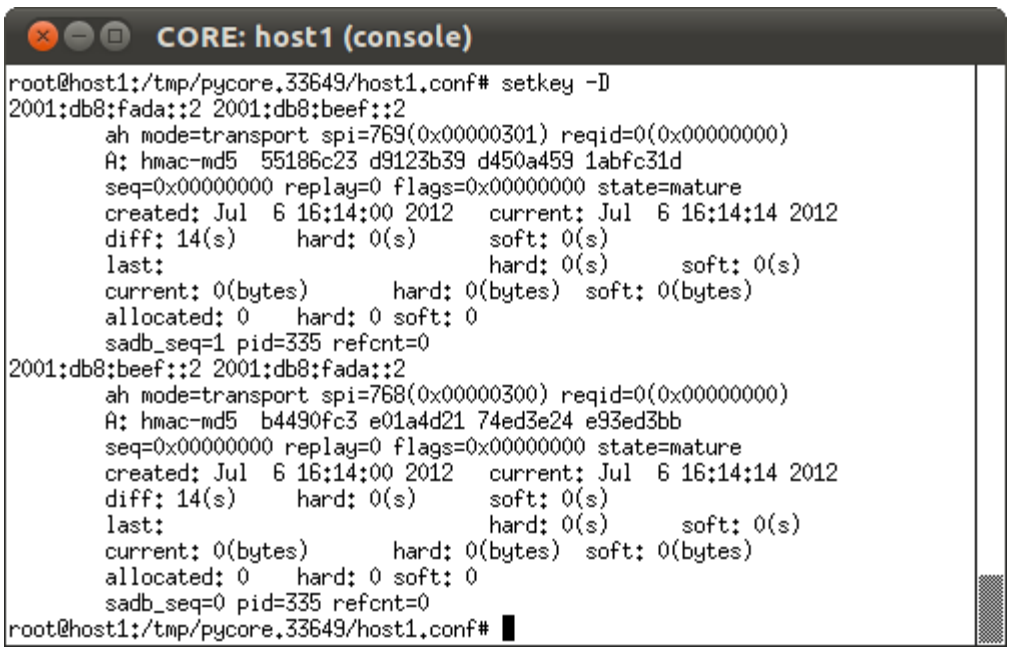
```
$ setkey -f ipsec-h1.conf
```

Se o arquivo for carregado corretamente, nenhuma mensagem será impressa após a execução do comando `setkey -f`. Caso surja alguma mensagem, o arquivo possuirá algum erro de sintaxe. Neste caso, verifique novamente os comandos descritos no passo 11.

Para verificar se as chaves foram carregadas, execute os seguintes comandos:

```
$ setkey -D
```

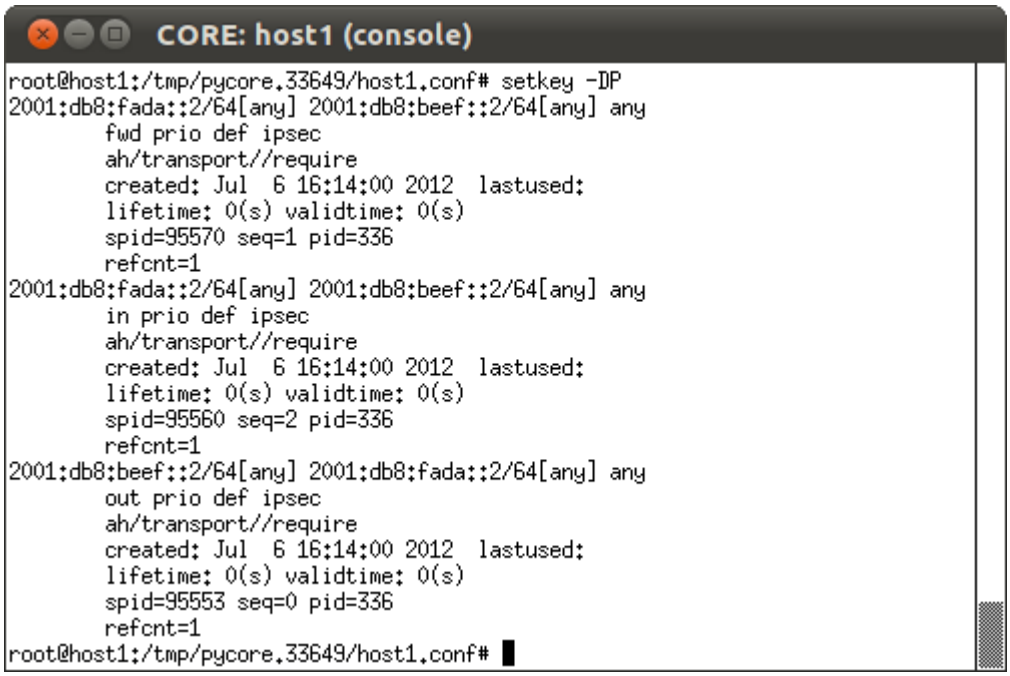
Este comando exibe as SA's que a máquina possui. Caso as SA's tenham sido corretamente carregadas, a seguinte tela será impressa no terminal:



Já o comando abaixo exibe as políticas de segurança implementadas.

```
$ setkey -DP
```

A seguinte tela será impressa no terminal:



14. Vá para o terminal do host2 e carregue as configurações do arquivo ".conf". Supondo que você tenha nomeado este arquivo como `ipsec-h2.conf`, o comando ficará igual ao abaixo:

```
$ setkey -f ipsec-h2.conf
```

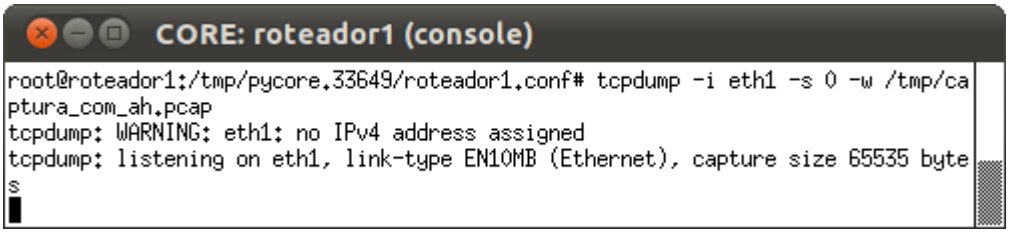
Para verificar se as chaves foram carregadas, execute os seguintes comandos:

```
$ setkey -D
$ setkey -DP
```

15. Reenvie pacotes entre os dois roteadores para análise do cabeçalho. Com as configurações do IPsec em execução, repita o procedimento de captura de pacotes.

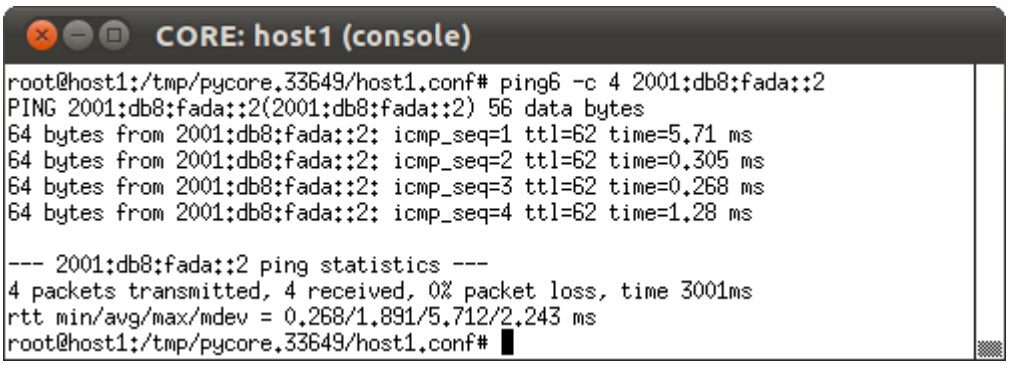
a. No terminal do roteador1:

```
$ tcpdump -i eth1 -s 0 -w /tmp/captura_com_ah.pcap
```

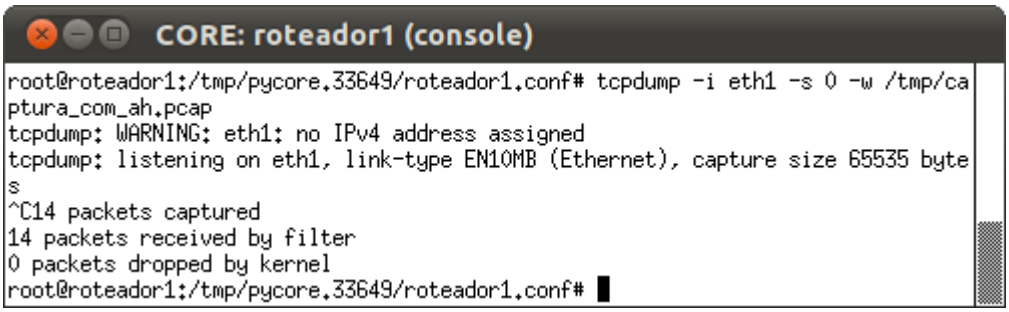


b. No terminal do host1:

```
$ ping6 -c 4 2001:db8:fada::2
```



c. No terminal do roteador1, encerre a captura de pacotes pressionando Ctrl+C.



Agora implementaremos a criptografia entre os dois terminais, em adição à autenticação já implementada. Para isso, alteraremos os arquivos ".conf" já gerados e os recarregaremos. Em seguida, faremos uma terceira captura de pacotes, para análise posterior.

16. Imprima e anote as chaves ESP geradas anteriormente para os hosts 1 e 2:

a. Vá para o terminal do host1 e imprima a chave ESP armazenada no arquivo `chave-esp-h1`:

```
$ cat chave-esp-h1
```

Uma tela similar à abaixo deverá ser impressa:

Anote o valor apresentado. Perceba como o valor é consideravelmente maior que o valor da chave AH. Contudo, o valor também é hexadecimal, e deverá ter o prefixo "0x" dentro do arquivo ".conf".

b. Vá para o terminal do host2 e imprima a chave ESP armazenada no arquivo `chave-esp-h2`:

```
$ cat chave-esp-h2
```

Anote o valor apresentado na tela.

17. Altere o arquivo de configuração `ipsec-h1.conf` para incluir a configuração da criptografia:

a. Vá para o terminal do host1 e abra o arquivo de configuração `ipsec-h1.conf`:

```
$ nano ipsec-h1.conf
```

b. Criaremos agora as SA's correspondentes ao processo de criptografia para o host1. Após o segundo comando `add` já presente no arquivo, insira mais duas linhas de comando `add`, que definirão mais um par de SA's. Para referência, reapresentamos a estrutura do comando `add`, que cria uma SA:

```
add [ip_origem] [ip_destino] [protocolo] [índice] [algoritmo]
[chave];
```

Os parâmetros usados serão:
[ip_origem]: 2001:db8:beef::2
[ip_destino]: 2001:db8:fada::2
[protocolo]: `esp`
[índice]: `0x302` (precisa ser diferente dos já existentes, 0x300 e 0x301)
[algoritmo]: `-E 3des-cbc`
[chave]: chave ESP presente no arquivo `chave-esp-h1`, gerado anteriormente. Este valor também é hexadecima, logo é necessário colocar o prefixo "0x" antes da chave.
Portanto, o comando assumirá a seguinte forma:

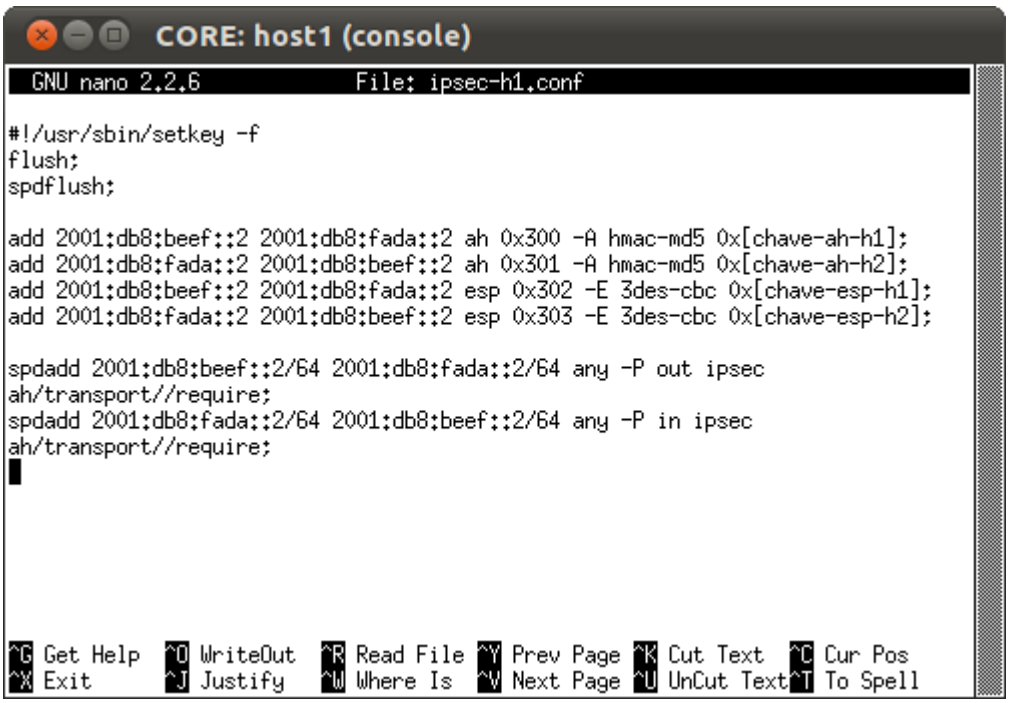
```
add 2001:db8:beef::2 2001:db8:fada::2 esp 0x302 -E 3des-cbc
0x[chave-esp-h1];
```

A linha de comando para gerar a segunda SA, no sentido oposto de comunicação, será praticamente igual à linha anterior, porém com as seguintes diferenças: 1º) os IPs serão trocados de posição (o IP "`fada::2`" será a origem e o IP "`beef::2`" será o destino), 2º) o índice terá que ser diferente (por exemplo, `0x303`) e 3º) a chave ESP será igualmente diferente

(neste exemplo, a chave será o conteúdo do arquivo `chave-esp-h2`). O comando assumirá a seguinte forma:

```
add 2001:db8:fada::2 2001:db8:beef::2 esp 0x303 -E 3des-cbc
0x[chave-esp-h2];
```

O arquivo `ipsec-h1.conf` apresentará, neste momento, o seguinte conteúdo na tela do `nano`:



c. A seguir, atualizaremos as políticas de segurança desta máquina para incluir a criptografia. Ao contrário dos comandos `add`, que definem apenas uma SA cada, os comandos `spdadd` podem definir mais de um protocolo de segurança como parte da política, numa única linha de comando. Lembrando a estrutura do comando `spdadd`:

```
spdadd [ips_origem] [ips_destino] [protocolo_camada_superior]
[política];
```

A política é definida após o parâmetro `[protocolo_camada_superior]`, e apresenta a seguinte forma no arquivo:

```
-P in ipsec
ah/transport//require
```

Para incluir um segundo protocolo de segurança, basta definir o protocolo adicional antes do primeiro, o que fará com que o parâmetro `[política]` assuma a seguinte forma:

```
-P in ipsec
esp/transport//require
ah/transport//require
```

Observação: a ordem em que os protocolos esp e ah são declarados é importante e deve ser respeitada, isto é, o protocolo esp deve ser declarado primeiro e em seguida o protocolo ah é declarado.

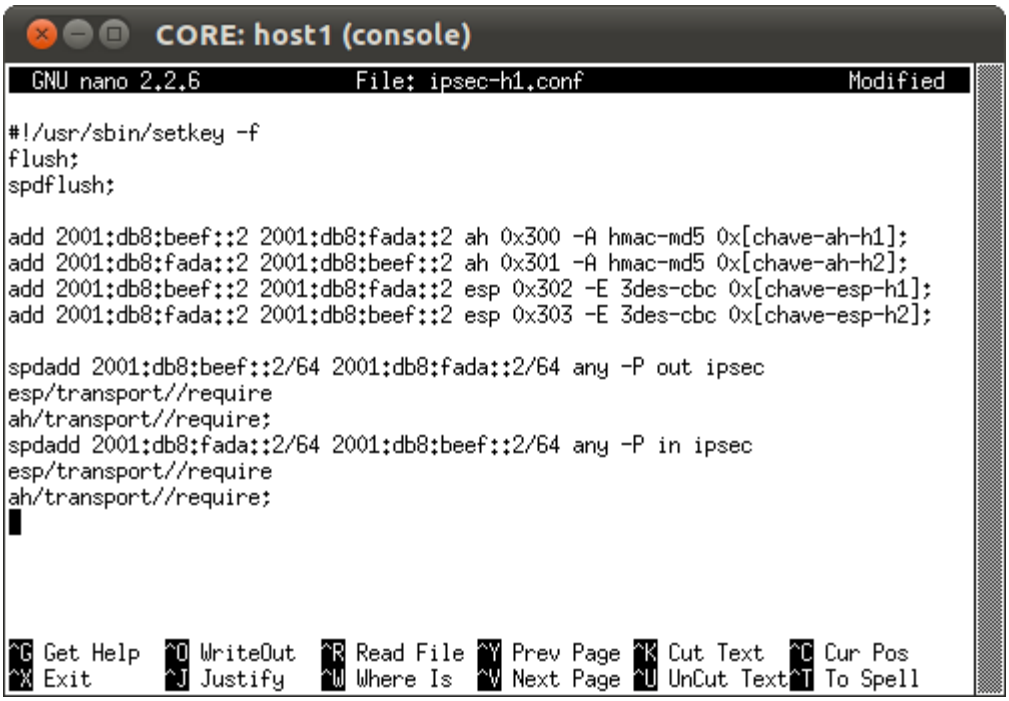
Logo, o comando `spdadd` atualizado assumirá a seguinte forma:

```
spdadd 2001:db8:beef::2/64 2001:db8:fada::2/64 any -P out ipsec
esp/transport//require
ah/transport//require;
```

Altere o segundo comando `spdadd` incluindo o mesmo trecho discutido acima. O comando assumirá a seguinte forma:

```
spdadd 2001:db8:fada::2/64 2001:db8:beef::2/64 any -P in ipsec
esp/transport//require
ah/transport//require;
```

O arquivo `ipsec-h1.conf` apresentará a seguinte tela final no `nano`:



d. Salve o arquivo e saia do `nano`, pressionando primeiro CTRL + O, depois ENTER e em seguida CTRL + X.

e. Imprima o conteúdo do arquivo:

```
$ cat ipsec-h1.conf
```

O conteúdo apresentado na tela deve ser similar à imagem abaixo (as chaves serão diferentes, pois serão geradas por você):

18. Atualize o arquivo de configuração do IPsec para o host2 para incluir a criptografia.

a. Vá para o terminal do host2 e abra o arquivo de configuração `ipsec-h2.conf`:

```
$ nano ipsec-h2.conf
```

b. Após o segundo comando `add` já presente no arquivo, insira mais duas linhas de comando `add`, que definirão o novo par de SA's para a criptografia. Lembrando que o par de SA's é idêntico para os dois hosts, copie as SA's de criptografia apresentadas no passo 17 (que serão reapresentadas abaixo):

```
add 2001:db8:beef::2 2001:db8:fada::2 esp 0x302 -E 3des-cbc
0x[chave-esp-h1];
add 2001:db8:fada::2 2001:db8:beef::2 esp 0x303 -E 3des-cbc
0x[chave-esp-h2];
```

c. A seguir, atualizaremos as políticas de segurança desta máquina para incluir a criptografia. Altere os comandos `spdadd` para incluir o texto `esp/transport//require` antes do texto `ah/transport//require`. Da mesma maneira, troque o parâmetro `out` por `in` no primeiro comando e troque o parâmetro `in` por `out` no segundo. Os comandos ficarão conforme o código abaixo (as diferenças estão em negrito):

```
spdadd 2001:db8:beef::2/64 2001:db8:fada::2/64 any -P in ipsec
esp/transport//require
ah/transport//require;
spdadd 2001:db8:fada::2/64 2001:db8:beef::2/64 any -P out ipsec
esp/transport//require
ah/transport//require;
```

d. Salve o arquivo e saia do `nano`, pressionando primeiro CTRL + O, depois ENTER e em seguida CTRL + X.

e. Imprima o conteúdo do arquivo:

```
$ cat ipsec-h2.conf
```

O conteúdo apresentado na tela deve ser similar à imagem abaixo (as chaves serão diferentes, pois serão geradas por você):

Perceba mais uma vez que a única diferença entre o conteúdo do `ipsec-h2.conf` e o conteúdo do `ipsec-h1.conf` é a inversão da direção de comunicação nas políticas de segurança (onde era `out` torna-se `in` e onde era `in` torna-se `out`).

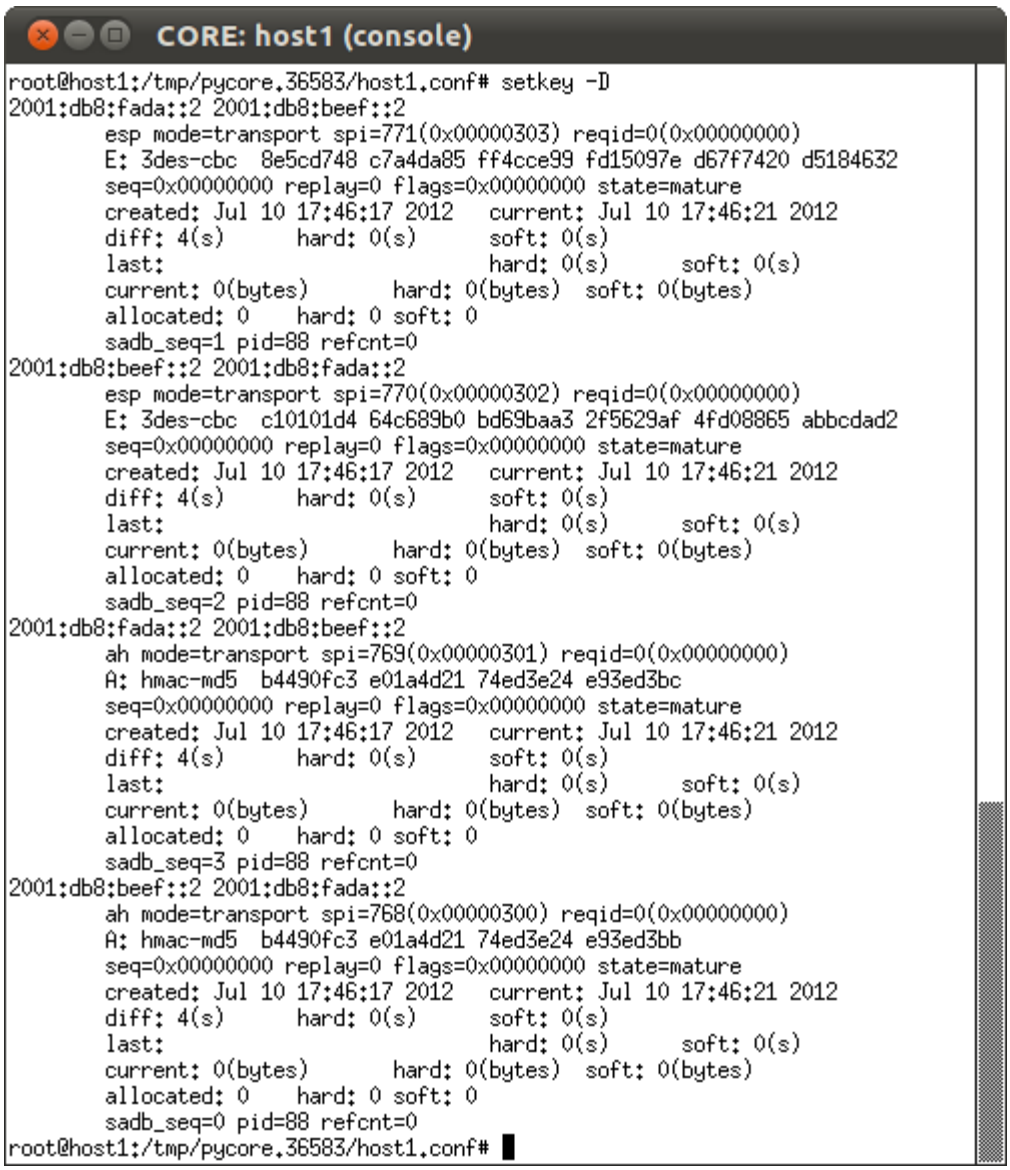
19. Vá para o terminal do host1 e recarregue as configurações do arquivo `ipsec-h1.conf` com o comando abaixo:

```
$ setkey -f ipsec-h1.conf
```

Para verificar se as chaves foram carregadas, execute os seguintes comandos:

```
$ setkey -D
```

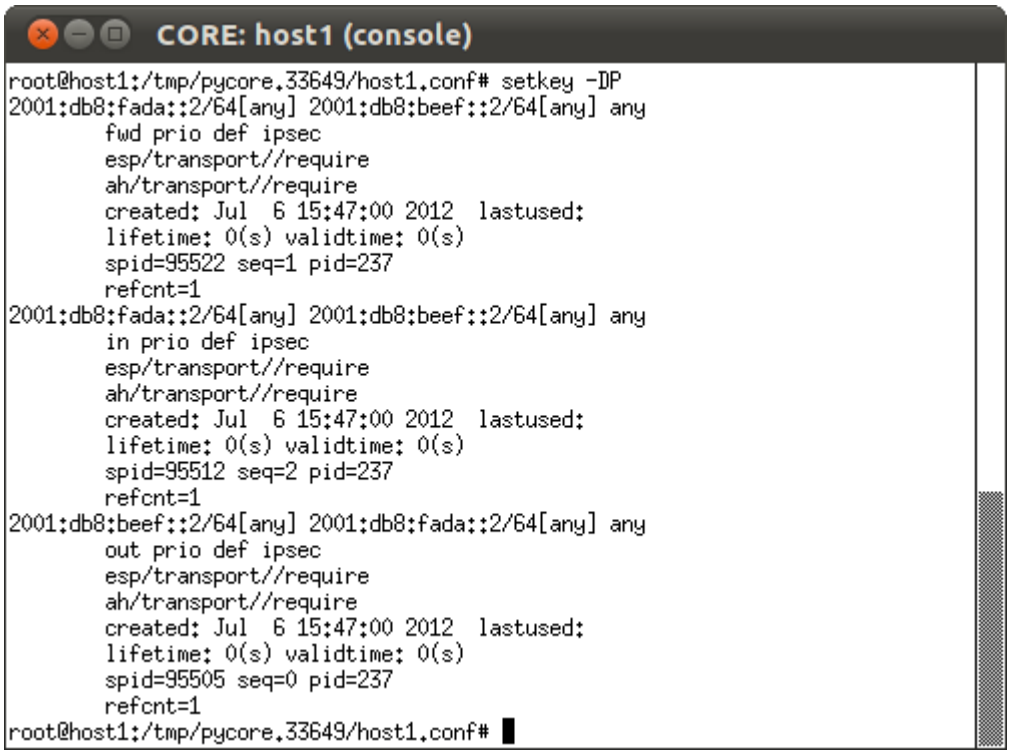
A seguinte tela será exibida:



Execute também o seguinte comando:

```
$ setkey -DP
```

A seguinte tela será exibida:



20. Vá para o terminal do host2 e recarregue as configurações do arquivo `ipsec-h2.conf` com o comando abaixo:

```
$ setkey -f ipsec-h2.conf
```

Para verificar se as chaves foram carregadas, execute os seguintes comandos:

```
$ setkey -D
$ setkey -DP
```

Uma tela similar às exibidas no passo anterior também será exibida após estes comandos.

21. Reenvie pacotes entre os dois roteadores para análise do cabeçalho.

Com as configurações do IPsec em execução, repita o procedimento de captura de pacotes.

a. No terminal do roteador1:

```
$ tcpdump -i eth1 -s 0 -w /tmp/captura_com_ah_esp.pcap
```

```
CORE: roteador1 (console)
root@roteador1:/tmp/pycore.33649/roteador1.conf# tcpdump -i eth1 -s 0 -w /tmp/ca
ptura_com_ah_esp.pcap
tcpdump: WARNING: eth1: no IPv4 address assigned
tcpdump: listening on eth1, link-type EN10MB (Ethernet), capture size 65535 byte
s
```

b. No terminal do host1:

```
$ ping6 -c 4 2001:db8:fada::2
```

```
CORE: host1 (console)
root@host1:/tmp/pycore.33649/host1.conf# ping6 -c 4 2001:db8:fada::2
PING 2001:db8:fada::2(2001:db8:fada::2) 56 data bytes
64 bytes from 2001:db8:fada::2: icmp_seq=1 ttl=62 time=7.14 ms
64 bytes from 2001:db8:fada::2: icmp_seq=2 ttl=62 time=0.475 ms
64 bytes from 2001:db8:fada::2: icmp_seq=3 ttl=62 time=0.619 ms
64 bytes from 2001:db8:fada::2: icmp_seq=4 ttl=62 time=0.681 ms

--- 2001:db8:fada::2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 0.475/2.230/7.146/2.839 ms
root@host1:/tmp/pycore.33649/host1.conf#
```

c. No terminal do roteador1, encerre a captura de pacotes pressionando Ctrl+C.

```
CORE: roteador1 (console)
root@roteador1:/tmp/pycore.33649/roteador1.conf# tcpdump -i eth1 -s 0 -w /tmp/ca
ptura_com_ah_esp.pcap
tcpdump: WARNING: eth1: no IPv4 address assigned
tcpdump: listening on eth1, link-type EN10MB (Ethernet), capture size 65535 byte
s
^C24 packets captured
24 packets received by filter
0 packets dropped by kernel
root@roteador1:/tmp/pycore.33649/roteador1.conf#
```

22. Encerre a simulação:

a. aperte o botão  ; ou
b. utilize o menu Experiment > Stop.

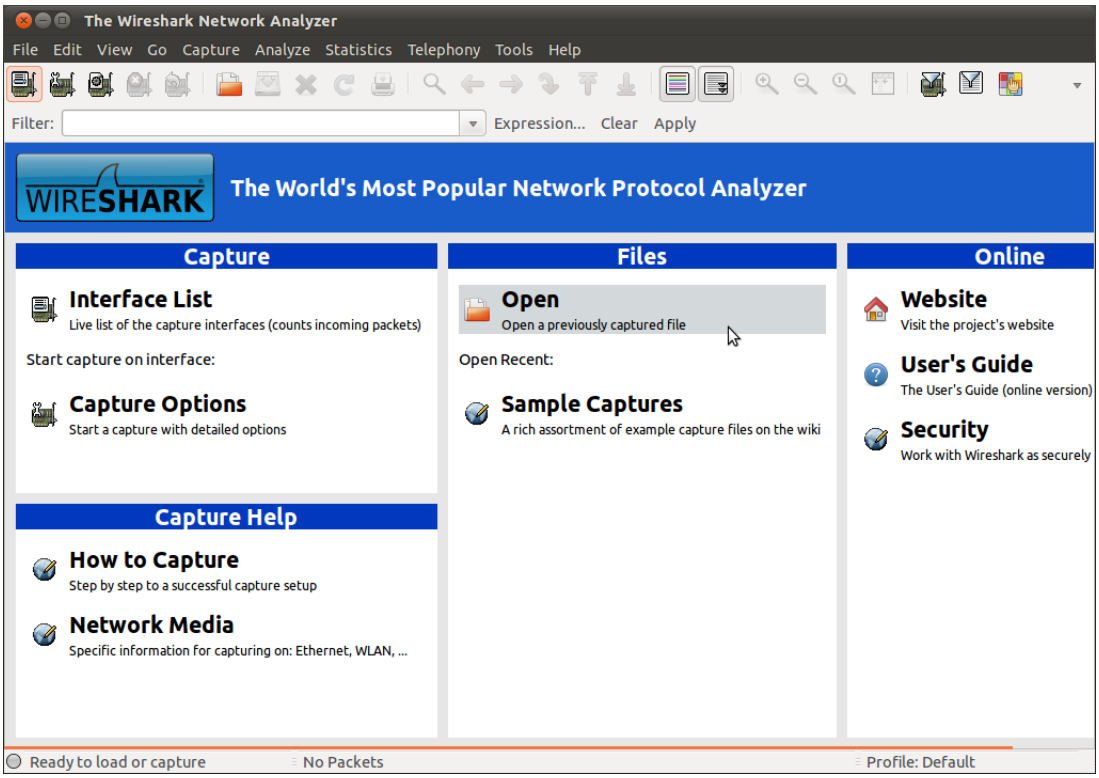
23. Analisaremos agora os pacotes capturados nas três situações:

- Topologia sem IPsec
- Topologia com somente autenticação
- Topologia com autenticação e criptografia

Para verificar os pacotes capturados, utilizaremos o programa Wireshark.

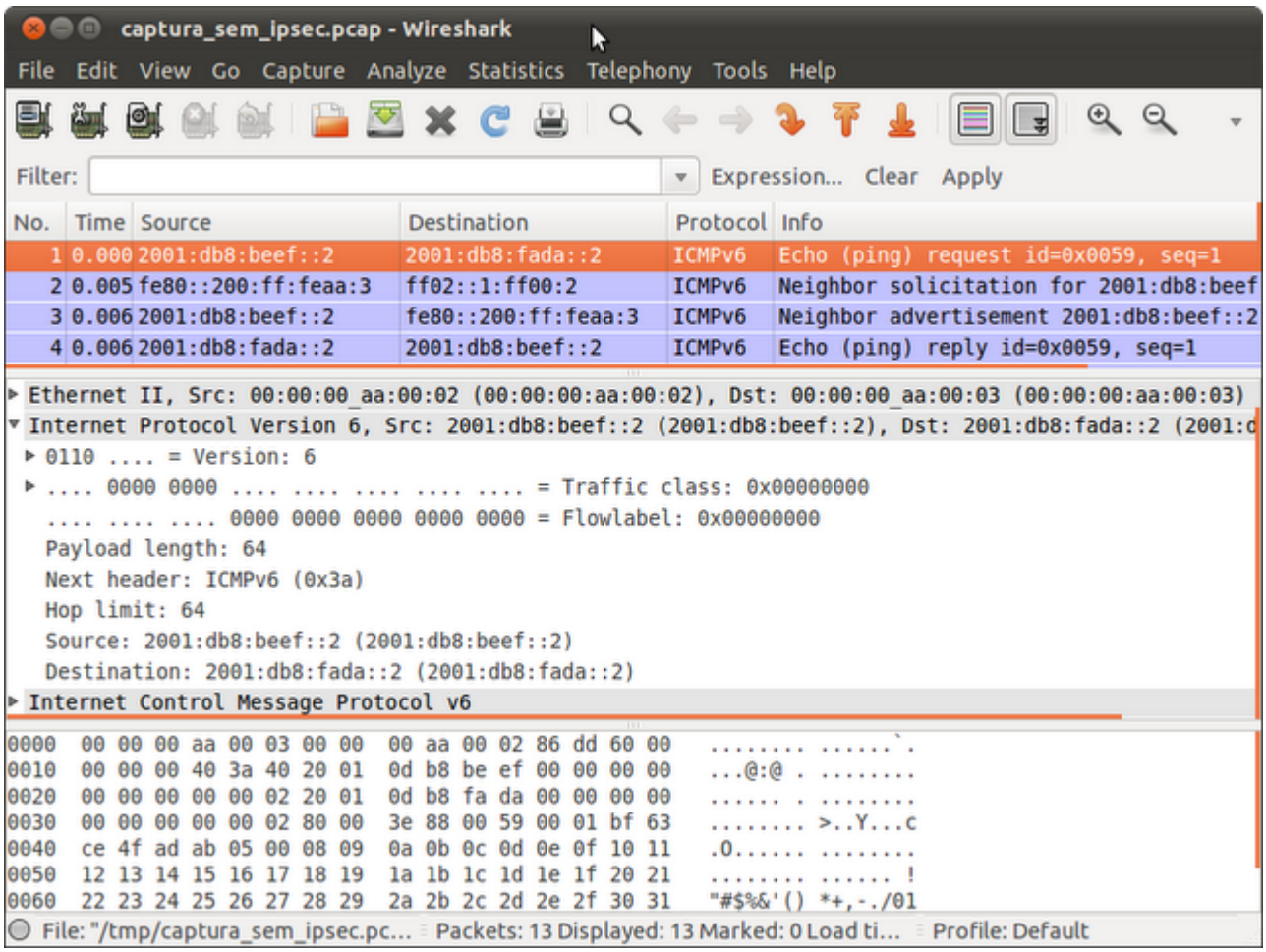
a. Inicie o programa Wireshark. Uma maneira de fazê-lo é através de um terminal na máquina virtual, com o comando:

```
$ wireshark
```



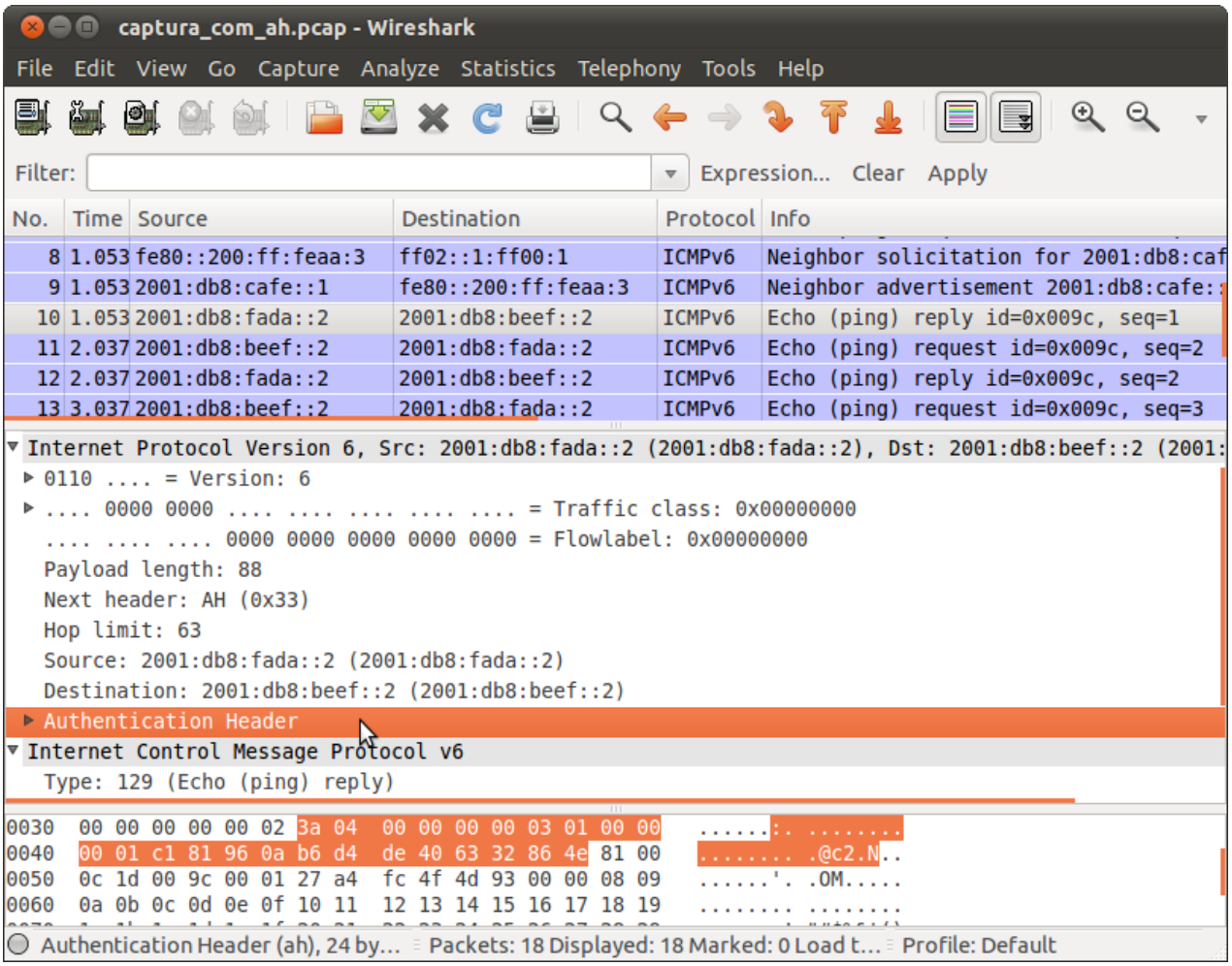
Este programa tem como principais funcionalidades a captura e análise de pacotes transmitidos por uma interface de rede. Através de seu uso, é possível visualizar melhor os pacotes que trafegam pela rede. Verifique o arquivo de captura previamente obtido.

b. Abra o arquivo `/tmp/captura_sem_ipsec.pcap` com o menu File>Open:



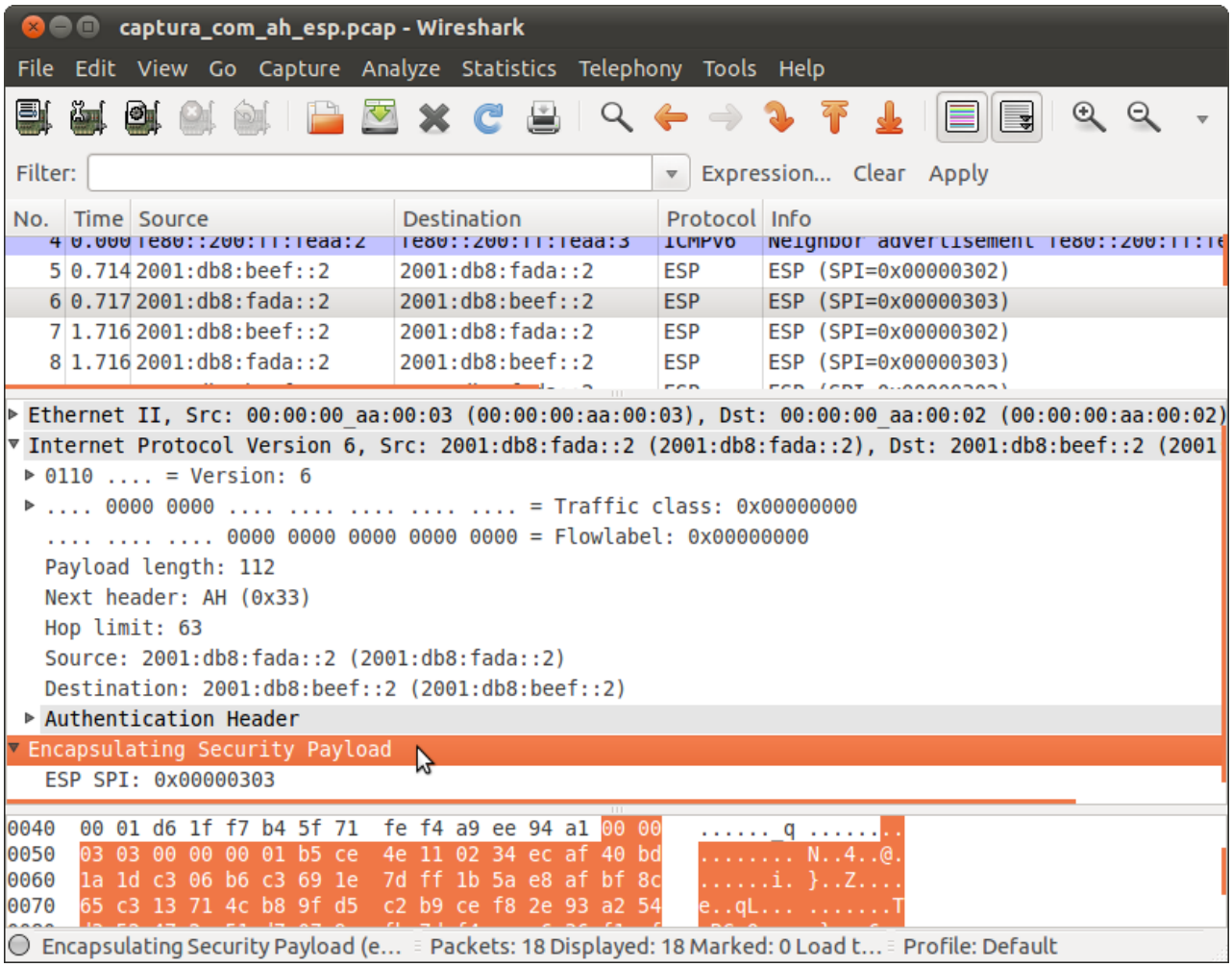
c. Analise um pacote ICMPv6 de Echo (ping) reply de `2001:db8:beef::2` para `2001:db8:fada::2`. Note que é possível analisar todo o conteúdo do pacote, inclusive o conteúdo do campo “data”. Caso o pacote Echo (ping) request fosse falsificado, com o dono se fazendo passar pelo dispositivo `2001:db8:beef::2`, a resposta seria enviada mesmo assim.

d. Abra o arquivo `/tmp/captura_com_ah.pcap` com o menu File>Open:



e. Analise um pacote de `2001:db8:beef::2` para `2001:db8:fada::2`. Note o cabeçalho de extensão AH (Authentication Header) que aparece no final do cabeçalho IP (Internet Protocol Version 6). Embora o conteúdo do pacote esteja visível para qualquer um que conseguir capturá-lo (seja o destinatário do pacote, seja um "sniffer" no meio da rede), o destinatário só responderá o pacote se o remetente possuir a chave de autenticação. Isso significa que embora os dados não sejam confidenciais, eles são confiáveis e íntegros (garante-se que a origem do pacote não foi forjada e que o pacote não foi modificado). A utilização de somente autenticação se justifica em casos onde o conteúdo não necessita ser protegido, mas é preciso garantir que o pacote foi enviado por um dispositivo autorizado. Utilizar somente autenticação também é uma técnica utilizada por dispositivos com capacidade de processamento limitada que não teriam capazes de processar pacotes criptografados em tempo hábil, uma vez que os algoritmos de criptografia necessitam de bastante processamento.

f. Abra o arquivo `/tmp/captura_com_ah_esp.pcap` com o menu File>Open:



g. Note que não é possível analisar todo o conteúdo do pacote, que está criptografado no ESP na camada de aplicação. Inclusive, é impossível saber que o pacote em questão é um pacote de Echo (ping) reply. Neste exemplo sabe-se o tipo do pacote apenas porque forçamos a geração dos mesmos através do ping. Outro ponto interessante é que para a camada de aplicação (programas) a criptografia dos pacotes é transparente e não afeta o funcionamento destas, pois elas recebem e enviam pacotes sem qualquer criptografia ou autenticação, uma vez que estas são geradas e removidas pela camada IP. Note também a existência do cabeçalho de autenticação AH, que impede que um pacote falsificado seja tratado e respondido pelo destino, que conhece a chave do dispositivo verdadeiro. Portanto, os dados deste pacote são confidenciais, confiáveis e íntegros.

Experiência 3.1 - IPsec: Modo de Transporte - Detalhamento de comandos

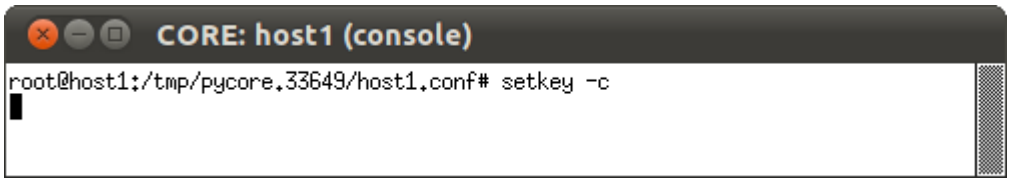
1. **Configuração manual por entrada direta de comandos**

   Para configurar o IPsec no Linux, utiliza-se o comando `setkey`. Conforme já mencionado, o comando `setkey` pode executar rotinas de manipulação dos bancos de dados tanto por entrada direta dos comandos quanto a partir de um arquivo de configuração (nome_do_arquivo.conf). A seguir, segue-se uma explicação de como realizar a configuração por entrada direta de comandos.

   Para ativar o `setkey` neste modo, usa-se o comando com o formato abaixo:

   ```
   $ setkey -c
   ```

   Executar este comando gera a seguinte tela:

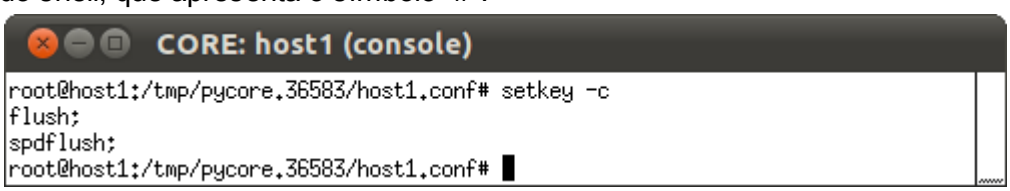
   

   Ao executá-lo, o terminal abre uma sessão própria do `setkey` para entrada dos comandos relacionados ao IPsec, isto é, uma sessão onde apenas comandos do `setkey` serão válidos (note que a linha após o comando `setkey -c` não começa com o símbolo "#", que indicaria que ainda estaríamos numa sessão do terminal do Linux). Após abrir esta sessão, digita-se um comando desejado, adiciona-se o caractere ";" e então pressiona-se ENTER. Para cada comando que se deseja executar, deve-se repetir o processo. Após digitar o último comando e apertar ENTER, o usuário poderá finalizar a sessão pressionando as teclas CTRL+D. Após isso, os comandos digitados serão executados.

   Embora seja trabalhoso (e propenso a erros) fazer a configuração manual completa do IPsec através da entrada direta de comandos, este método é útil quando desejamos apenas limpar as configurações do IPsec. Para fazê-lo, digita-se as seguintes linhas, pressionando ENTER no final de cada uma delas:

   ```
   flush;
   spdflush;
   ```

   Então pressiona-se CTRL+D para finalizar a sessão e o terminal volta para a sessão do shell, que apresenta o símbolo "#":

   ATENÇÃO: as alterações do IPsec executadas por entrada direta de comandos

afetam somente a sessão atual do sistema operacional. Uma vez que o SO seja reinicializado, estas configurações serão perdidas e o sistema carregará as configurações escritas no arquivo de configuração "/etc/ipsec-tools.conf". Caso este arquivo esteja em branco, não exista ou esteja totalmente comentado, nenhuma configuração nova será carregada e o IPsec não será aplicado a nenhum pacote.

2. **Configurações permanentes: inicializando, suspendendo e alterando em tempo de execução**

   Conforme já foi explicado, para aplicar uma configuração permanente no IPsec, é necessário colocar as configurações dentro do arquivo "/etc/ipsec-tools.conf". Para abrir este arquivo para escrita e alteração, execute o comando abaixo:

```
$ sudo nano /etc/ipsec-tools.conf
```

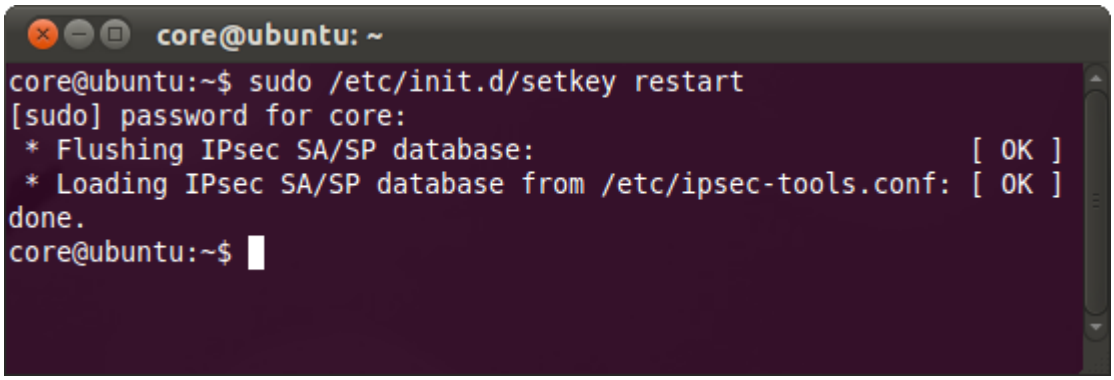
   OBS: até o momento, não utilizamos o prefixo `sudo` em nenhum comando de configuração do IPsec porque dentro do Core os terminais são abertos com o usuário `root`, o qual possui privilégios de alteração e execução de qualquer arquivo protegido. Todos os arquivos ".conf" que estão presentes dentro da pasta `/etc` e suas subpastas são arquivos protegidos. Logo, caso você esteja configurando o IPsec em sua máquina real, não esqueça de colocar o prefixo `sudo` em todos os comandos relacionados ao IPsec, seja o `setkey`, seja o próprio `nano` na hora de alterar um arquivo protegido do sistema (como o "/etc/ipsec-tools.conf").

   Todos os detalhes de configuração do IPsec que foram discutidos na experiência 1 valem para este arquivo. Após alterá-lo, as configurações não serão automaticamente carregadas, sendo necessário reinicializar o sistema primeiro. Porém, haverá situações onde o sistema não poderá ser desligado e as alterações terão que ser feitas e aplicadas dentro da mesma sessão do SO. Para trabalhar com estas situações, o usuário deverá utilizar o script de inicialização do IPsec presente dentro da pasta `/etc/init.d/`. Através deste script, é possível inicializar, reinicializar e suspender o IPsec dentro de uma mesma sessão do Linux.

   Considerando novamente a situação onde o usuário acabou de escrever as configurações do IPsec no arquivo "/etc/ipsec-tools.conf" e deseja aplicá-las imediatamente sem reinicializar o sistema operacional, deve-se utilizar o comando abaixo:

```
$ sudo /etc/init.d/setkey restart
```

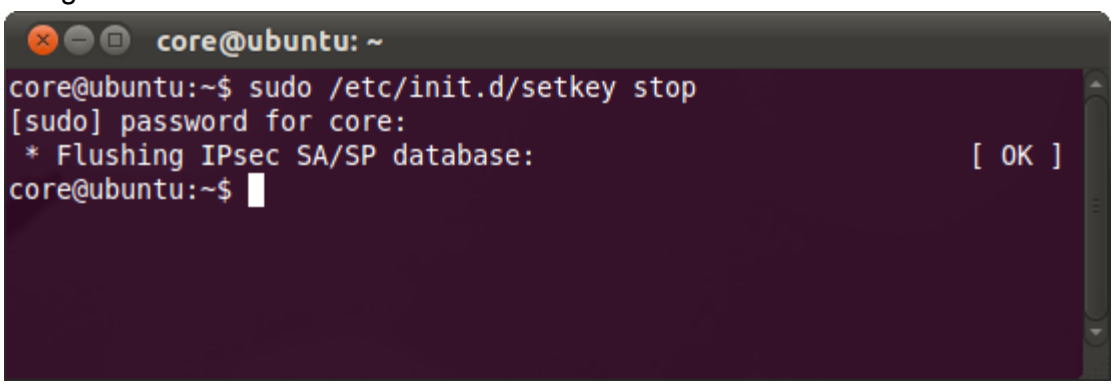
Este comando exibirá a seguinte tela:



Uma outra situação que poderá ocorrer é a necessidade de suspender as configurações do IPsec temporariamente e reestabelecê-las depois, isto é, desfazer as configurações sem que elas sejam perdidas permanentemente. Para fazer isto, execute o comando abaixo:

```
$ sudo /etc/init.d/setkey stop
```

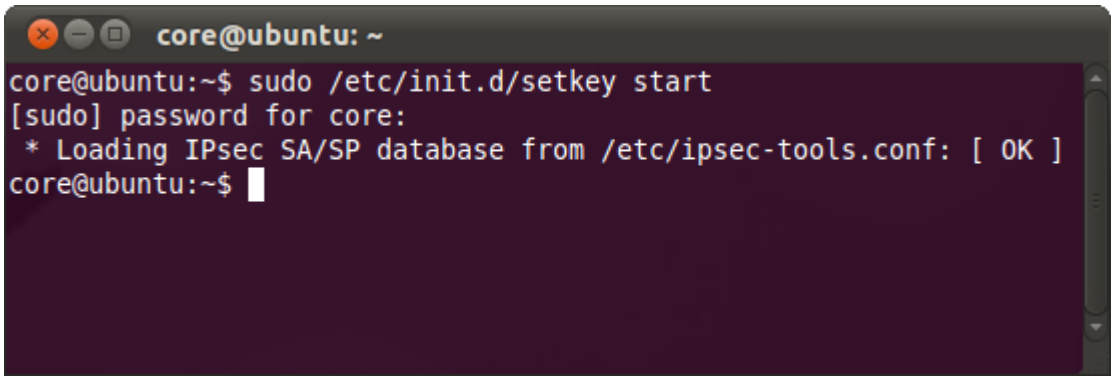
A seguinte tela será exibida:



Caso se deseje reestabelecer as configurações do IPsec, execute o comando abaixo:

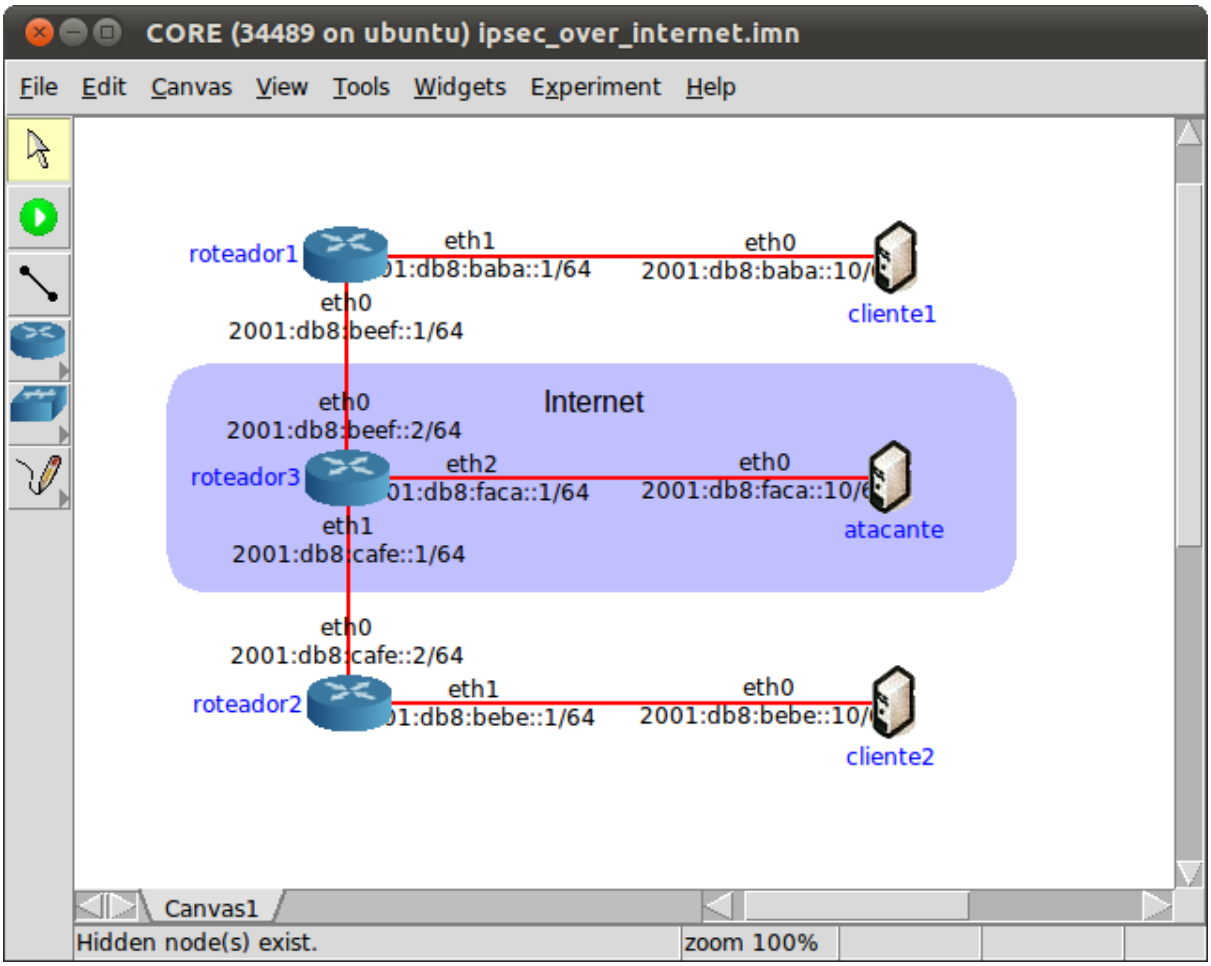
```
$ sudo /etc/init.d/setkey stop
```

A seguinte tela será exibida:



Experiência 4 - IPsec: Modo de Túnel

1. No CORE, abra o arquivo **"seguranca-ipsec-tunel.imn"**. A seguinte topologia inicial de rede deve aparecer deve aparecer:



Comparando com a topologia do experimento anterior, foram adicionados o roteador3 e o dispositivo atacante para simular a Internet, além de um dispositivo cliente para cada roteador da topologia original.

2. Inicie a simulação:

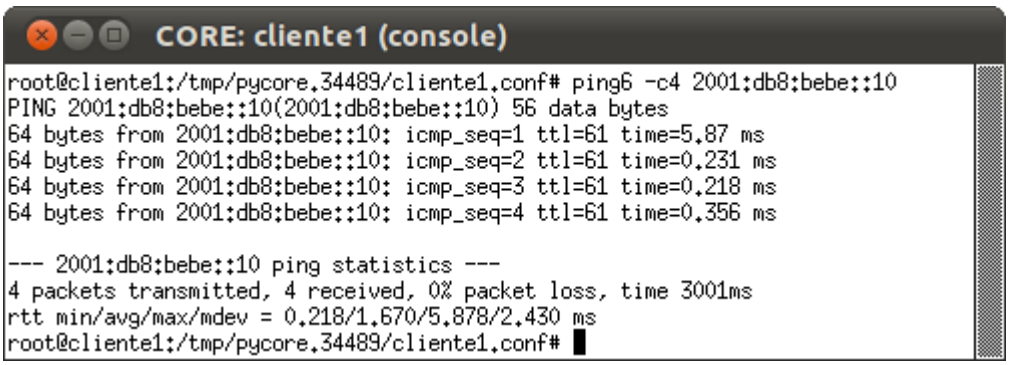
   a. aperte o botão  ;
   ou utilize o menu Experiment > Start.

   a. Espere até que o CORE termine a inicialização da simulação.

   b. Abra o terminal do cliente1, através de duplo-clique. Verifique a conectividade entre o cliente1 e o cliente2 através do seguinte comando:

```
$ ping6 -c4 2001:db8:bebe::10
```

O resultado deve ser:

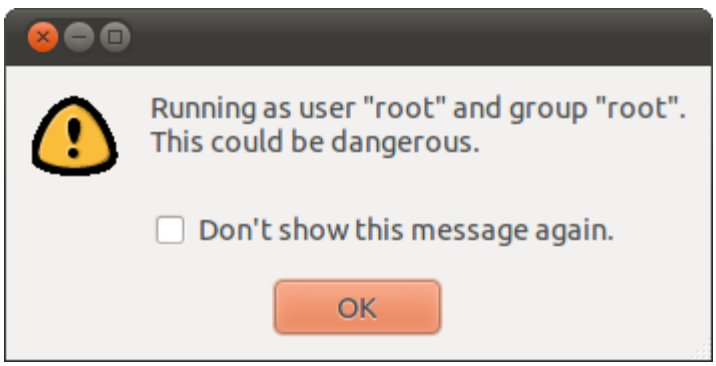


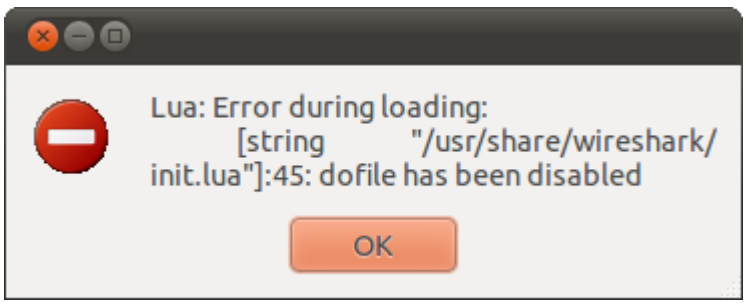
Este resultado mostra que a conexão entre o cliente1 e o cliente2 está funcionando corretamente.

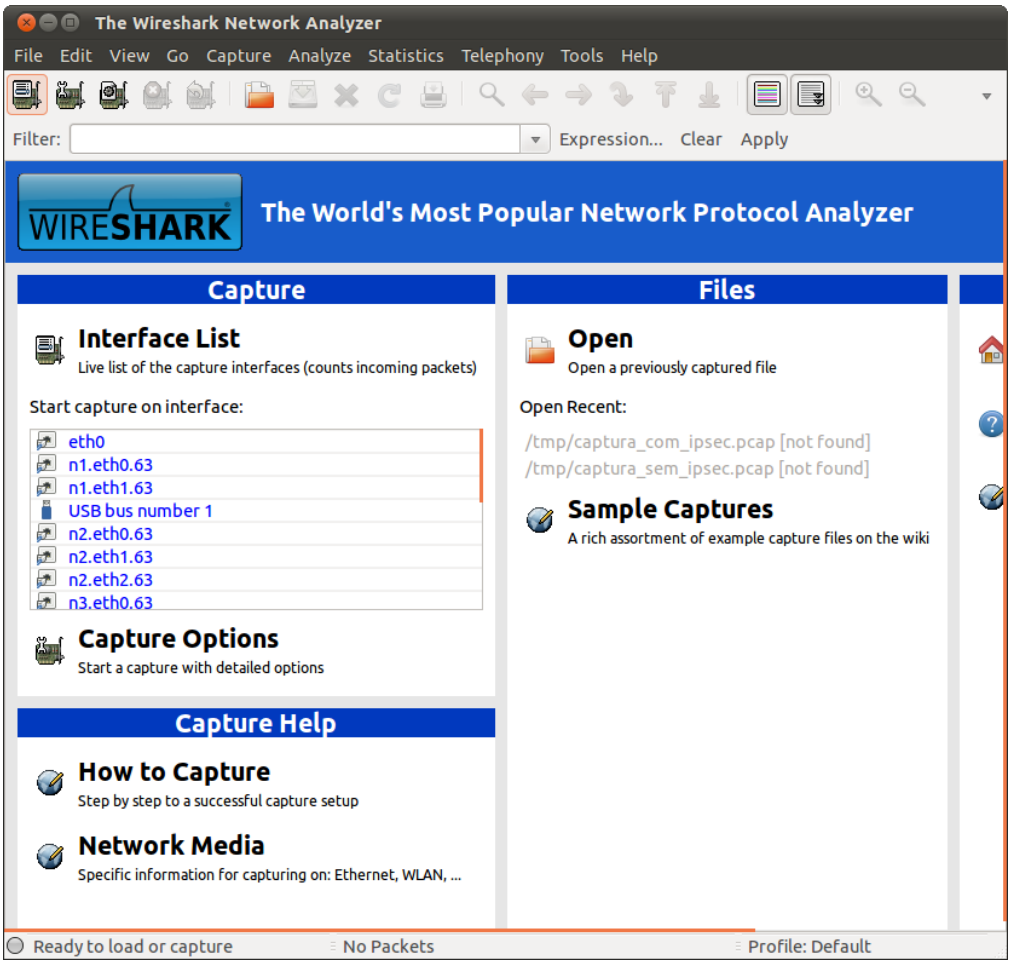
c. Abra um console no sistema operacional que está executando o Core e execute o Wireshark como “root”, a senha quando solicitada é “core”. Isto é necessário pois as interfaces da simulação do core somente são visíveis para “super users”.:

```
$ sudo wireshark
```

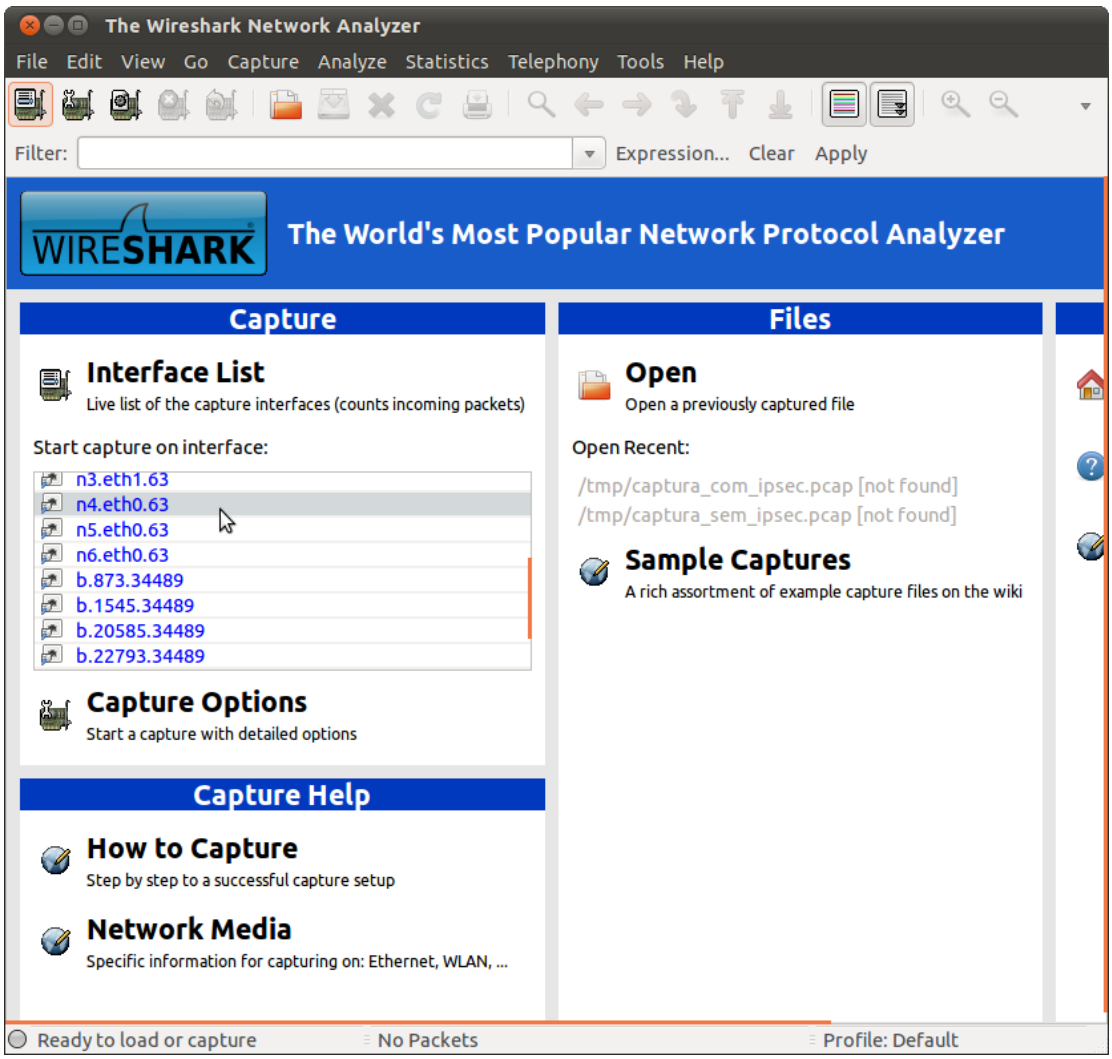
Você deve visualizar as seguintes janelas neste processo, clique em OK e ignore as janelas de aviso que aparecerem:






The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply
WIRESHARK
The World's Most Popular Network Protocol Analyzer
Capture
Interface List
Live list of the capture interfaces (counts incoming packets)
Start capture on interface:
eth0
n1.eth0.63
n1.eth1.63
USB bus number 1
n2.eth0.63
n2.eth1.63
n2.eth2.63
n3.eth0.63
Capture Options
Start a capture with detailed options
Capture Help
How to Capture
Step by step to a successful capture setup
Network Media
Specific information for capturing on: Ethernet, WLAN, ...
Files
Open
Open a previously captured file
Open Recent:
/tmp/captura_com_ipsec.pcap [not found]
/tmp/captura_sem_ipsec.pcap [not found]
Sample Captures
A rich assortment of example capture files on the wiki
Ready to load or capture
No Packets
Profile: Default

d. Abra a captura na interface n4.eth0 (cliente1) conforme a figura:

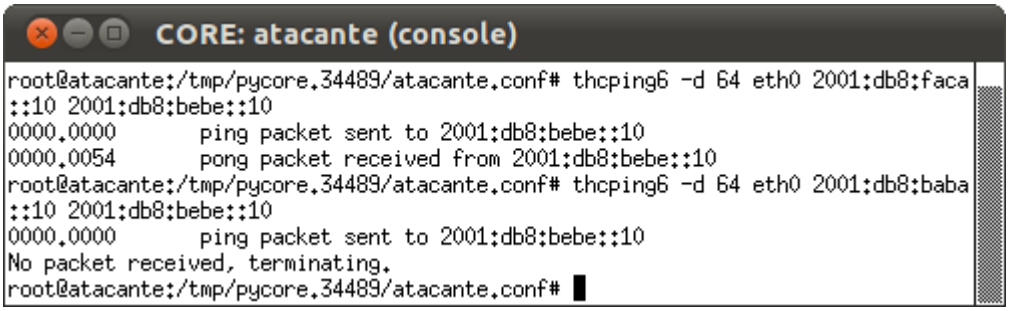


e. Deixe o Wireshark aberto capturando pacotes e abra o terminal de atacante, através do duplo-clique. Agora utilize a ferramenta thc-ping6 para mandar pacotes para o dispositivo 2001:db8:bebe::10, mande um pacote com o IP de origem correto (2001:db8:faca::10) e um segundo pacote informando como IP de origem o IP do cliente1 (2001:db8:baba::10):

```
$ ping6 -c 4 2001:db8:bebe::10
$ thcping6 -d 64 eth0 2001:db8:faca::10 2001:db8:bebe::10
$ thcping6 -d 64 eth0 2001:db8:baba::10 2001:db8:bebe::10
```

Obs: O primeiro ping é necessário para que o programa thcping6 funcione corretamente no CORE.

O resultado deve ser:



f. Volte ao Wireshark e procure por um pacote ICMP (echo) reply com origem 2001:db8:bebe::10 e destino 2001:db8:baba::10. Pode-se observar que o cliente1 recebeu um pacote ICMP (echo) reply sem fazer o ICMP (echo) request. Esta é a resposta ao ICMP request feito pelo dispositivo atacante com o IP forjado do cliente1. Este tipo de pacote forjado pode ser usado em diversos tipos de ataque, como man-in-the-middle, smurf, Denial-of-Service e outros.

g. Para resolver esta falha de segurança será implementado um túnel IPsec entre o roteador1 e o roteador2 fazendo com que todo tráfego entre as redes 2001:db8:baba:: e 2001:db8:bebe:: trafegue criptografado pela Internet e somente seja aceito na rede destino se possuir cabeçalho de autenticação IPsec válido.
Para configurar o túnel IPsec abra o terminal de roteador1 através do duplo-clique e verifique o conteúdo do arquivo ipsec.conf com o seguinte comando:

```
# cat ipsec.conf
```

O conteúdo do arquivo é o seguinte:

```
#!/usr/sbin/setkey -f
# Flush the SAD and SPD
flush;
spdflush;

# ESP SAs doing encryption using 192 bit long keys (168 + 24
# parity) and authentication using 128 bit long keys
add 2001:db8:beef::1 2001:db8:cafe::2 esp 0x201 -m tunnel -E 3des-cbc
0x7aeaca3f87d060a12f4a4487d5a5c3355920fae69a96c831 -A hmac-md5
0xc0291ff014dccdd03874d9e8e4cdf3e6;

add 2001:db8:cafe::2 2001:db8:beef::1 esp 0x301 -m tunnel -E 3des-cbc
0xf6ddb555acfd9d77b03ea3843f2653255afe8eb5573965df -A hmac-md5
0x96358c90783bbfa3d7b196ceabe0536b;
```

```
# Security policies
spdadd 2001:db8:baba::0/64 2001:db8:bebe::0/64 any -P out ipsec
           esp/tunnel/2001:db8:beef::1-2001:db8:cafe::2/require;

spdadd 2001:db8:bebe::0/64 2001:db8:baba::0/64 any -P in ipsec
           esp/tunnel/2001:db8:cafe::2-2001:db8:beef::1/require;
```

Note que as chaves deste exemplo não devem ser utilizadas em aplicações reais. Para gerar suas próprias chaves execute os comandos:

```
$ dd if=/dev/random count=16 bs=1| xxd -ps

$ dd if=/dev/random count=24 bs=1| xxd -ps
```

i. Recarregue as configurações do IPsec para que os dados inseridos no arquivo de configuração sejam executados:

```
$ setkey -f ipsec.conf
```

j. Para verificar se as chaves foram carregadas, execute os seguintes comandos:

```
$ setkey -D
$ setkey -DP
```

k. Abra o terminal do roteador2, através do duplo-clique.

l. Verifique o conteúdo do arquivo ipsec.conf com o seguinte comando:

```
#!/usr/sbin/setkey -f
# Flush the SAD and SPD
flush;
spdflush;

# ESP SAs doing encryption using 192 bit long keys (168 + 24
# parity) and authentication using 128 bit long keys
add 2001:db8:beef::1 2001:db8:cafe::2 esp 0x201 -m tunnel -E 3des-cbc
0x7aeaca3f87d060a12f4a4487d5a5c3355920fae69a96c831 -A hmac-md5
0xc0291ff014dccdd03874d9e8e4cdf3e6;

add 2001:db8:cafe::2 2001:db8:beef::1 esp 0x301 -m tunnel -E 3des-cbc
0xf6ddb555acfd9d77b03ea3843f2653255afe8eb5573965df -A hmac-md5
0x96358c90783bbfa3d7b196ceabe0536b;

# Security policies
```

```
spdadd 2001:db8:baba::0/64 2001:db8:bebe::0/64 any -P in ipsec
   esp/tunnel/2001:db8:beef::1-2001:db8:cafe::2/require;

spdadd 2001:db8:bebe::0/64 2001:db8:baba::0/64 any -P out ipsec
   esp/tunnel/2001:db8:cafe::2-2001:db8:beef::1/require;
```

m. Recarregue as configurações do IPsec para que os dados inseridos no arquivo de configuração sejam executados:

```
$ setkey -f ipsec.conf
```

n. Para verificar se as chaves foram carregadas, execute os seguintes comandos:

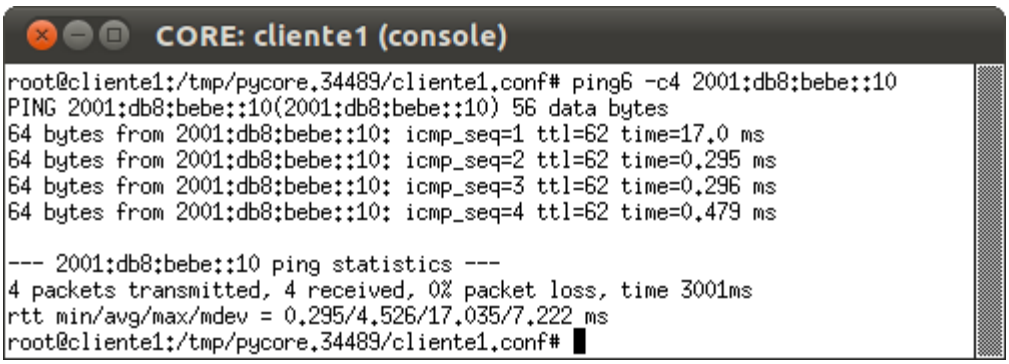
```
$ setkey -D
$ setkey -DP
```

o. Volte ao Wireshark e altere a interface analisada para a eth0 do roteador3 que está conectado ao roteador1 (n2.eth0).

p. Abra o console do cliente1 e envie um ping6 para o cliente2:

```
# ping6 -c 4 2001:db8:bebe::10
```

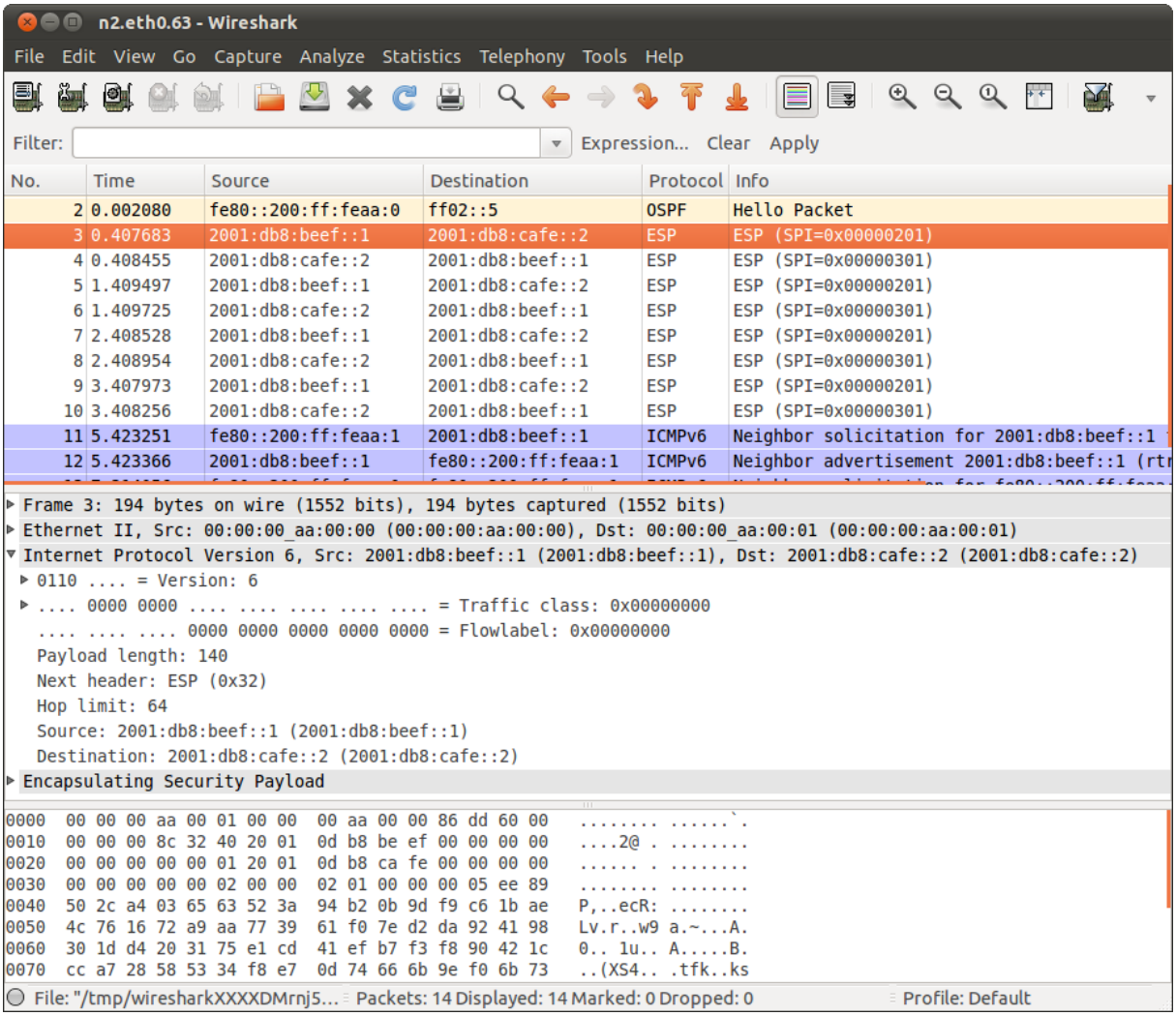
O resultado será um ping com funcionamento correto, conforme a figura:



q. No Wireshark procure pacotes do tipo ICMP (echo) request ou reply.

r. Note que não é possível encontrar este tipo de pacotes apesar de o ping ter funcionado corretamente, isto ocorre pois o túnel estabelecido entre o roteador1 e o roteador2 está encriptando os pacotes ICMP (echo) request e reply. Para confirmar isto procure por pacotes com protocolo ESP. É possível notar a existência de 4 pacotes do roteador1 para o roteador2 e 4 pacotes do roteador2 para o roteador1. Esta é exatamente a quantidade de

pacotes ping que foi gerada na conversa entre o cliente1 e o cliente2 e como o tráfego de rede nesta simulação é controlado, pode-se concluir que estes pacotes realmente são os pacotes ping. Analise um dos pacotes e é possível notar que os IPs de origem e destino são os IPs dos roteadores. A captura do Wireshark será similar a figura:



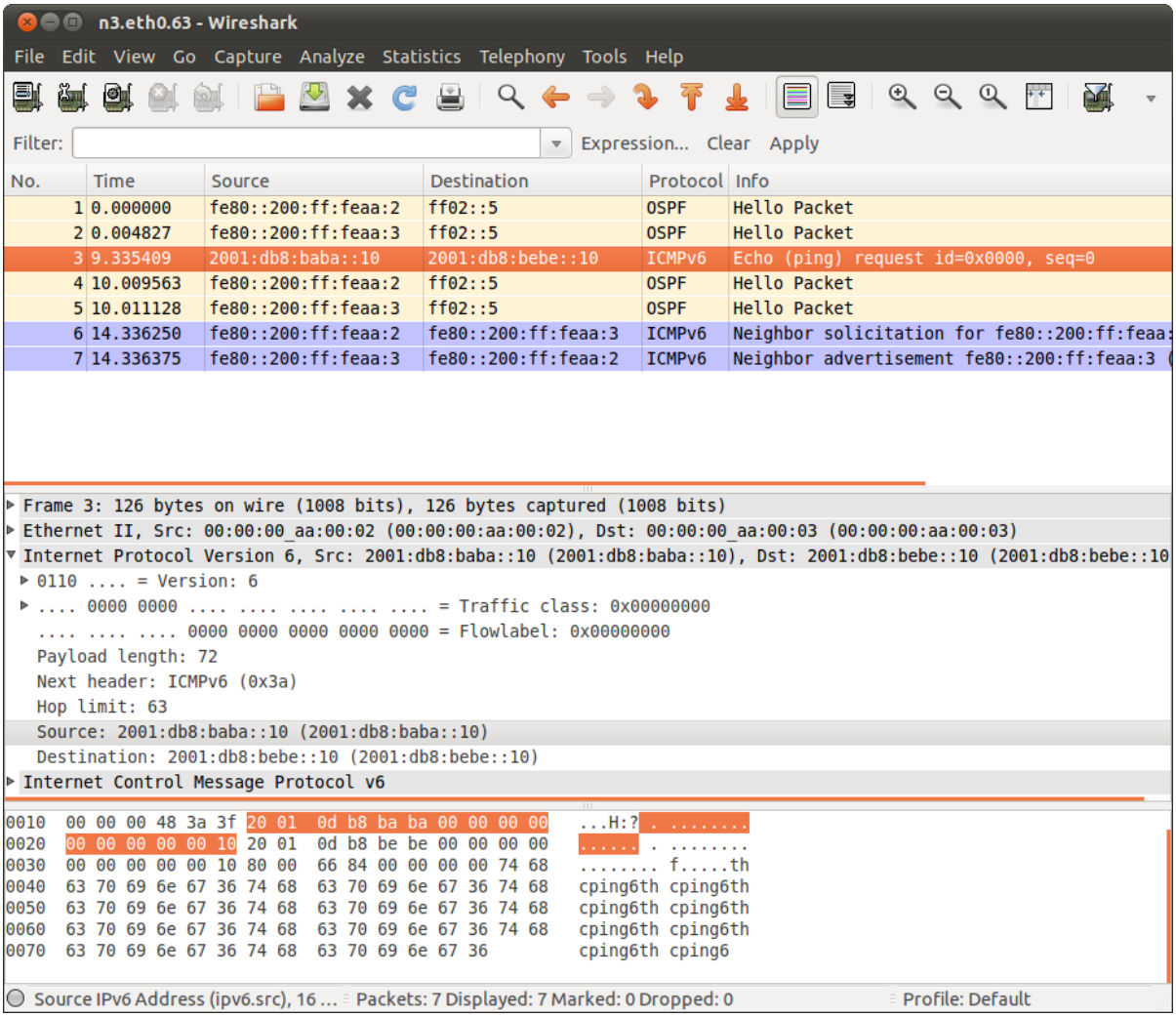
s. Configure o Wireshark para capturar pacotes recebidos no cliente1 (n4.eth0). Gere um novo pacote ping falsificado no console do atacante, na tentativa de que o pacote de resposta chegue ao cliente1 originado pelo cliente2. Para isto utilize o comando:

```
$ ping6 -c 4 2001:db8:bebe::10
$ thcping6 -d 64 eth0 2001:db8:baba::10 2001:db8:bebe::10
```

Obs: O primeiro ping é necessário para que o programa thcping6 funcione corretamente no CORE.

t. Procure no Wireshark um pacote ICMP (ping) reply originário do cliente2. Não é possível encontrar este pacote, pois ele não chegou ao cliente1.

u. Configure agora o Wireshark para analisar a eth0 do roteador2 (n3.eth0) e faça repita o envio do pacote forjado. Procure o pacote forjado. É possivel ver que ele é recebido pelo roteador2, mas um pacote de ICMP (ping) reply não passa por esta interface.



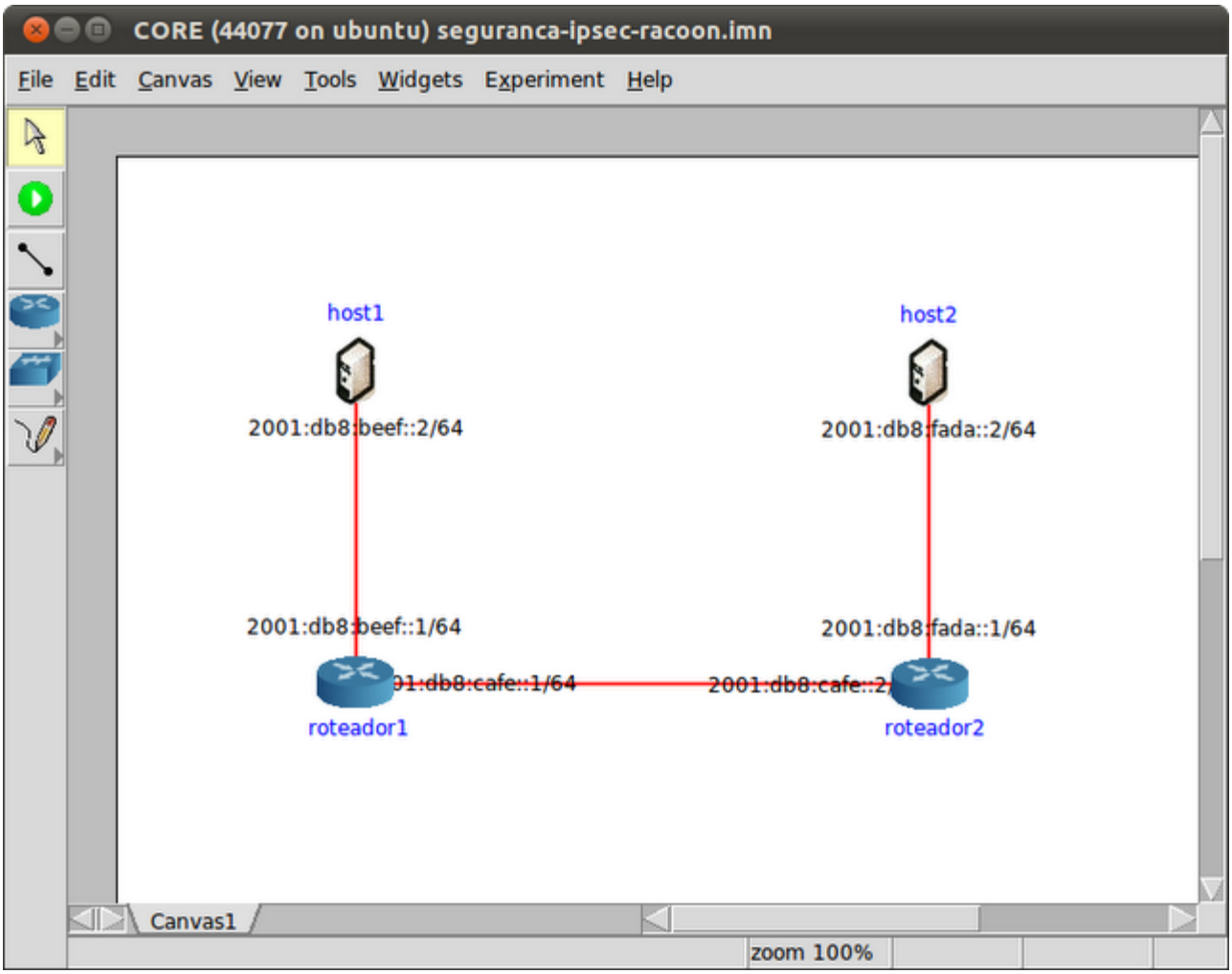
v. Configure agora o Wireshark para analisar a eth1 do roteador2 (n3.eth1) e repita o envio do pacote forjado. Note que o pacote ICMP (ping) request chega a interface eth0 do roteador2, mas não é redirecionado para a interface eth1 como acontecia quando o IPsec não estava configurado. Isto ocorre pois o roteador recebe um pacote vindo da rede 2001:db8:baba:: sem estar autenticado e criptografado. Neste caso o comportamento do roteador é descartar o pacote, impedindo o ataque que utiliza a falsificação do endereço de origem.

Experiência 5 - IPsec: Configuração automática através de chaves

1. Se você estiver utilizando a máquina virtual fornecida, vá para o passo 2. Caso se verifique que o IPSec ou o Racoon não estejam instalados na sua máquina virtual, execute o comando abaixo:

```
$ sudo apt-get install ipsec-tools racoon
```

2. No CORE, abra o arquivo **"seguranca-ipsec-racoon.imn"**. A seguinte topologia inicial de rede deve aparecer:



3. Inicie a simulação:

   a. aperte o botão  ;
   ou utilize o menu Experiment > Start.

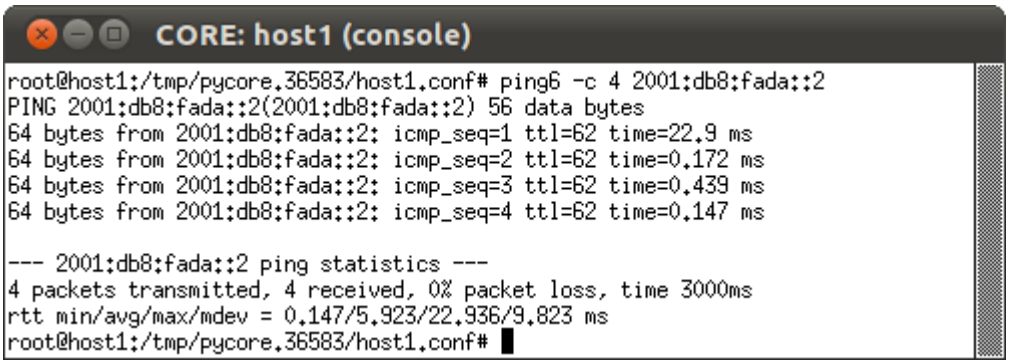
   b. Espere até que o CORE termine a inicialização da simulação e abra um

terminal do roteador1, um do host1 e um do host2, através do duplo-clique no símbolo de cada um dos três nós.

c. Embora o IPSec e o Racoon já tenham sido instalados na máquina virtual, nenhum dos dois está configurado quando a simulação é inicializada. Isso deverá ser feito individualmente para cada um dos nós da simulação no CORE. Devido à ausência de configuração inicial do IPSec entre os nós, a comunicação entre eles não está protegida e pode ocorrer livremente. Caso você queira verificar isto, vá para o terminal do host1 e execute um ping entre para o host2 com o comando abaixo:

```
$ ping6 -c 4 2001:db8:fada::2
```

A tela do terminal deverá exibir mensagens como as da imagem abaixo:

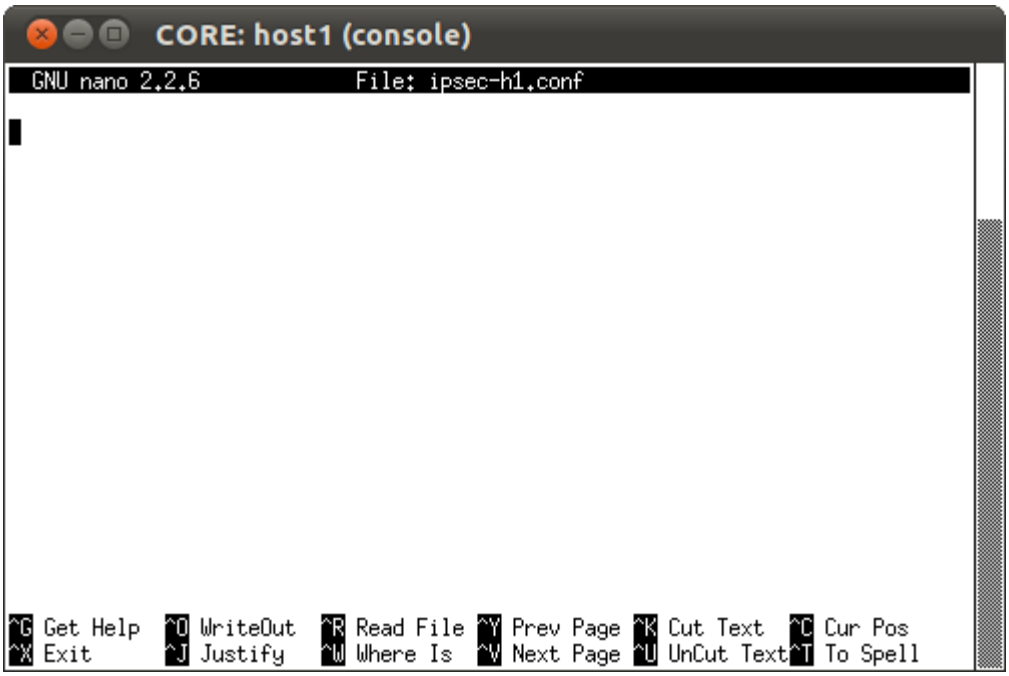


4. Escreva o arquivo de configuração do IPsec para o host1.
Conforme já mencionado, embora o Racoon faça a troca de chaves de autenticação e de criptografia automaticamente, é necessário definir primeiro as políticas de segurança manualmente através do comando `setkey`, da mesma maneira como foi feito na experiência 1. A seguir, repetiremos estes passos de escrita do arquivo de configuração, porém de maneira mais suscinta. Qualquer dúvida relacionada aos comandos que serão apresentados nestes próximos sub-itens pode ser solucionada revisando a experiência 1, que explica em detalhes como a configuração manual do IPsec é feita.

a. Vá para o terminal do host1. Crie um arquivo ".conf" com o comando abaixo:

```
$ nano ipsec-h1.conf
```

A seguinte tela irá surgir:



b. Digite as linhas que executarão os comandos de reinicialização da configuração do IPsec:

```
#!/usr/sbin/setkey -f
flush;
spdflush;
```

c. A seguir insira as linhas que definem as políticas de segurança desta máquina.
Lembre-se que uma política possui o seguinte formato simplificado:

```
spdadd [ips_origem] [ips_destino] [protocolo_camada_superior]
[política];
```

A configuração automática através do Racoon funciona com o IPsec tanto em modo de transporte quanto em modo túnel. Nesta experiência, configuraremos o IPsec entre o host1 e o host2 em modo de transporte.

Os parâmetros usados para a primeira política serão:

[ips_origem]: 2001:db8:beef::2
[ips_destino]: 2001:db8:fada::2
[protocolo_camada_superior]: any
[política]: -P in ipsec
esp/transport//require
ah/transport//require

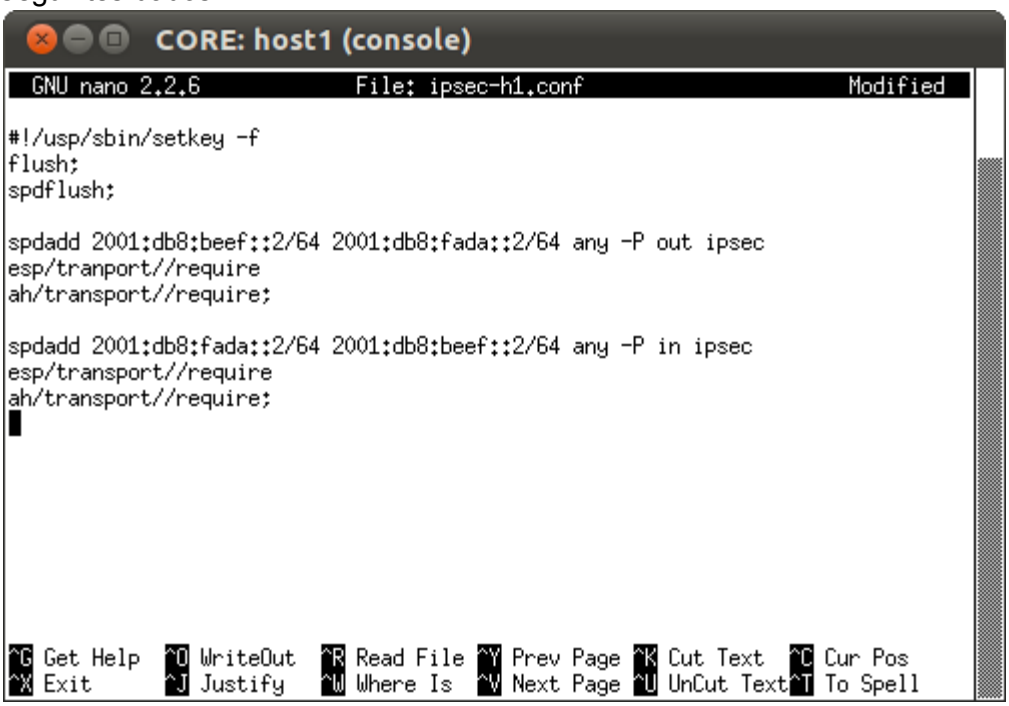
Portanto, o comando assumirá a seguinte forma:

```
spdadd 2001:db8:beef::2/64 2001:db8:fada::2/64 any -P out ipsec
esp/transport//require
ah/transport//require;
```

A linha de comando para gerar a segunda política, no sentido oposto de comunicação, será praticamente igual à linha anterior, porém com as seguintes diferenças: 1º) os IPs serão trocados de posição (o IP "fada::2" será a origem e o IP "beef::2" será o destino) e 2º) o campo "política" terá o sentido `in` no lugar de `out`. O comando assumirá a seguinte forma:

```
spdadd 2001:db8:fada::2/64 2001:db8:beef::2/64 any -P in ipsec
esp/transport//require
ah/transport//require;
```

No `nano`, escreva as duas linhas acima após a linha que contém o `spdflush`. Após escrever todas estas linhas, a tela do `nano` deverá apresentar os seguintes dados:



d. Salve o arquivo e saia do `nano`, pressionando primeiro CTRL + O, depois ENTER e em seguida CTRL + X.

5. Escreva o arquivo de configuração do IPsec para o host2.

a. Vá para o terminal do host2. Crie um arquivo ".conf" com o comando abaixo:

```
$ nano ipsec-h2.conf
```

b. Digite as linhas que executarão os comandos de reinicialização da configuração do IPsec:

```
#!/usr/sbin/setkey -f
flush;
spdflush;
```

c. A seguir insira as linhas que definem as políticas de segurança desta máquina.
Os parâmetros usados para a primeira política serão:

[ips_origem]: 2001:db8:fada::2
[ips_destino]: 2001:db8:beef::2
[protocolo_camada_superior]: any
[política]: -P in ipsec
esp/transport//require
ah/transport//require

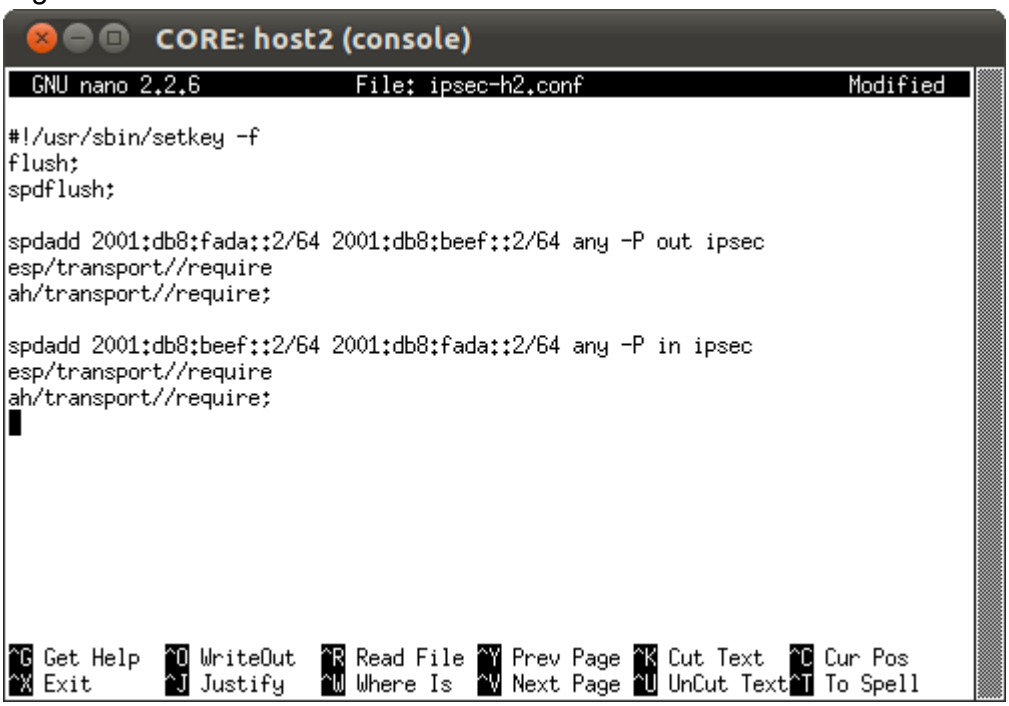
Portanto, o comando assumirá a seguinte forma:

```
spdadd 2001:db8:fada::2/64 2001:db8:beef::2/64 any -P out ipsec
esp/transport//require
ah/transport//require;
```

A linha de comando para gerar a segunda política, no sentido oposto de comunicação, será praticamente igual à linha anterior, porém com as seguintes diferenças: 1º) os IPs serão trocados de posição (o IP "beef::2" será a origem e o IP "fada::2" será o destino) e 2º) o campo "política" terá o sentido "out" no lugar de "in". O comando assumirá a seguinte forma:

```
spdadd 2001:db8:beef::2/64 2001:db8:fada::2/64 any -P in ipsec
esp/transport//require
ah/transport//require;
```

No `nano`, escreva as duas linhas acima após a linha que contém o `spdflush`. Após escrever todas estas linhas, a tela do `nano` deverá apresentar os seguintes dados:



d. Salve o arquivo e saia do `nano`, pressionando primeiro CTRL + O, depois ENTER e em seguida CTRL + X.

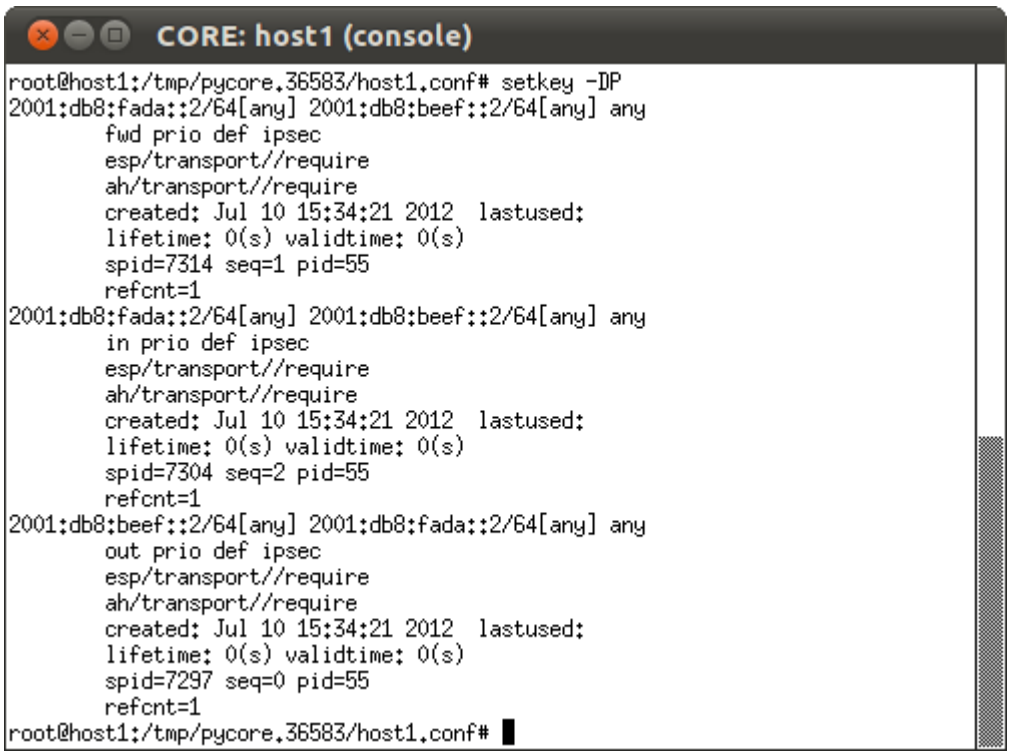
6. Vá para o terminal do host1 e carregue as configurações do arquivo "ipsec-h1.conf" com o comando abaixo:

```
$ setkey -f ipsec-h1.conf
```

Para verificar se as políticas foram carregadas, execute o seguinte comando:

```
$ setkey -DP
```

A tela abaixo deverá ser exibida:



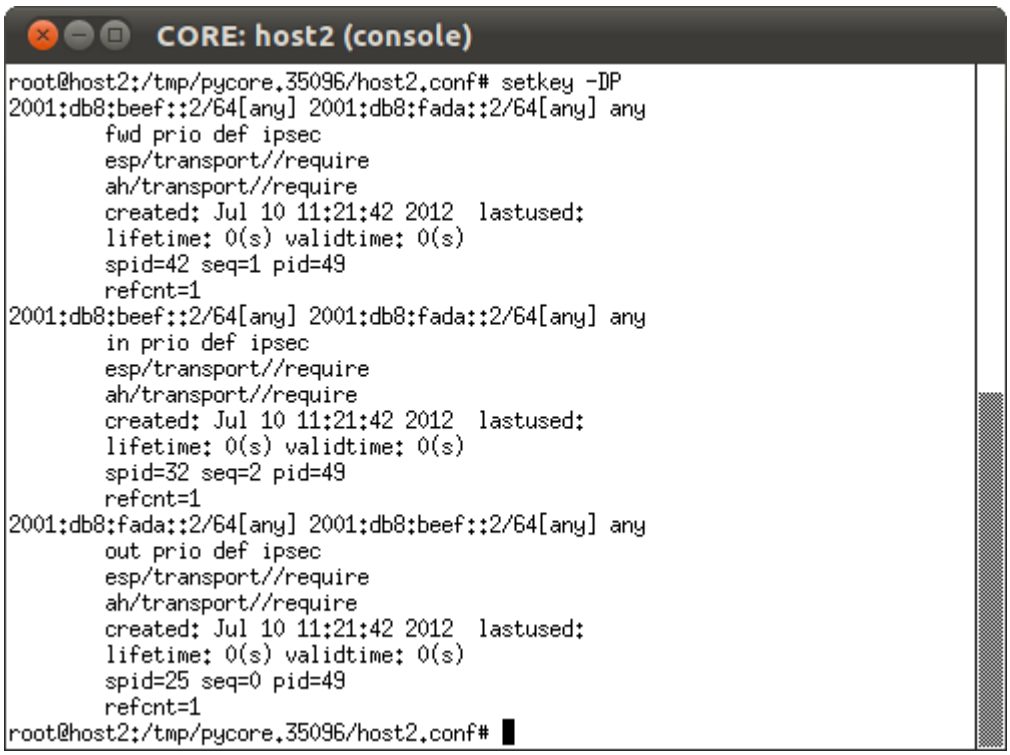
7. Vá para o terminal do host2 e carregue as configurações do arquivo ".conf". Supondo que você tenha nomeado este arquivo como "ipsec-h2.conf", o comando ficará igual ao abaixo:

```
$ setkey -f ipsec-h2.conf
```

Para verificar se as políticas foram carregadas, execute o seguinte comando:

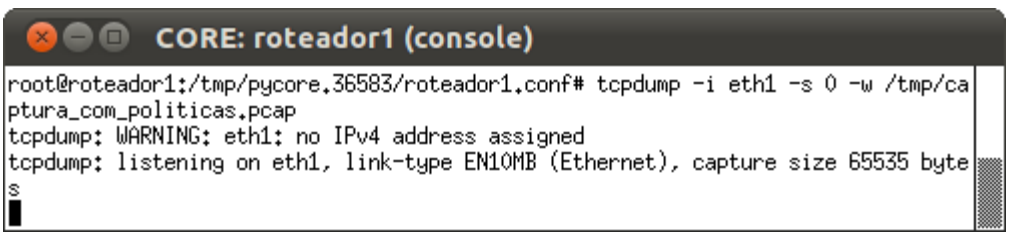
```
$ setkey -DP
```

A tela abaixo deverá ser exibida:



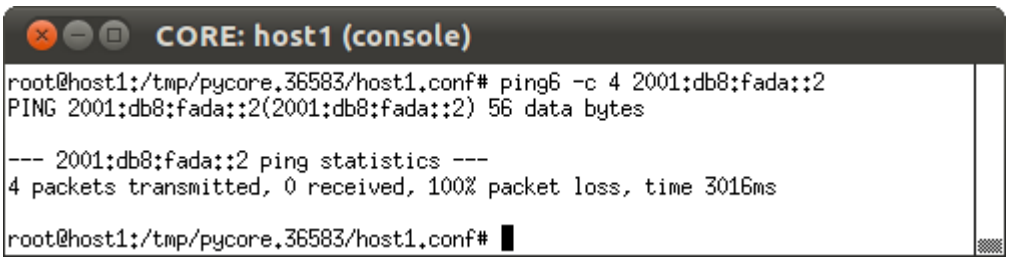
8. Envie pacotes entre os dois roteadores para análise do cabeçalho.
   Com as políticas de segurança do IPsec configuradas, repita o procedimento de captura de pacotes.
   a. No terminal do roteador1:

```
$ tcpdump -i eth1 -s 0 -w /tmp/captura_com_politicas.pcap
```



   b. No terminal do host1:

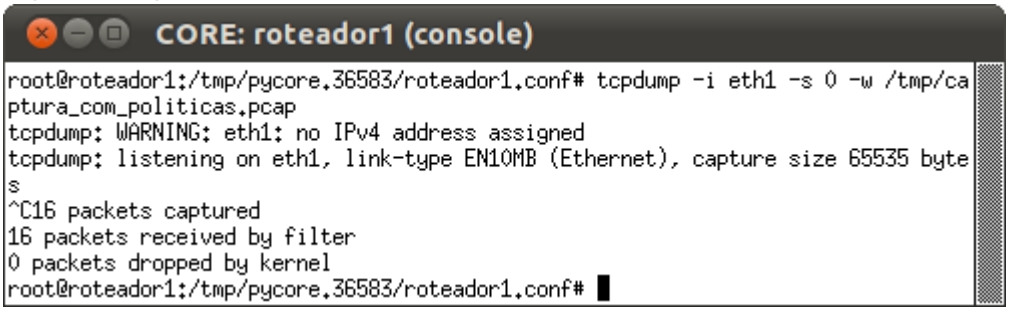
```
$ ping6 -c 4 2001:db8:fada::2
```



   Perceba que o resultado indica "4 packets transmitted" (4 pacotes enviados)

e “0 received” (0 recebidos), o que significa que não houve resposta para os quatro pings enviados. Isso está de acordo com a configuração do IPsec, pois as políticas de segurança já estão implementadas no host1 mas as chaves não foram criadas, o que impede a comunicação normal entre o par de máquinas host1 e host2.

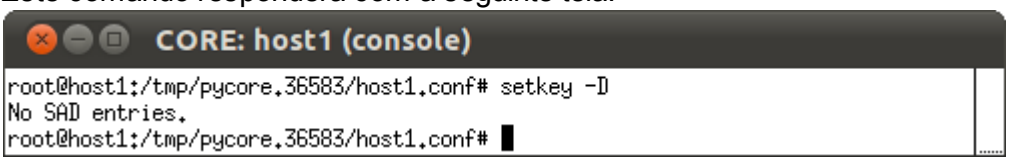
c. Espere alguns segundos, vá para o terminal do roteador1 e encerre a captura de pacotes pressionando Ctrl+C.
O resultado deve ser similar ao abaixo, já que o número de pacotes capturados pode variar:



Mais tarde verificaremos que os pacotes nunca saíram do host1, pois ele está configurado para enviar (ou receber) os pacotes para (do) host2 somente se possuir as chaves de autenticação e criptografia do host2. Porém, conforme já mencionado, nem o host1 nem o host2 possuem SA’s com chaves de segurança. Caso você queira verificar a ausência de SA’s, digite o seguinte comando no terminal do host1:

```
$ setkey -D
```

Este comando responderá com a seguinte tela:



9. Escreva o arquivo de configuração do Racoon para o host1:
Assim como o `setkey`, o Racoon também possui um arquivo de configuração que é carregado automaticamente na inicialização do sistema. Assim, caso desejássemos fazer uma configuração permanente na máquina, seria necessário alterar o arquivo “/etc/racoon/racoon.conf”, colocando nele as configurações necessárias. Para executá-las, usaria-se então o comando “racoon -f <nome_do_arquivo.conf>” ou reinicializaria-se o Racoon através do comando abaixo:

```
$ /etc/init.d/racoon restart
```

Porém, como nós faremos apenas uma configuração de teste, criaremos um novo arquivo ".conf", o qual precisará ser carregado através do comando "racoon". Apresentamos abaixo a estrutura mais simples possível que este arquivo deve possuir e a seguir descreveremos os parâmetros relacionados na estrutura:

```
path pre_shared_key ["xyz"];

remote [xyz] {
        exchange_mode [xyz];
        proposal {
                encryption_algorithm [xyz];
                hash_algorithm [xyz];
                authentication_method [xyz];
                dh_group [xyz];
        }
}

sainfo [xyz] {
        pfs_group [xyz];
        lifetime time [xyz];
        encryption_algorithm [xyz];
        authentication_algorithm [xyz];
        compression_algorithm [xyz];
}
```

A seguir, escreveremos nosso arquivo ".conf". Cada parâmetro e quais valores se aplicam para a nossa experiência serão descritos a seguir:

a. Vá para o terminal do host1. Crie um arquivo ".conf" com o comando abaixo:

```
$ nano racoon-h1.conf
```

b. Primeiramente, digite a seguinte linha:

```
path pre_shared_key "/etc/racoon/psk1.txt";
```

Esta linha define o diretório e o nome do arquivo que contém as chaves pré-compartilhadas.

c. Pule duas linhas e inclua o seguinte campo:

```
remote anonymous {

}
```

Dentro das chaves escreveremos os parâmetros que o Racoon usará na primeira fase da criação de SA's, que é o estabelecimento de uma conexão

segura para uso particular do Racoon, onde ele então trocará as chaves do IPsec com seu par na rede. É possível definir como é feita essa troca de chaves individualmente para cada par que a máquina possui na rede, ou para um grupo limitado de máquinas dentro de uma faixa de IPs; para fazer isto, trocados a palavra “anonymous” pelo endereço IP do par (ou por uma faixa de IPs). Porém, se deixarmos o parâmetro “remote” como “anonymous”, as configurações se aplicarão ao processo de troca de chaves de todos os pares de máquinas que estiverem definidos nas políticas de segurança.

d. Dentro do campo “remote”, escreva as seguintes linhas:

```
exchange_mode main;
proposal {

}
```

A primeira linha define a forma pela qual estabelece-se a conexão segura da fase 1 do processo de troca de chaves. As opções são “main”, “aggressive” e “base”. As duas últimas formas são alternativas mais rápidas e menos seguras de estabelecimento da conexão; por isso, devem ser usadas somente em situações especiais não descritas neste curso. A segunda linha inicia um segundo campo aninhado dentro do primeiro. Este campo (“proposal”) define detalhes técnicos da fase 1 como o algoritmo de criptografia que é usado nesta fase (não confundir com o algoritmo de criptografia usado pelo IPSec, que não será necessariamente o mesmo) e o método de autenticação.

e. Dentro do campo “proposal”, escreva as seguintes linhas:

```
encryption_algorithm 3des;
hash_algorithm sha1;
authentication_method pre_shared_key;
dh_group modp1024;
```

Dentre os quatro parâmetros indicados, o terceiro possui um significado especial e próprio do Racoon: define o método de autenticação da fase 1, que no nosso exemplo será “pre_shared_key”, ou seja, o método no qual os pares de dispositivos possuem de antemão uma senha comum, “pré-compartilhada”, já armazenada em disco em ambos os pares. Um outro método importante é a autenticação por certificado X.509 (parâmetro “rsasig”); este método centraliza em uma outra máquina a função de autenticação dos pares. Esta máquina é chamada Autoridade Certificadora.
As outras linhas definem parâmetros de criptografia e funções similares, executadas somente nesta fase 1 da troca de chaves. A primeira linha define o algoritmo de criptografia; neste exemplo, usaremos o algoritmo “3des”. Já a segunda linha define o algoritmo de hash; usaremos o “sha1”. Finalmente, a quarta linha define o grupo usado para o algoritmo de Diffie-Hellman;

usaremos o modp1024.
Para mais opções de parâmetros, acesse o manual (“man page”) do arquivo “racoon.conf” (`man racoon`). O texto escrito no arquivo até agora deve ser igual ao apresentado abaixo:

```
path pre_shared_key "/etc/racoon/psk1.txt";

remote anonymous {
  exchange_mode main;
  proposal {
    encryption_algorithm 3des;
    hash_algorithm sha1;
    authentication_method pre_shared_key;
    dh_group modp1024;
  }
}
```

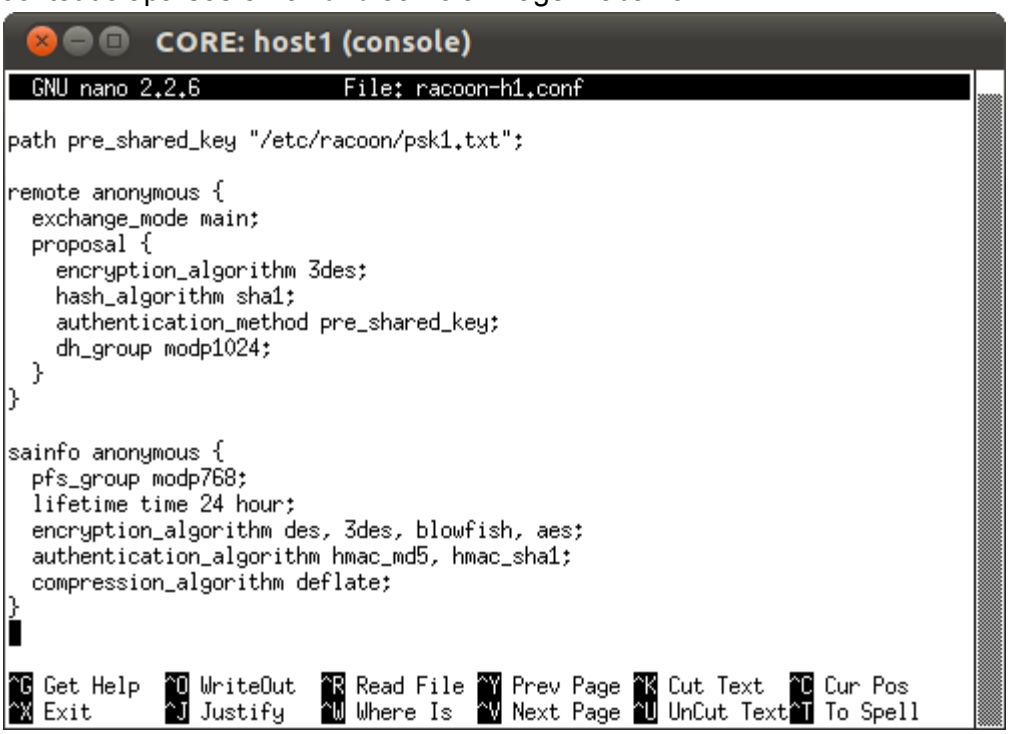
f. Depois da chave de encerramento do campo “remote”, insira as seguintes linhas:

```
sainfo anonymous {
  pfs_group modp768;
  lifetime time 24 hour;
  encryption_algorithm des, 3des, blowfish, aes;
  authentication_algorithm hmac_md5, hmac_sha1;
  compression_algorithm deflate;
}
```

Este campo define os parâmetros da conexão segura final que será estabelecida entre esta máquina e os seus pares indicados após a palavra “sainfo”. Assim como no campo “remote”, se definirmos um IP (ou faixa de IPs) no lugar de “anonymous”, as configurações serão aplicadas somente na SA entre esta máquina e o par com este IP (ou faixa de IPs).
A primeira linha define o grupo usado para o algoritmo de Diffie-Hellman. Note que não é necessário usar o mesmo grupo no parâmetro “dh_group”, que está dentro do campo “proposal”; aqui usaremos o modp768.
A segunda linha define o tempo de validade das chaves do IPsec que serão geradas pelo Racoon. Após o período determinado neste parâmetro (neste exemplo, 24 horas), o Racoon trocará automaticamente todas as chaves dos pares definidos no parâmetro sainfo (caso o parâmetro seja preenchido com “anonymous”, as chaves de todos os pares desta máquina serão trocadas periodicamente). Esta é uma das funções mais úteis do Racoon, pois uma vez que ele seja corretamente configurado e inicializado, o administrador da rede não precisará mais cuidar da renovação das chaves do IPsec periodicamente, o que é um procedimento de segurança muito importante a ser implementado. Este parâmetro “lifetime” pode ser retirado do arquivo

".conf" se o usuário desejar, o que fará com que as chaves implementadas não sejam renovadas automativamente e, uma vez criadas, sejam usadas indefinidamente. Contudo, recomenda-se fortemente usar tal parâmetro.
A terceira e a quarta linha do exemplo acima listam o algoritmo de criptografia e o de autenticação que podem ser usados por uma SA nesta máquina. ATENÇÃO! Os algoritmos que serão efetivamente utilizados não são definidos aqui, mas sim nas políticas de segurança, que foram escritas no arquivo de configuração do IPsec. Aqui, os algoritmos listados são apenas a lista de algoritmos que o Racoon reconhecerá. Por isso, recomenda-se listar todos os que a máquina é capaz de usar (porém, observe que o kernel do Linux pode não reconhecer alguns algoritmos dentre todos os disponíveis para estas funções).
Para a quinta linha, por enquanto só há disponível a opção "deflate", a qual será usada.

Após digitar todas estas informações no arquivo "racoon-h1.conf", seu conteúdo aparecerá no `nano` como a imagem abaixo:



10. Escreva o arquivo de configuração do Racoon para o host2.
Este arquivo será igual em quase tudo ao arquivo de configuração do host1. O único item que mudará será o arquivo com as chaves pré-compartilhadas do host2, que chamaremos de "psk2.txt". Assim:

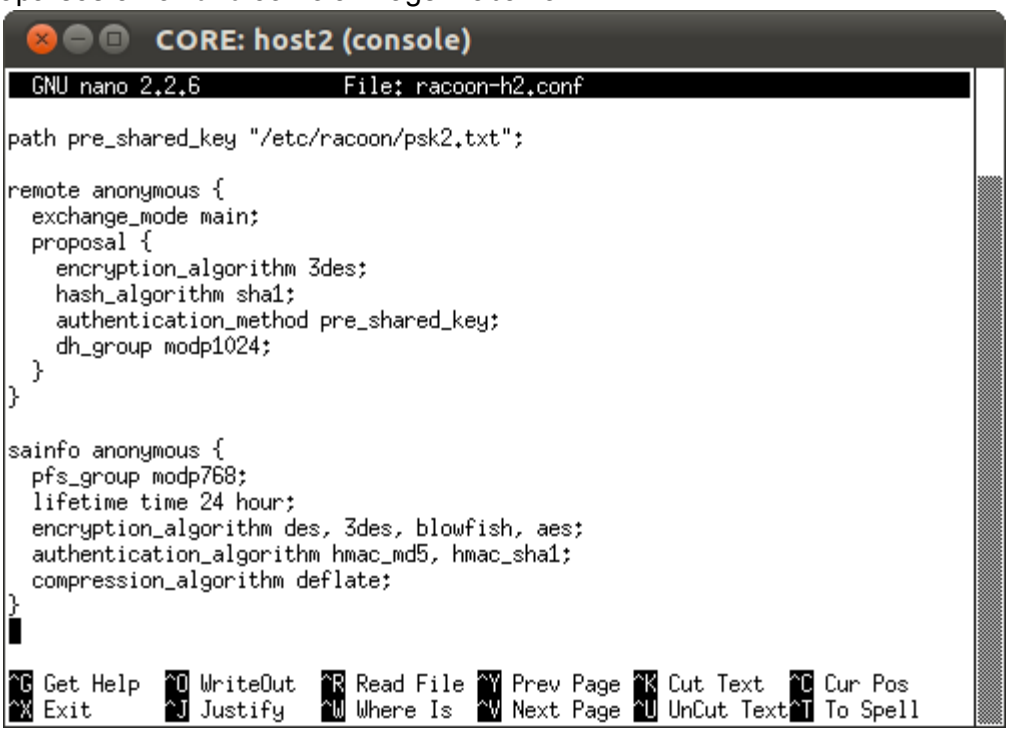
a. Vá para o terminal do host2. Crie um arquivo ".conf" com o comando abaixo:

```
$ nano racoon-h2.conf
```

b. Digite as linhas abaixo. O conteúdo deve ser exatamente igual.

```
path pre_shared_key "/etc/racoon/psk2.txt";

remote anonymous {
  exchange_mode main;
  proposal {
    encryption_algorithm 3des;
    hash_algorithm sha1;
    authentication_method pre_shared_key;
    dh_group modp1024;
  }
}

sainfo anonymous {
  pfs_group modp768;
  lifetime time 24 hour;
  encryption_algorithm des, 3des, blowfish, aes;
  authentication_algorithm hmac_md5, hmac_sha1;
  compression_algorithm deflate;
}
```

Observando o conteúdo acima, note que a única diferença (em negrito) entre este arquivo e o arquivo "racoon-h1.conf" é o parâmetro "psk2.txt", que é o nome do arquivo onde estão as chaves pré-compartilhadas. Após digitar todas estas informações no arquivo "racoon-h2.conf", seu conteúdo aparecerá no `nano` como a imagem abaixo:



11. Escreva o arquivo com as chaves pré-compartilhadas para o host1.

Este arquivo deve possuir uma linha para cada par com quem a máquina irá estabelecer uma conexão IPsec. Esta linha deverá possuir duas colunas separadas pelo primeiro espaço que aparecer na linha, conforme o formato indicado abaixo:

```
[IP_do_outro_par] [senha]
```

A primeira coluna de cada linha possuirá o IP do par com quem a máquina ir
á estabelecer uma conexão IPsec. Após o IP, deve-se colocar um espaço, que indicará que tudo o que vier em seguida será a senha pré-compartilhada. Esta senha será o conteúdo da segunda coluna. A senha poderá incluir qualquer caractere, inclusive outros espaços. Observe que esta senha deve ser a mesma para os dois pares de máquinas, isto é, no arquivo de chaves pré-compartilhadas de cada par deverá existir uma linha com o IP do outro par, um espaço, e em seguida a mesma senha para as duas máquinas.

ATENÇÃO! Para o Racoon funcionar corretamente, este arquivo precisa possuir uma permissão de acesso e leitura muito restrita: somente o dono do arquivo deve ter permissões para ler, alterar ou executar este arquivo. Isto se deve à importância do mesmo, que pede medidas de proteção contra acessos externos, uma vez que, caso um agente externo malicioso ganhe acesso a estas chaves pré-compartilhadas, toda a segurança IPsec na rede estará comprometida e poderá ser contornada. As instruções para alterar as permissões do arquivo serão dadas adiante.

b. Vá para o terminal do host1. Crie o arquivo em questão com o seguinte comando:

```
$ nano /etc/racoon/psk1.txt
```

ATENÇÃO! O arquivo foi criado dentro da pasta “/etc/racoon”, ao contrário dos arquivos anteriores que foram todos criados dentro da pasta do nó da simulação (algo igual ou similar a “/tmp/pycore.33649/” na sua máquina virtual). Isso foi feito porque as permissões de criação e alteração de arquivos da pasta “/etc” e suas subpastas são concedidas apenas ao superusuário do sistema (root). Criar o arquivo “psk1.txt” neste local é uma primeira medida de segurança, a fim de evitar que usuários externos do computador consigam manipular o arquivo com as chaves pré-compartilhadas. Note que os arquivos “.conf” para a configuração permanente do IPsec (“ipsec-tools.conf”) e do Racoon (“racoon.conf”) já estão dentro da pasta “/etc”. Nós criamos os arquivos “.conf” destas experiências fora da pasta “/etc” apenas porque estas configurações serão temporárias.

c. Na janela aberta pelo `nano`, insira a seguinte linha:

```
2001:db8:fada::2 senhadeteste
```

d. Salve o arquivo e saia do `nano`, pressionando primeiro CTRL + O, depois ENTER e em seguida CTRL + X.

e. Execute o comando para verificar se o arquivo foi salvo corretamente:

```
$ cat /etc/racoon/psk1.txt
```

O resultado deve ser:

f. Execute o seguinte comando para alterar as permissões do arquivo:

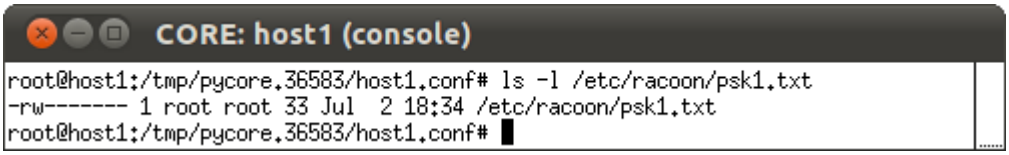
```
$ chmod 600 /etc/racoon/psk1.txt
```

Este comando proibe a alteração, leitura e execução do arquivo para qualquer um que não for o superuser, permitindo que o mesmo altere e leia o arquivo. OBS: caso mais alguém possua permissões com este arquivo, o Racoon não aceitará o mesmo como fonte de chaves pré-compartilhadas e não finalizará a fase 1 do processo de troca de chaves, impedindo que o IPsec seja corretamente configurado nas máquinas!

g. Verifique se o arquivo está com as permissões corretas, digitando o comando abaixo:

```
$ ls -l /etc/racoon/psk1.txt
```

A tela abaixo deverá ser exibida:



Observe o começo da segunda linha (que é a resposta do comando “ls”). Se a linha não começar com os caracteres “-rx-------” nesta exata forma, o comando chmod acima não foi executado ou foi executado incorretamente (ou com um parâmetro diferente de 600 ou com um arquivo diferente de “psk1.txt”).

12. Escreva o arquivo com as chaves pré-compartilhadas para o host2.

b. Vá para o terminal do host2. Crie o arquivo em questão com o seguinte comando:

```
$ nano /etc/racoon/psk2.txt
```

c. Na janela aberta pelo `nano`, insira a seguinte linha:

```
2001:db8:beef::2 senhadeteste
```

ATENÇÃO: a senha no arquivo psk2.txt DEVE SER A MESMA que está no arquivo psk1.txt do host1. A diferença está no IP indicado no arquivo (deve ser o IP do par da máquina em questão, portanto o par indicado no arquivo psk1.txt do host1 será `2001:db8:fada::2`, enquanto que o par indicado no arquivo psk1.txt do host2 será `2001:db8:beef::2`).

d. Salve o arquivo e saia do `nano`, pressionando primeiro CTRL + O, depois ENTER e em seguida CTRL + X.

e. Execute o comando para verificar se o arquivo foi salvo corretamente:

```
$ cat /etc/racoon/psk2.txt
```

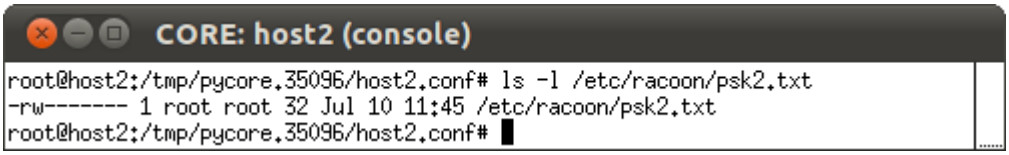
O resultado deve ser:

f. Execute o seguinte comando para alterar as permissões do arquivo:

```
$ chmod 600 /etc/racoon/psk2.txt
```

g. Verifique se o arquivo está com as permissões corretas, digitando o comando abaixo:

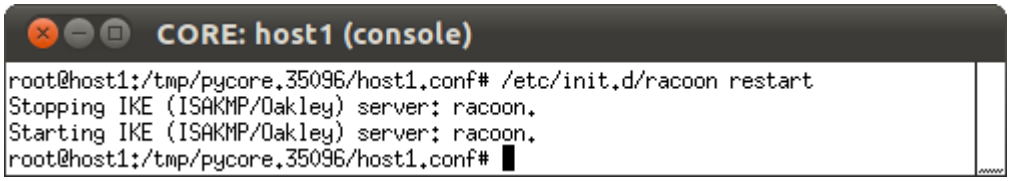
```
$ ls -l /etc/racoon/psk2.txt
```



13. Inicialize o Racoon no host1. Vá para o terminal do host1 e digite:

```
$ /etc/init.d/racoon restart
```

A resposta abaixo deverá ser exibida se o racoon foi inicializado corretamente:



14. Carregue as configurações a partir do arquivo "racoon-h1.conf" que acabamos de escrever:

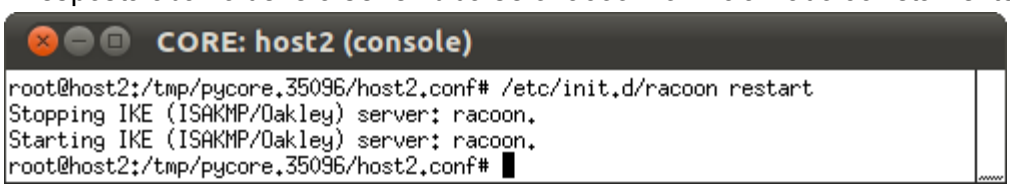
```
$ racoon -f racoon-h1.conf
```

Observe que este comando só irá funcionar se as políticas de segurança do IPSec já tiverem sido manualmente inseridas na máquina, o que foi feito no passo 6 . Caso o comando tenha sido executado corretamente, não surgirá nenhuma mensagem após escrever o comando e digitar ENTER.

15. Inicialize o Racoon no host2. Vá para o terminal do host2 e digite:

```
$ /etc/init.d/racoon restart
```

A resposta abaixo deverá ser exibida se o racoon foi inicializado corretamente:



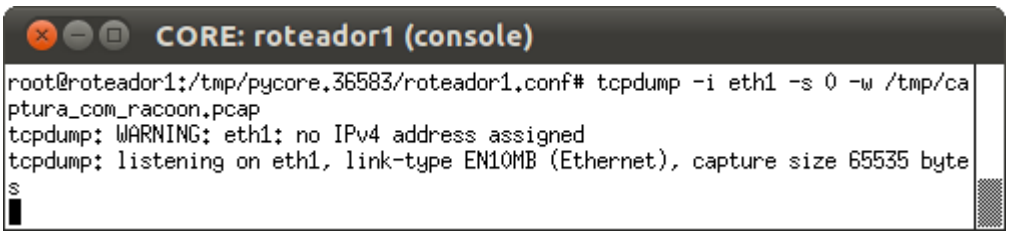
16. Carregue as configurações a partir do arquivo "racoon-h2.conf" que acabamos de escrever:

```
$ racoon -f racoon-h2.conf
```

17. Envie pacotes entre os dois roteadores para análise do cabeçalho.
Com as configurações do Racoon em execução, repita o procedimento de captura de pacotes.

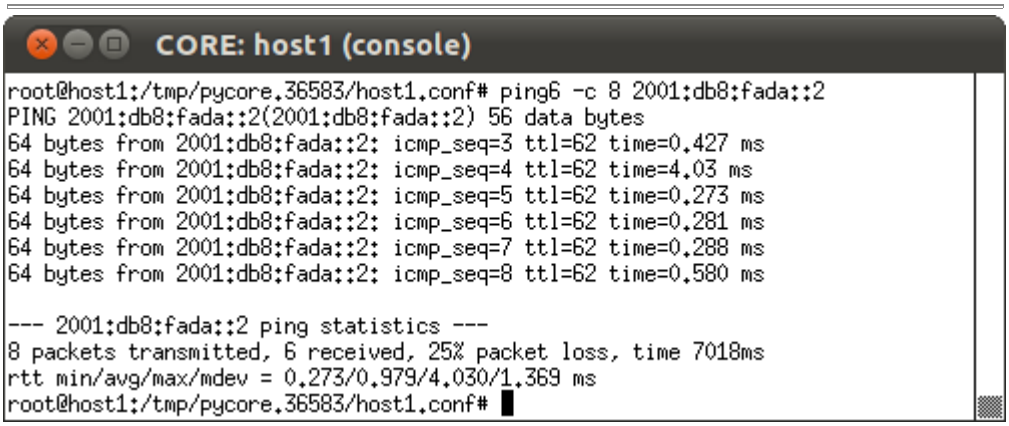
a. No terminal do roteador1:

```
$ tcpdump -i eth1 -s 0 -w /tmp/captura_com_racoon.pcap
```



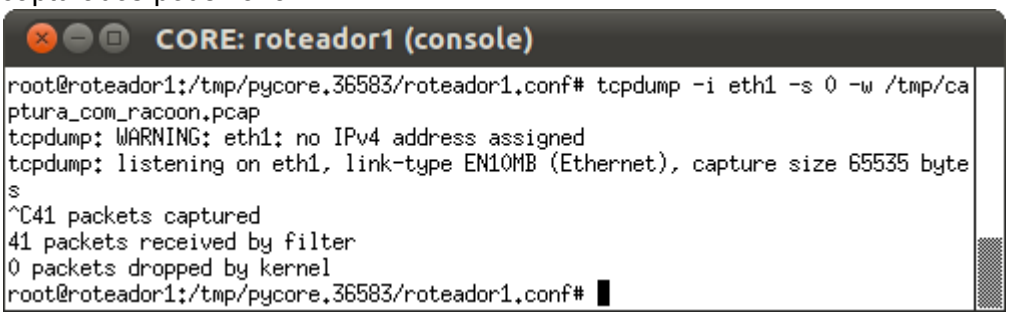
b. No terminal do host1:

```
$ ping6 -c 8 2001:db8:fada::2
```



Desta vez, enviamos 8 pings em vez do número usal de 4 pings para que você perceba bem o fato de que os primeiros pacotes de ping foram perdidos. Isso acontece porque o processo de criação de chaves do IPsec através do Racoon é inicializado somente quando ocorre uma tentativa de comunicação. Como o processo leva um certo tempo (veremos mais tarde, analisando a captura de pacotes, que várias mensagens são trocadas pelo par de máquinas), os primeiros pacotes de ping são enviados sem que as chaves tenham sido trocadas, o que leva à sua perda. Porém, logo que as chaves são criadas os pacotes passam a ser respondidos, o que normalmente acontece a partir do terceiro pacote de ping enviado.

c. Espere alguns segundos, vá para o terminal do roteador1 e encerre a captura de pacotes pressionando Ctrl+C.
O resultado deve ser similar ao abaixo, já que o número de pacotes capturados pode variar:



18. Encerre a simulação:

Aperte o botão  ;
ou utilize o menu Experiment > Stop.

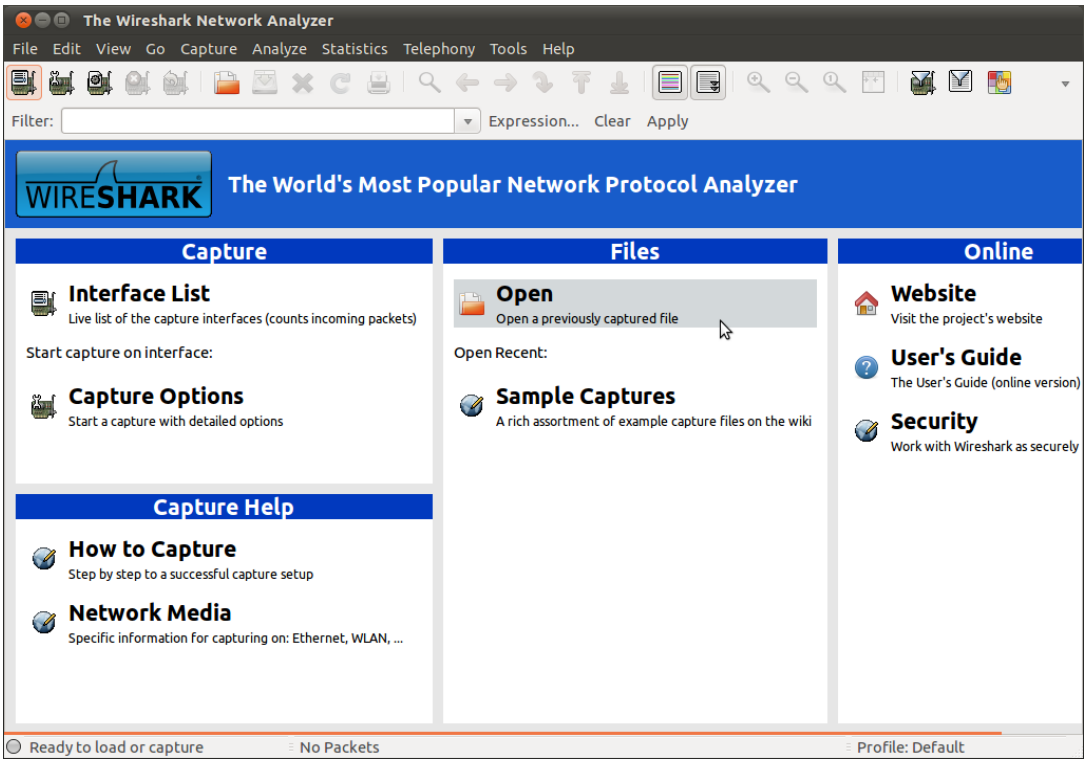
19. Analisaremos agora os pacotes capturados nas duas situações:

- Topologia com somente políticas de segurança
- Topologia com racoon

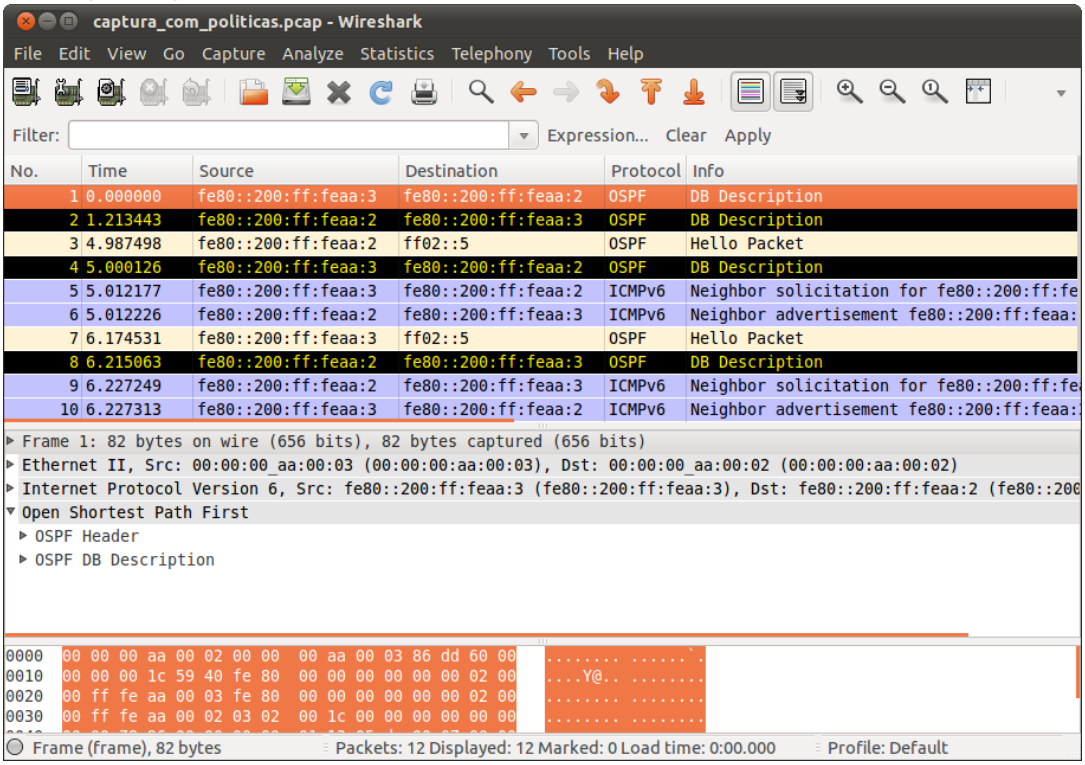
Para verificar os pacotes capturados, utilizaremos o programa Wireshark.

a. Inicie o programa Wireshark. Uma maneira de fazê-lo é através de um terminal na máquina virtual, com o comando:

```
$ wireshark
```

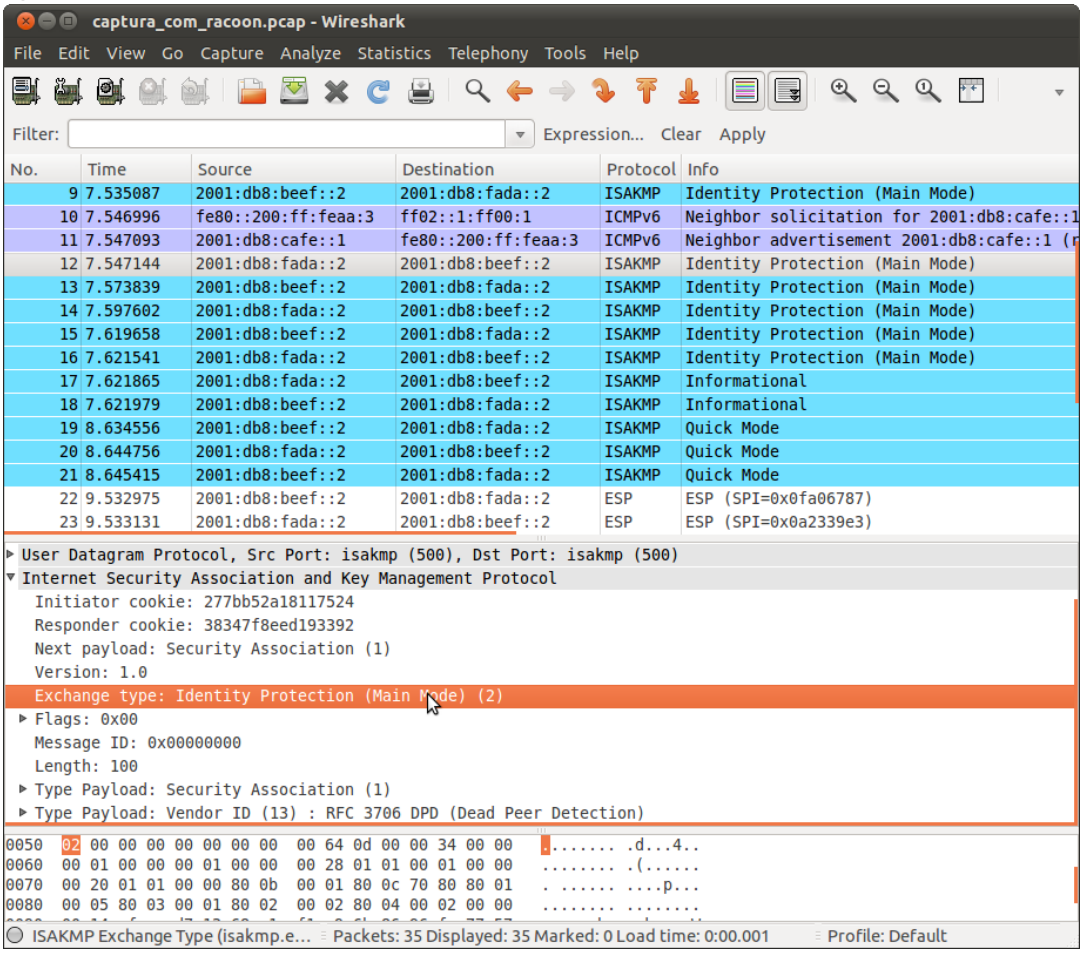


b. Abra o arquivo `/tmp/captura_com_politicas.pcap` com o menu File>Open. Aparecerá uma tela muito semelhante a esta:



Conforme já mencionado anteriormente, ao fazer a captura de pacotes com as políticas de segurança implementadas mas sem as chaves AH ou ESP terem sido criadas, os pacotes que seriam mandados do host1 para o host2 nunca saem do host1, pois o mesmo foi configurado para só enviar pacotes para o seu par IPSec se possuir as chaves do par (e da mesma maneira, ele irá rejeitar pacotes de um par IPSec se este par não tiver as suas chaves). Como o host1 não sabe as chaves do host2 pois ainda não foram configuradas, os pacotes nunca saem do host1. Isto pode ser verificado na captura pela ausência de mensagens ICMPv6 ou de mensagens com cabeçalho IPv6 seguido de cabeçalho de criptografia (o ICMPv6 estaria escondido dentro do cabeçalho de criptografia ESP).

c. Abra o arquivo `/tmp/captura_com_racoon.pcap` com o menu File>Open. Aparecerá uma tela muito semelhante a esta:



Nesta captura pode-se enxergar todas as trocas de pacotes ISAKMP antes das chaves de autenticação e criptografia serem trocadas entre o par de máquinas. Quando um primeiro envio de pacotes é inicializado, o remetente envia pacotes ISAKMP do tipo “Identity Protection”, esperando pela resposta do destinatário para enviar cada um deles em sequência. O primeiro tipo de pacote ISAKMP é o “Security Association” onde o remetente gera um desafio “Initiator Cookie” e usando a senha pré-compartilhada o destinatário irá responder preenchendo o valor do “Responder Cookie”. Se o valor for igual ao esperado pelo remetente a chave pré-compartilhada foi confirmada e o canal seguro é estabelecido com as mensagens “Identify Protection” do tipo “Key Exchange” e “Identification”, terminando assim a fase 1 da troca automática de chaves de segurança IPSec, que é o estabelecimento de uma conexão segura própria do Racoon com a criação de uma SA própria do Racoon.

O processo automático passa, então, para a fase 2, que é a geração e troca automáticas de chaves IPSec. Esta fase pode ser reconhecida pelas mensagens ISAKMP do tipo “Informational” e “Quick Mode”. Uma vez que as chaves são trocadas, o par começa a comunicar-se com a segurança IPSec plenamente estabelecida.

IPv6 - Laboratório de túneis 6over4

Objetivo

Os túneis 6over4 utilizam o protocolo 41, ou 6in4, para prover encapsulamento de pacotes IPv6 em pacotes IPv4, de modo a permitir o tráfego de pacotes IPv6 através de uma rede que suporta apenas IPv4.

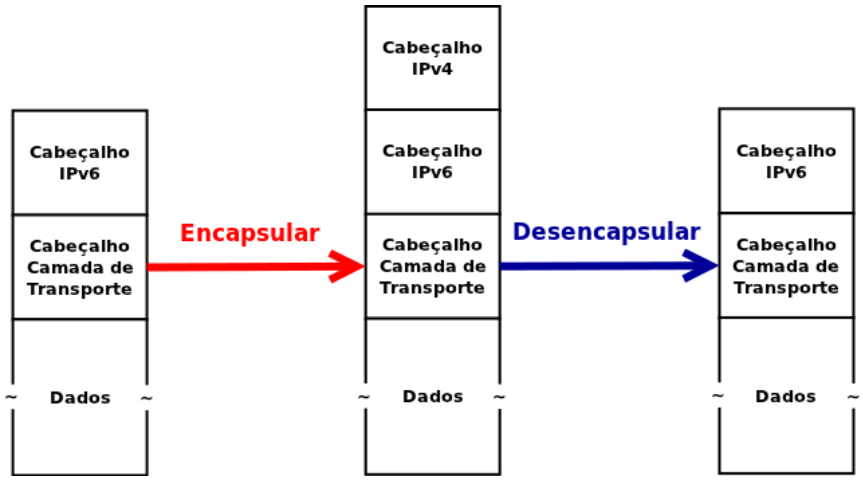
Esse laboratório consiste na configuração manual de túneis 6over4 entre dois computadores, interligados por um roteador que transporta apenas IPv4, a fim de validar seu funcionamento.

Para o presente exercício será utilizado a topologia descrita no arquivo:
tecnicas-transicao-6over4.imn.

Introdução Teórica

Técnicas de tunelamento fazem o encapsulamento de pacotes IPv6 em pacotes IPv4, ou vice versa, permitindo que pacotes de um tipo, trafeguem por uma rede de outro tipo. O encapsulamento de pacotes IPv6 em pacotes IPv4 é conhecido como 6in4 ou IPv6-in-IPv4 (**RFC 4213**). Ele consiste em colocar o pacote IPv6 dentro de um pacote IPv4, diretamente, adequar os endereços de origem e destino para o IPv4 e colocar no cabeçalho o tipo 41 (29 em hexadecimal). Por conta disso o encapsulamento 6in4 é conhecido também como “protocolo 41”. Quando o destino receber o pacote com tipo 41 ele irá remover o cabeçalho IPv4 e tratar o pacote como IPv6. Esse tipo de túnel pode ser utilizado para contornar um equipamento ou enlace sem suporte a IPv6 numa rede, ou para criar túneis estáticos entre duas redes IPv6 através da Internet IPv4.

A figura seguinte exemplifica visualmente como o processo de encapsulamento de IPv6 em IPv4 acontece.



A técnica 6over4 (**RFC 4213**) utiliza um túnel estabelecido manualmente entre dois hosts para enviar o tráfego IPv6. Todo o tráfego IPv6 a ser enviado é encapsulado em IPv4 através do 6in4, explicado anteriormente.

Roteiro Experimental

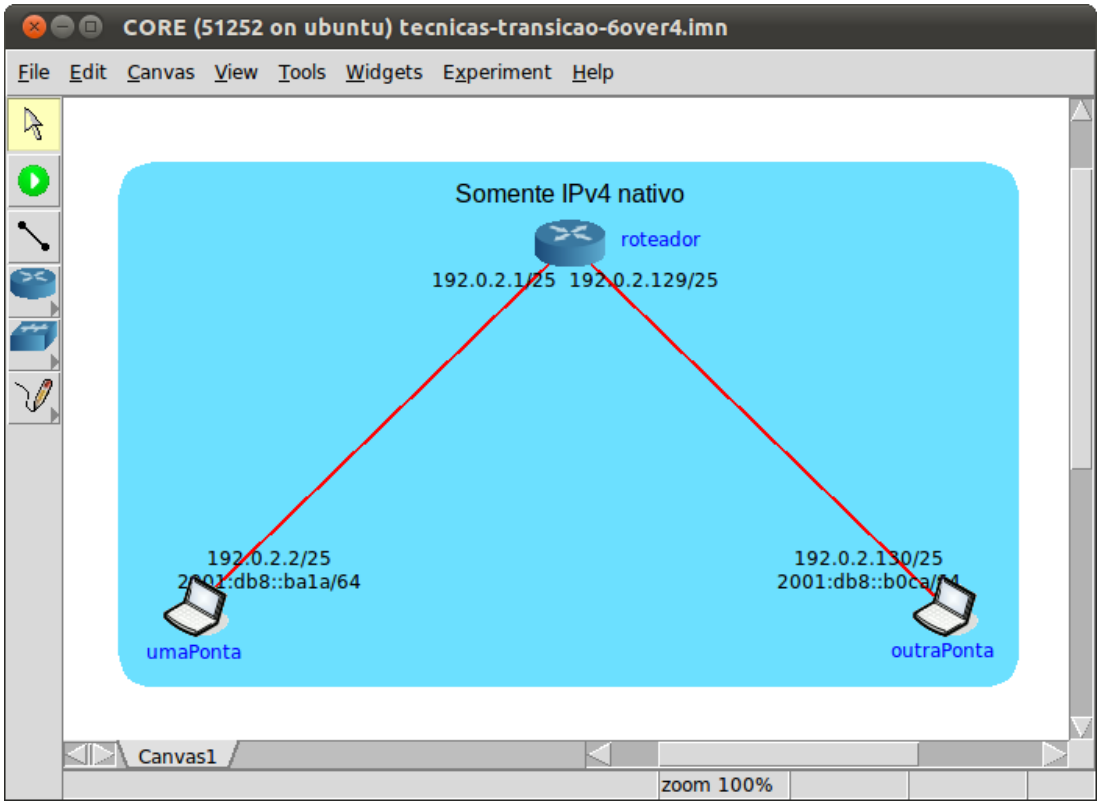
Experiência 1 - Túnel 6over4

1. Para fazer algumas verificações durante o experimento também será necessária a utilização do programa Wireshark que realiza a verificação dos pacotes que são enviados na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

   Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

2. Inicie o CORE e abra o arquivo “tecnicas-transicao-6over4.imn”. A seguinte topologia inicial de rede deve aparecer deve aparecer:



O objetivo dessa topologia de rede é representar o mínimo necessário para que o túnel 6in4 seja entendido. O roteador entre os dois hosts transporta apenas IPv4. A rede foi configurada com rotas estáticas de forma que todas as máquinas possam conectar-se via IPv4.

3. Verifique a configuração dos nós da topologia.

a. Inicie a simulação:

i. aperte o botão  ; ou

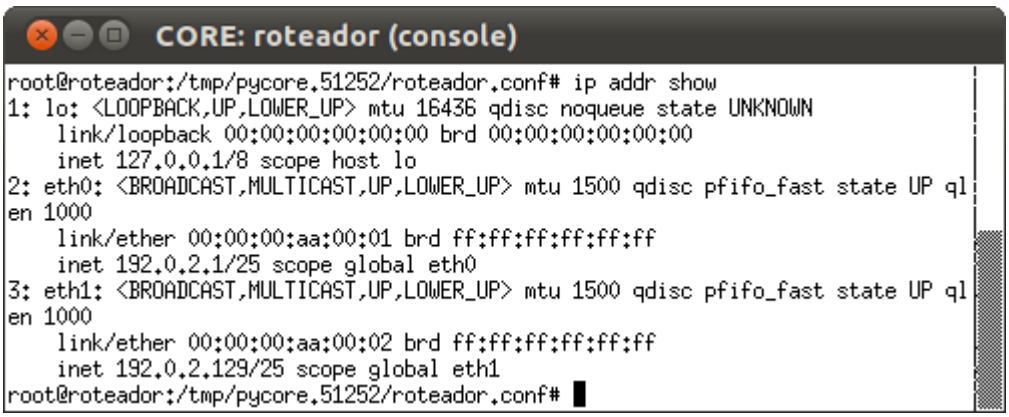
ii. utilize o menu Experiment > Start.

b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do roteador, através do duplo-clique.

c. Verifique que o roteador não suporta IPv6 através do seguinte comando:

```
# ip addr show
```

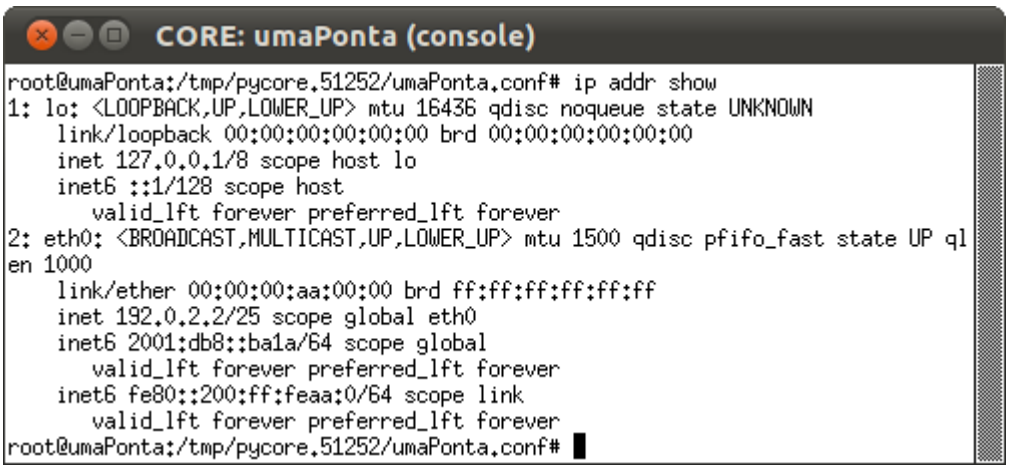
O resultado deve ser:

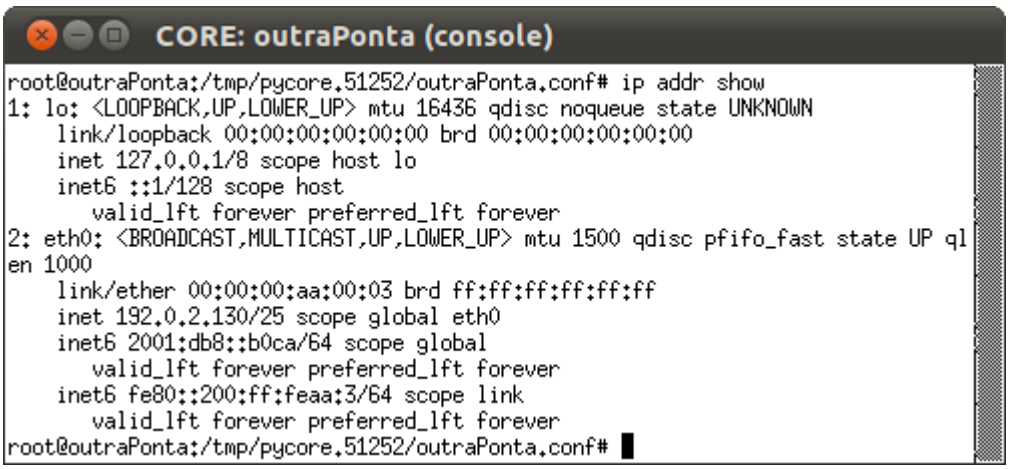


Pode observar-se que não há endereços IPv6 nas interfaces, nem mesmo endereços do tipo link local, o que indica que o roteador não suporta o protocolo.

d. Verifique também para os nós umaPonta e outraPonta.

O resultado deve ser:





Verifique que ambos os nós possuem endereços IPv4 e IPv6.

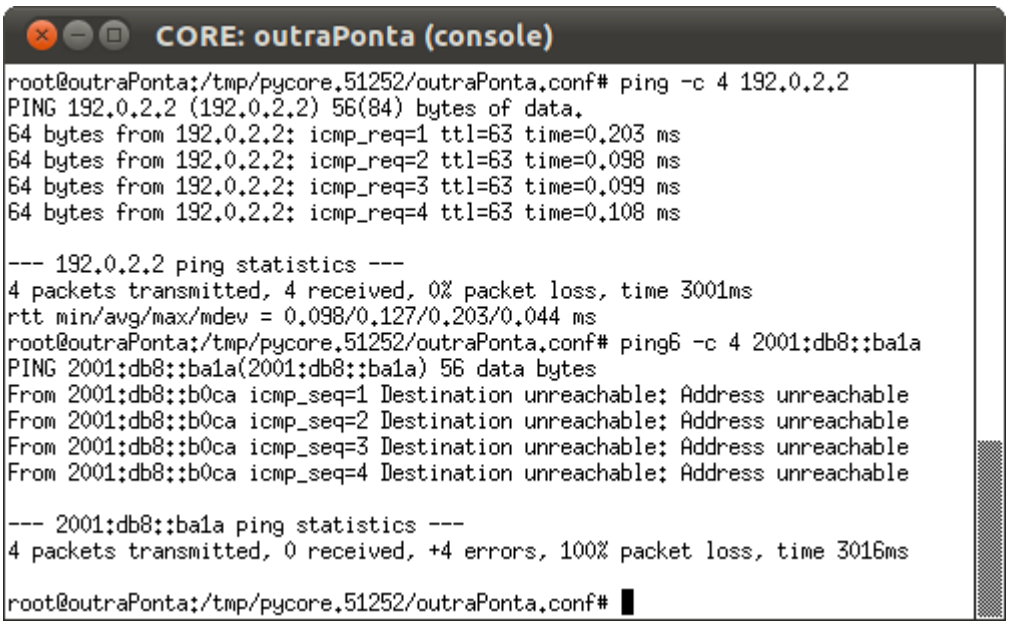
4. Verifique a conectividade entre umaPonta e outraPonta.

   a. Abra o terminal de outraPonta, através do duplo-clique.

   b. Utilize os seguintes comandos para verificar a conectividade:

```
# ping -c 4 192.0.2.2
# ping6 -c 4 2001:db8::ba1a
```

   O resultado deve ser:



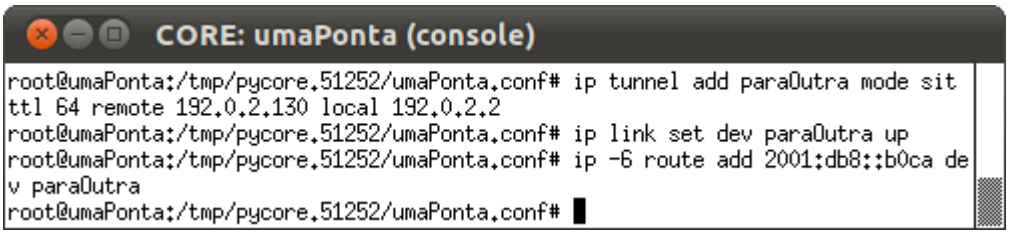
   Observe que há conectividade IPv4, mas ainda não há conectividade IPv6. O túnel a ser criado proverá essa conectividade.

5. Configure o túnel 6in4 entre umaPonta e outraPonta.

   a. Abra o terminal de umaPonta através do duplo-clique e utilize os seguintes comandos para a configuração:

```
# ip tunnel add paraOutra mode sit ttl 64 remote 192.0.2.130
  local 192.0.2.2
# ip link set dev paraOutra up
# ip -6 route add 2001:db8::b0ca dev paraOutra
```

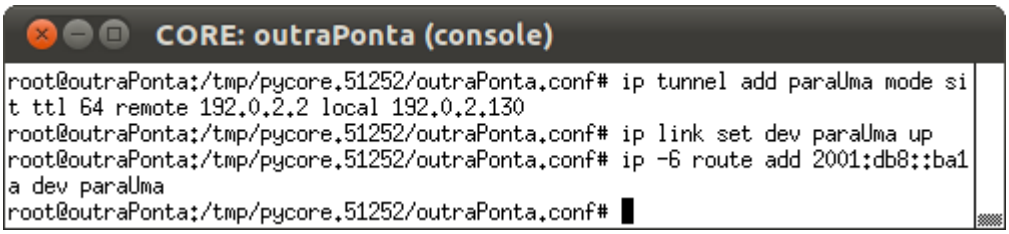
O resultado deve ser:



b. Abra o terminal de outraPonta através do duplo-clique e utilize os seguintes comandos para a configuração:

```
# ip tunnel add paraUma mode sit ttl 64 remote 192.0.2.2
  local 192.0.2.130
# ip link set dev paraUma up
# ip -6 route add 2001:db8::ba1a dev paraUma
```

O resultado deve ser:



Observe que em cada um dos nós o túnel foi criado especificando-se o nome de interfaces de rede virtuais: paraUma e paraOutra, o tipo de túnel: **sit**, que é o nome utilizado pelo Linux para identificar o encapsulamento 6in4. Especificou-se também os endereços de origem e destino IPv4. Além disso foi criada uma rota estática para a outra rede, que aponta para a interface virtual criada para o túnel. Pode-se ainda utilizar comandos como: `ip addr show` e `ip -6 route show` para verificar que a interface de túnel foi criada e que as rotas foram estabelecidas.

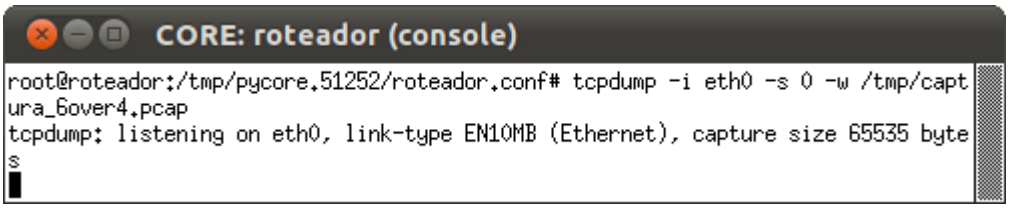
6. Verifique a conectividade via IPv6 entre umaPonta e outraPonta.

a. Abra o terminal do roteador, através do duplo-clique.

b. Utilize o seguinte comando para iniciar a captura de pacotes do roteador:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_6over4.pcap
```

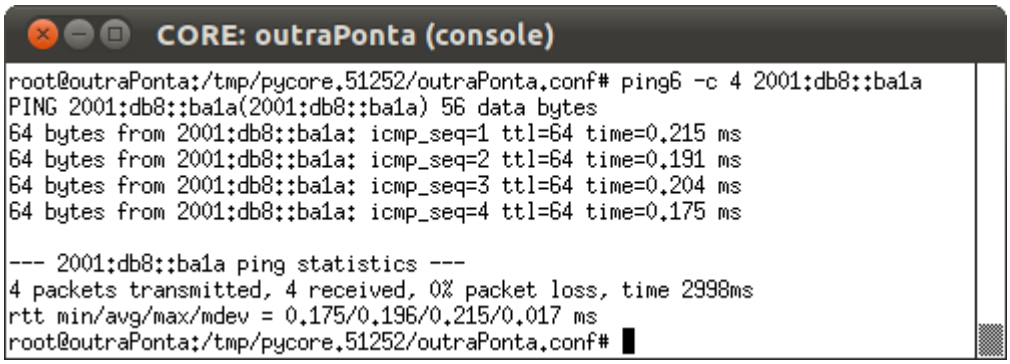
O resultado deve ser:



c. No terminal de outraPonta, utilize novamente o seguinte comando :

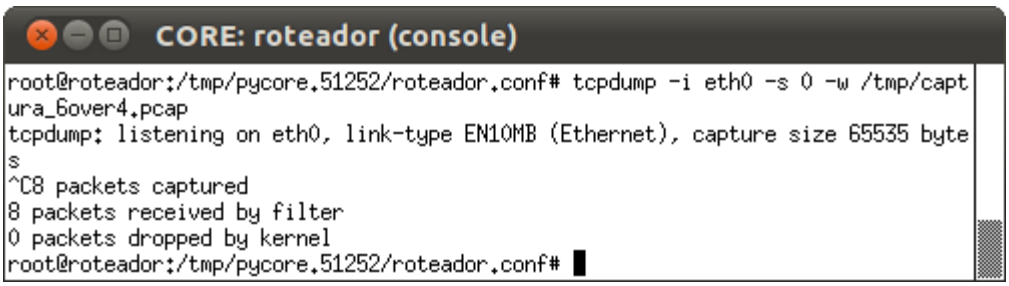
```
# ping6 -c 4 2001:db8::ba1a
```

O resultado deve ser:



d. No terminal do roteador, encerre a captura de pacotes através da sequência Ctrl+C.

O resultado deve ser:

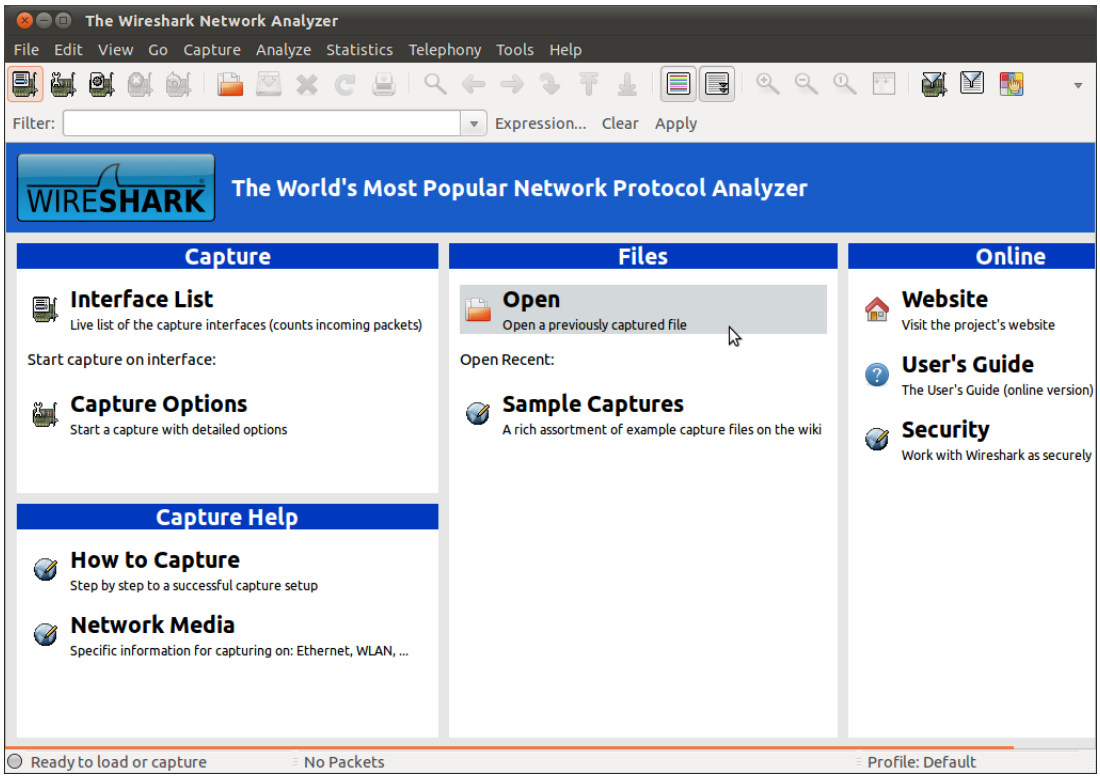


7. Encerre a simulação:

a. aperte o botão  ; ou
b. utilize o menu Experiment > Stop.

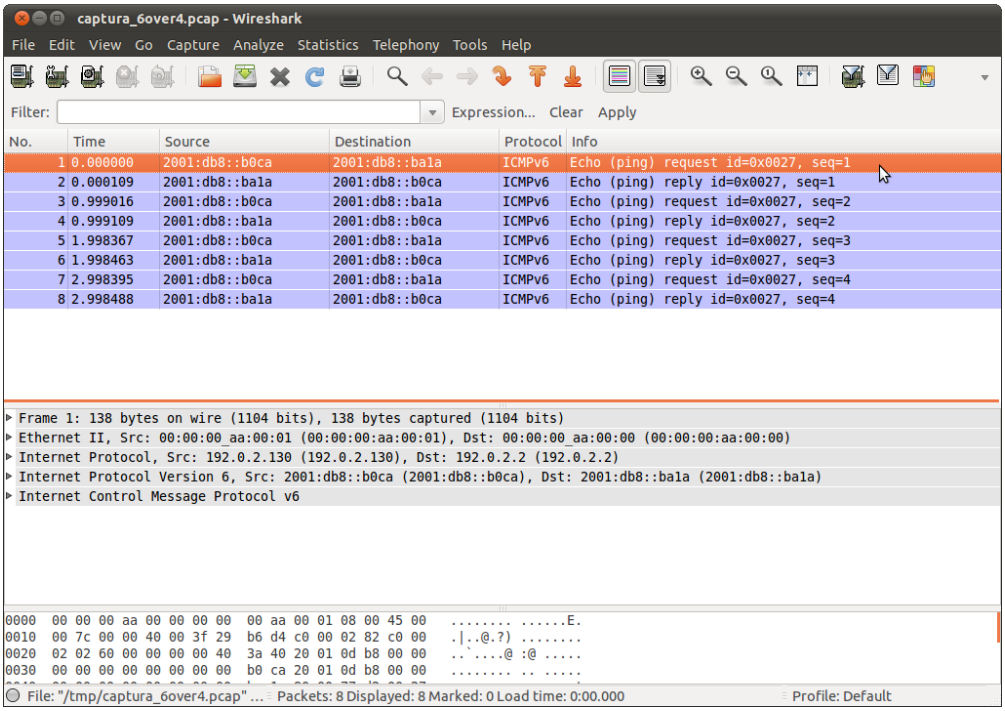
8. Para verificar os pacotes capturados, utilizaremos o programa Wireshark. Para abri-lo, inicie o programa wireshark. Uma maneira é através de um terminal na máquina virtual com o comando:

```
$ wireshark
```

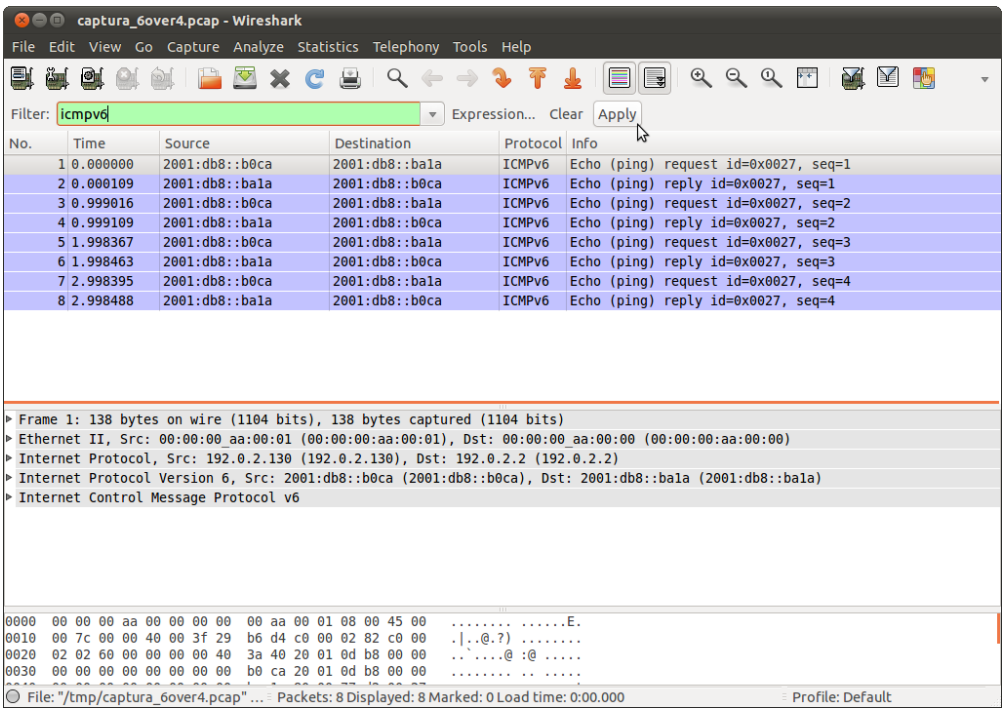


Esse programa tem como principais funcionalidades a captura e análise de pacotes transmitidos por uma interface de rede. Através de seu uso, é possível visualizar os pacotes que trafegam pela rede. Verifique o arquivo de captura previamente obtido.

a. Abra o arquivo `/tmp/captura_6over4.pcap` com o menu File>Open:

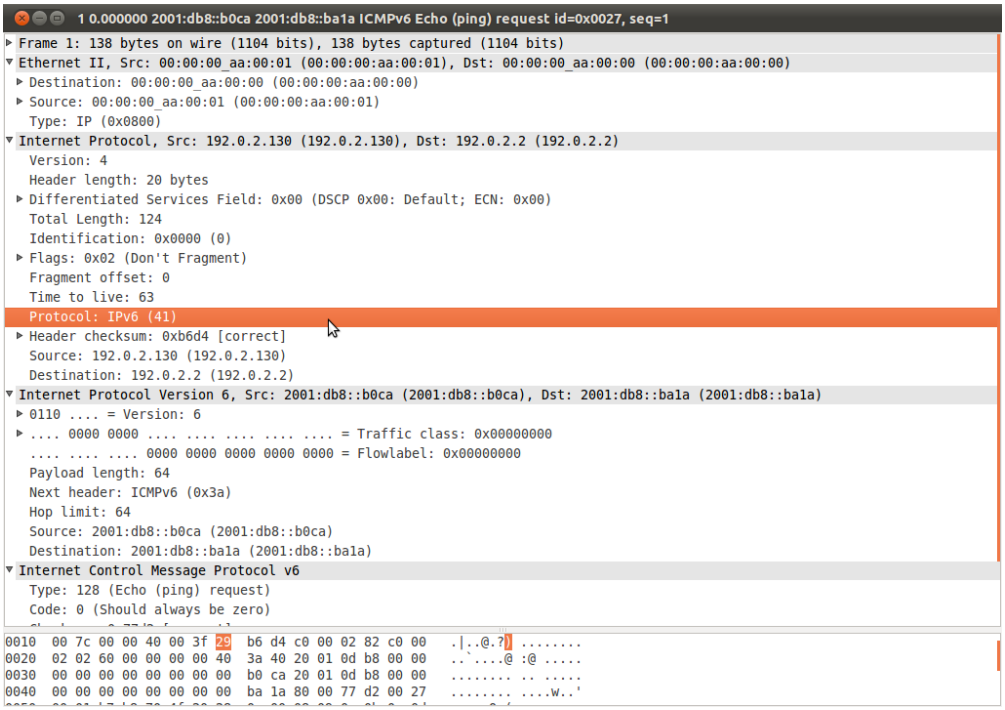


b. Pode-se utilizar o filtro icmpv6 para visualizar apenas os pacotes que queremos observar:



Note que as versões mais atuais do Wireshark são inteligentes o suficiente para mostrar os pacotes como do tipo ICMPv6, mesmo eles estando encapsulados em pacotes IPv4.

Analise os pacotes *echo request* e *echo reply* veja se os dados contidos nos pacotes conferem com a teoria, prestando atenção à forma com que os pacotes IPv6 foram encapsulados em pacotes IPv4.



Campos importantes:

- Type (camada Ethernet): indica que a mensagem utiliza o protocolo IPv4.
- Protocol (camada IPv4): indica que a mensagem encapsula o protocolo IPv6 (protocolo número 41 - 6in4).
- Destination (camada IPv4): o destino é o endereço IPv4 de umaPonta (192.0.2.2).
- Source (camada IPv4): a fonte é o endereço IPv4 de outraPonta (192.0.2.130).
- Destination (camada IPv6): o destino é o endereço IPv6 de umaPonta (2001:db8::ba1a).
- Source (camada IPv6): a fonte é o endereço IPv6 de outraPonta (2001:db8::b0ca).

IPv6 - Laboratório de túneis GRE

Objetivo

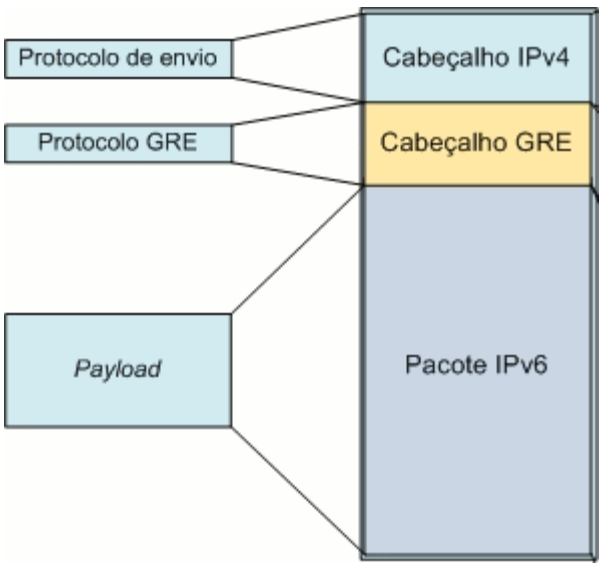
Os túneis GRE (Generic Routing Encapsulation) foram desenvolvidos para transportar diversos tipos de protocolos. Especificamente, neste experimento, serão encapsulados pacotes IPv6 em pacotes IPv4, usando GRE, de modo a permitir o tráfego de pacotes IPv6 através uma rede que suporta apenas IPv4.

Esse laboratório consiste na configuração manual de túneis GRE entre dois computadores, interligados por um roteador que transporta apenas IPv4, a fim de validar seu funcionamento.

Para o presente exercício será utilizado a topologia descrita no arquivo:
tecnicas-transicao-GRE.imn.

Introdução Teórica

Técnicas de tunelamento, no contexto da transição para IPv6, fazem o encapsulamento deste em IPv4, ou vice versa, permitindo que pacotes de um tipo, trafeguem por uma rede de outro tipo. O túnel GRE (**RFC 2784**) é um túnel estático entre dois hosts originalmente desenvolvido com a finalidade de encapsular vários tipos diferentes de protocolos. Este tipo de encapsulamento é suportado na maioria do sistemas operacionais e roteadores, e consiste em um link ponto a ponto. Sua configuração é manual, de modo que pode gerar um esforço em sua manutenção e gerenciamento proporcional à quantidade de túneis.



O funcionamento deste tipo de túnel é muito simples: consiste em pegar os pacotes originais, adicionar o cabeçalho GRE e o cabeçalho IPv4, e enviar ao IP de destino. Quando o pacote encapsulado chegar na outra ponta do túnel (IP de destino) remove-se dele os cabeçalhos IPv4 e GRE, restando apenas o pacote original, que é encaminhado normalmente ao destinatário.

Roteiro Experimental

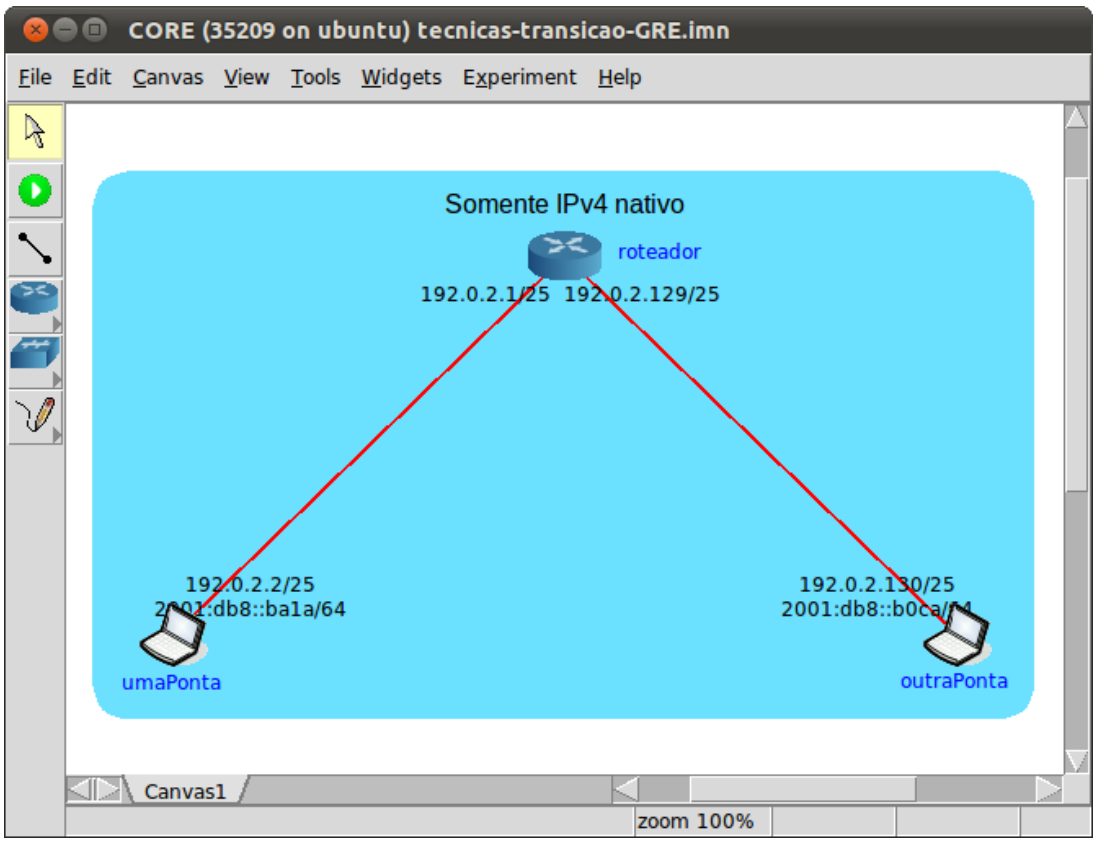
Experiência 2 - Túnel GRE

1. Para fazer algumas verificações durante o experimento também será necessária a utilização do programa Wireshark que realiza a verificação dos pacotes que são enviados na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

   Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

2. Inicie o CORE e abra o arquivo “tecnicas-transicao-GRE.imn”. A seguinte topologia inicial de rede deve aparecer deve aparecer:



   O objetivo dessa topologia de rede é representar o mínimo necessário para que o túnel GRE seja entendido. O roteador entre as duas pontas do túnel transporta apenas IPv4. A rede foi configurada com rotas estáticas de forma que todas as

máquinas possam conectar-se via IPv4.

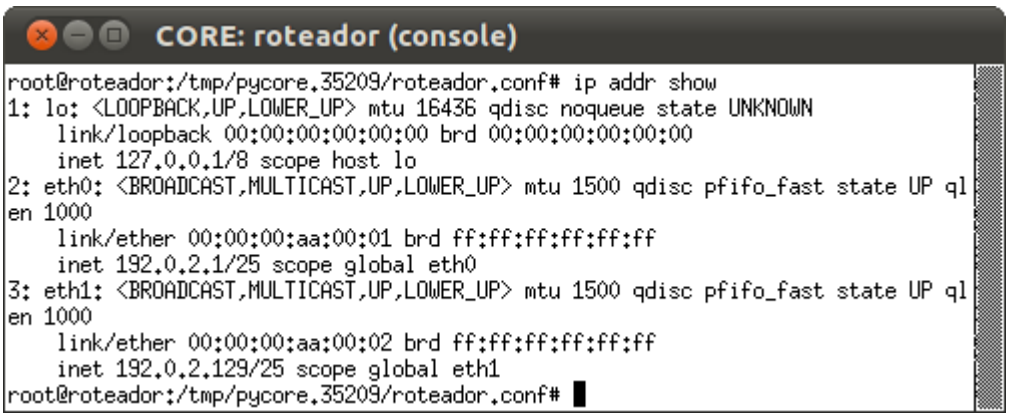
3. Verifique a configuração dos nós da topologia.
   a. Inicie a simulação:
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do roteador, através do duplo-clique.

   c. Verifique que o roteador não suporta IPv6 através do seguinte comando:

```
# ip addr show
```

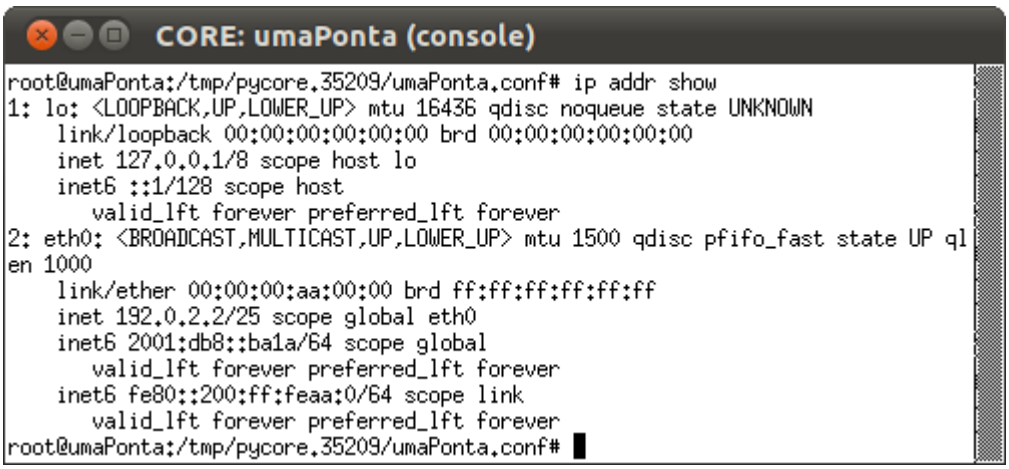
   O resultado deve ser:

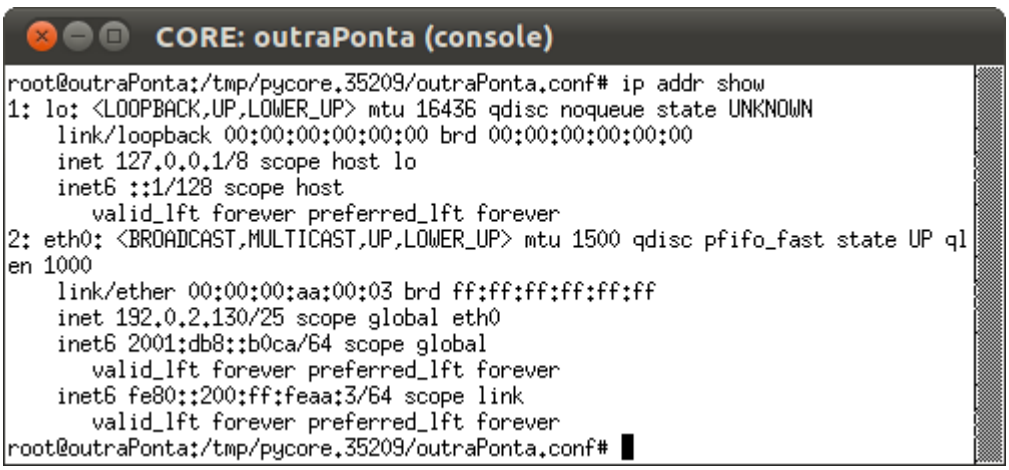


   Pode observar-se que não há endereços IPv6 nas interfaces, nem mesmo endereços do tipo link local, o que indica que o roteador não suporta o protocolo.

   d. Verifique também para os nós umaPonta e outraPonta.

O resultado deve ser:





Verifique que ambos os nós possuem endereços IPv4 e IPv6.

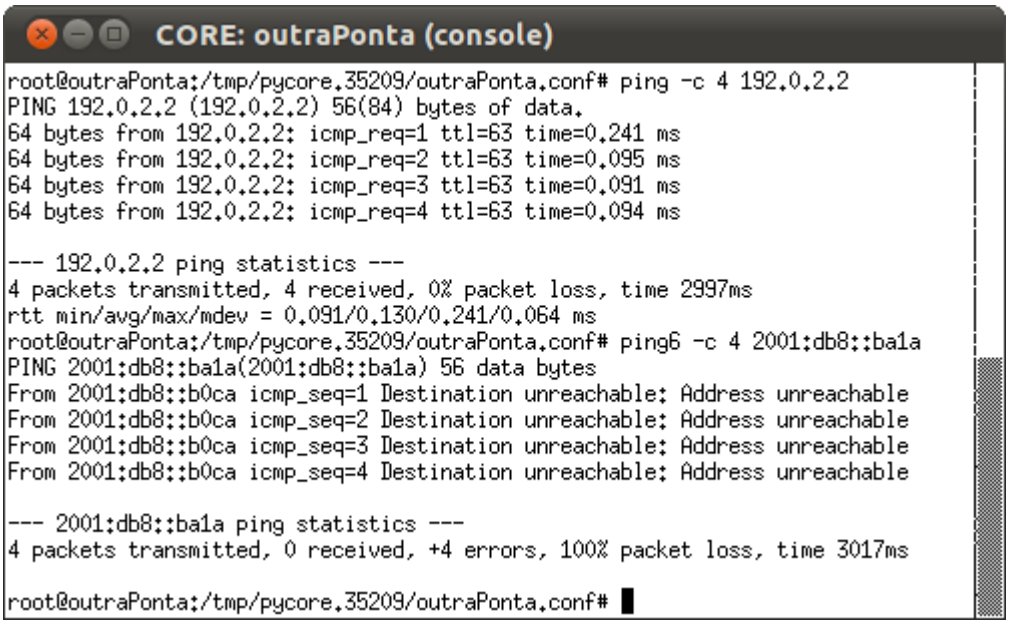
4. Verifique a conectividade entre umaPonta e outraPonta.

   a. Abra o terminal de outraPonta, através do duplo-clique.

   b. Utilize os seguintes comandos para verificar a conectividade:

```
# ping -c 4 192.0.2.2
# ping6 -c 4 2001:db8::ba1a
```

O resultado deve ser:



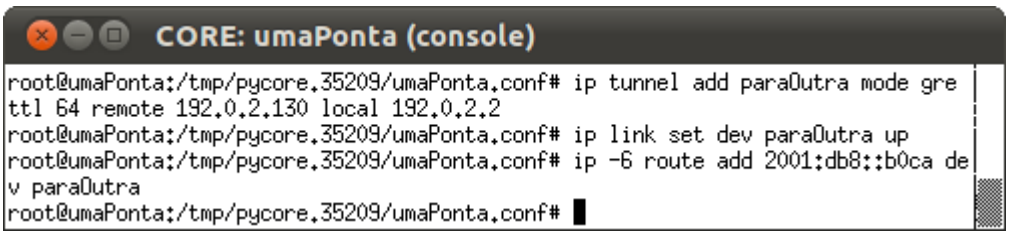
Observe que há conectividade IPv4, mas ainda não há conectividade IPv6. O túnel a ser criado proverá essa conectividade.

5. Configure o túnel GRE entre umaPonta e outraPonta.

a. Abra o terminal de umaPonta através do duplo-clique e utilize os seguintes comandos para a configuração:

```
# ip tunnel add paraOutra mode gre ttl 64 remote 192.0.2.130
  local 192.0.2.2
# ip link set dev paraOutra up
# ip -6 route add 2001:db8::b0ca dev paraOutra
```

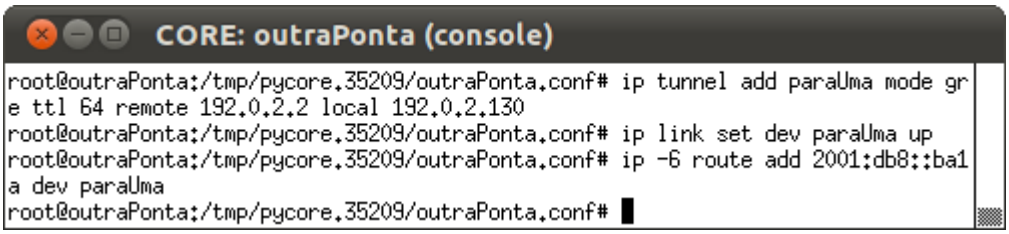
O resultado deve ser:



b. Abra o terminal de outraPonta através do duplo-clique e utilize os seguintes comandos para a configuração:

```
# ip tunnel add paraUma mode gre ttl 64 remote 192.0.2.2
  local 192.0.2.130
# ip link set dev paraUma up
# ip -6 route add 2001:db8::ba1a dev paraUma
```

O resultado deve ser:



Observe que em cada um dos nós o túnel foi criado especificando-se o nome de interfaces de rede virtuais: paraUma e paraOutra, o tipo de túnel: **gre**, que é o nome utilizado pelo Linux para identificar o encapsulamento GRE. Especificou-se também os endereços de origem e destino IPv4. Além disso foi criada uma rota estática para a outra rede, que aponta para a interface virtual criada para o túnel.

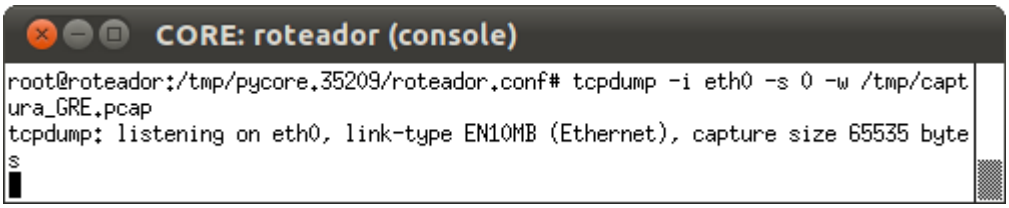
6. Verifique a conectividade via IPv6 entre umaPonta e outraPonta.

a. Abra o terminal do roteador, através do duplo-clique.

b. Utilize o seguinte comando para iniciar a captura de pacotes do roteador:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_GRE.pcap
```

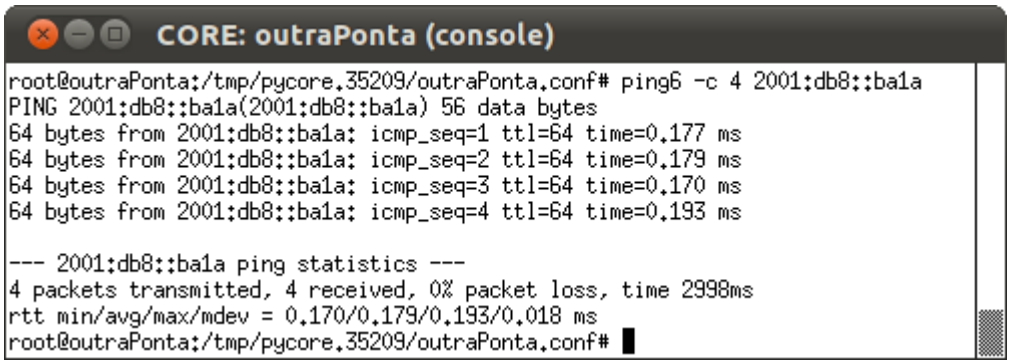
O resultado deve ser:



c. No terminal de outraPonta, utilize novamente o seguinte comando :

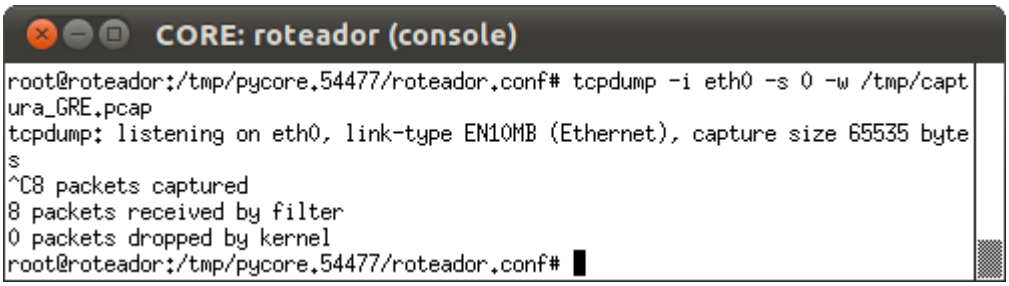
```
# ping6 -c 4 2001:db8::ba1a
```

O resultado deve ser:



d. No terminal do roteador, encerre a captura de pacotes através da sequência Ctrl+C.

O resultado deve ser:

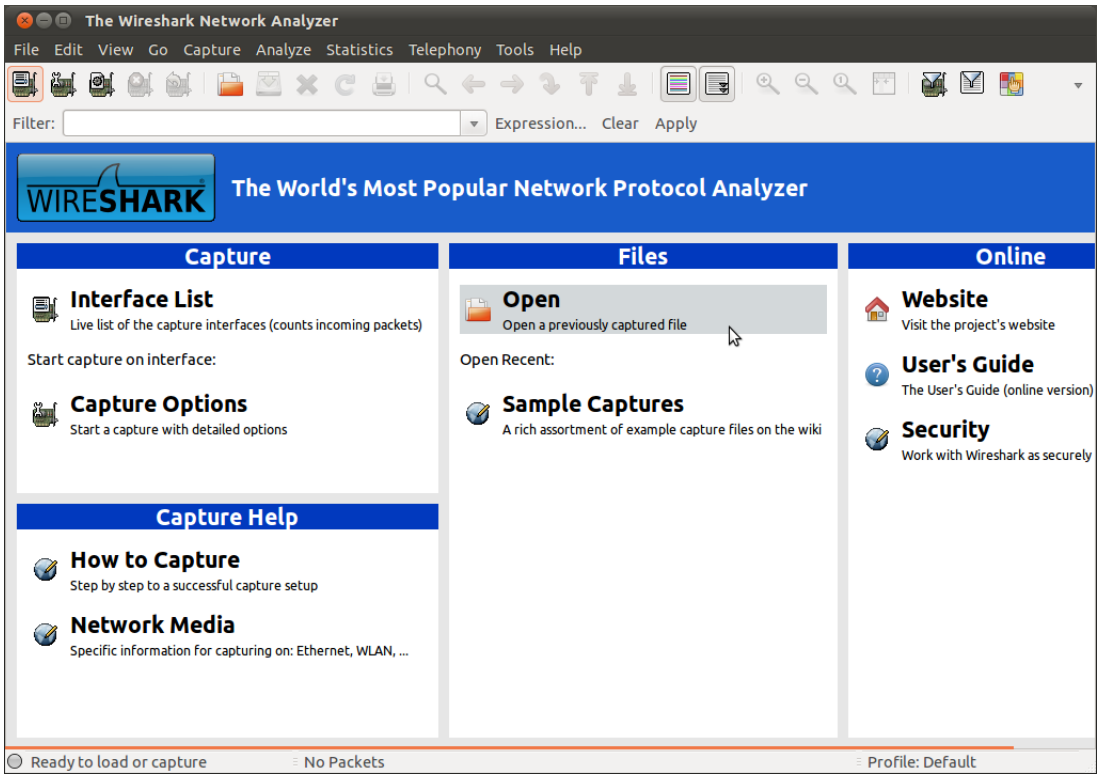


7. Encerre a simulação:

a. aperte o botão  ; ou
b. utilize o menu Experiment > Stop.

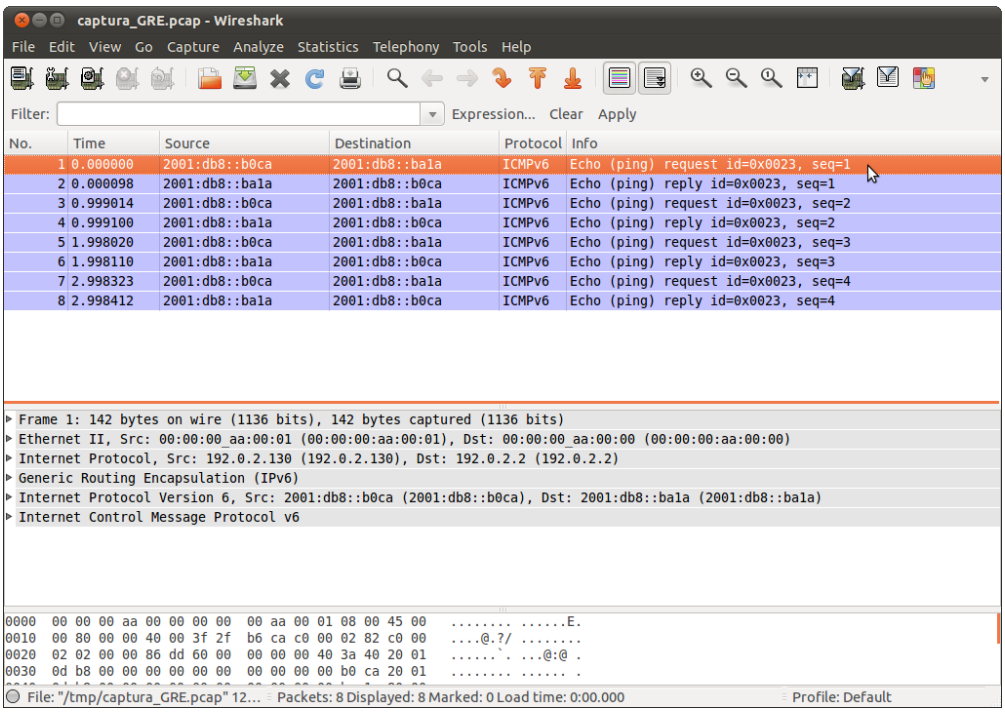
8. Para verificar os pacotes capturados, utilizaremos o programa Wireshark. Para abri-lo, inicie o programa wireshark. Uma maneira é através de um terminal na máquina virtual com o comando:

```
$ wireshark
```



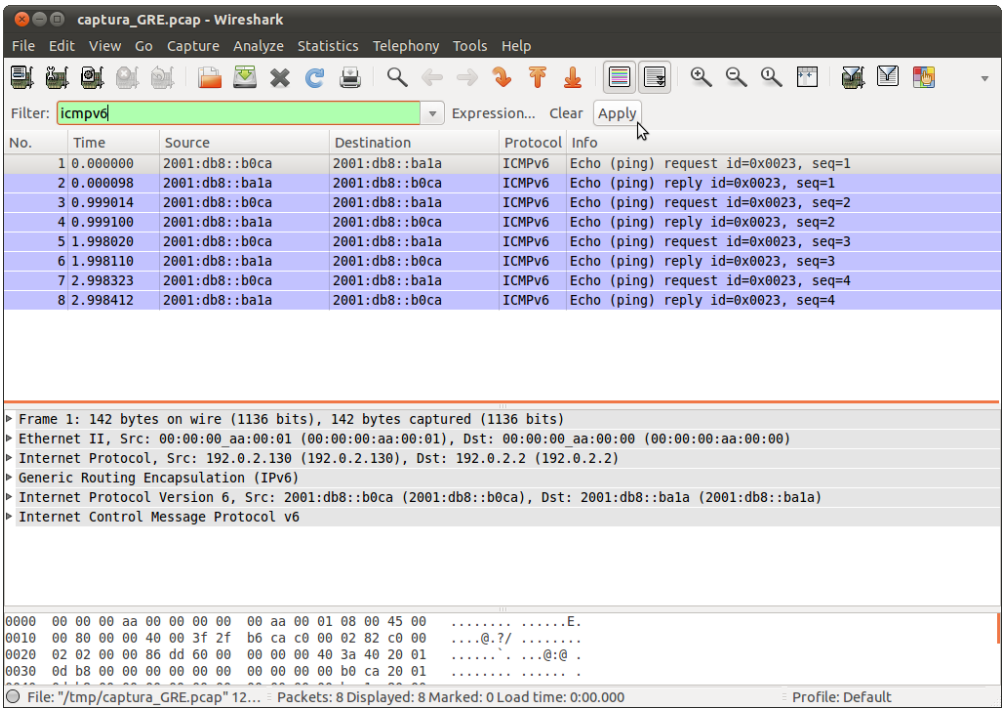
Esse programa tem como principais funcionalidades a captura e análise de pacotes transmitidos por uma interface de rede. Através de seu uso, é possível melhor visualizar os pacotes que trafegam pela rede. Verifique o arquivo de captura previamente obtido.

a. Abra o arquivo `/tmp/captura_GRE.pcap` com o menu File>Open:



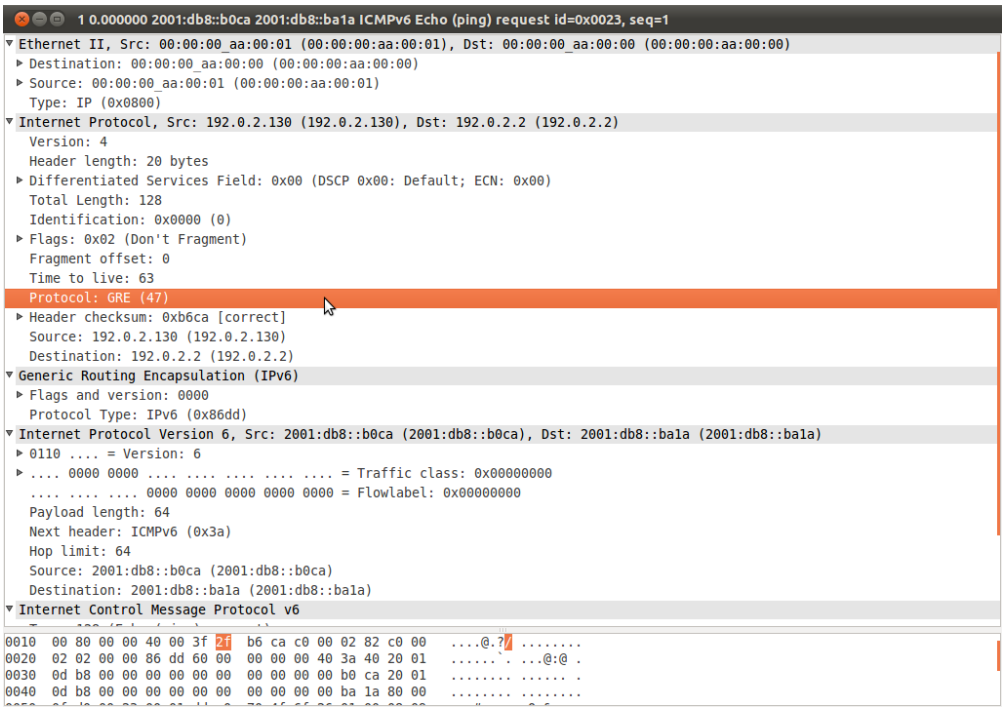
b. Coloque o filtro icmpv6 para visualizar apenas os pacotes que queremos

observar:



Note que as versões mais atuais do Wireshark são inteligentes o suficiente para mostrar os pacotes como do tipo ICMPv6, mesmo eles estando encapsulados em pacotes IPv4, utilizando GRE.

Analise os pacotes *echo request* e *echo reply* veja se os dados contidos nos pacotes conferem com a teoria, prestando atenção à forma com que os pacotes IPv6 foram encapsulados em pacotes IPv4, usando GRE.



Campos importantes:

- Type (camada Ethernet): indica que a mensagem utiliza o protocolo IPv4.
- Protocol (camada IPv4): indica que a mensagem encapsula o protocolo IPv6 (número 47 - GRE).
- Destination (camada IPv4): o destino é o endereço IPv4 de umaPonta (192.0.2.2).
- Source (camada IPv4): a fonte é o endereço IPv4 de outraPonta (192.0.2.130).
- Protocol Type (camada GRE): o protocolo encapsulado é o IPv6.
- Destination (camada IPv6): o destino é o endereço IPv6 de umaPonta (2001:db8::ba1a).
- Source (camada IPv6): a fonte é o endereço IPv6 de outraPonta (2001:db8::b0ca).

IPv6 - Laboratório Dual Stack Lite (DS-Lite)

Objetivo

O principal objetivo desse laboratório é o de apresentar algumas das principais características e funcionalidades da técnica Dual Stack Lite (DS-Lite) a partir da simulação da rede de um ISP inicialmente configurada apenas com o protocolo IPv6. O laboratório está dividido em 3 experiências, a primeira descreve como se implementar o DS-lite, a segunda mostra o quão fácil é expandi-lo para novos clientes e, a terceira, mostra a integração da técnica A+P (Address plus Port) sobre a rede com DS-Lite.

Ele utilizará as topologias descritas nos arquivos: **tecnicas-transicao-dsl1.imn**, **tecnicas-transicao-dsl2.imn** e **tecnicas-transicao-dsl3.imn**.

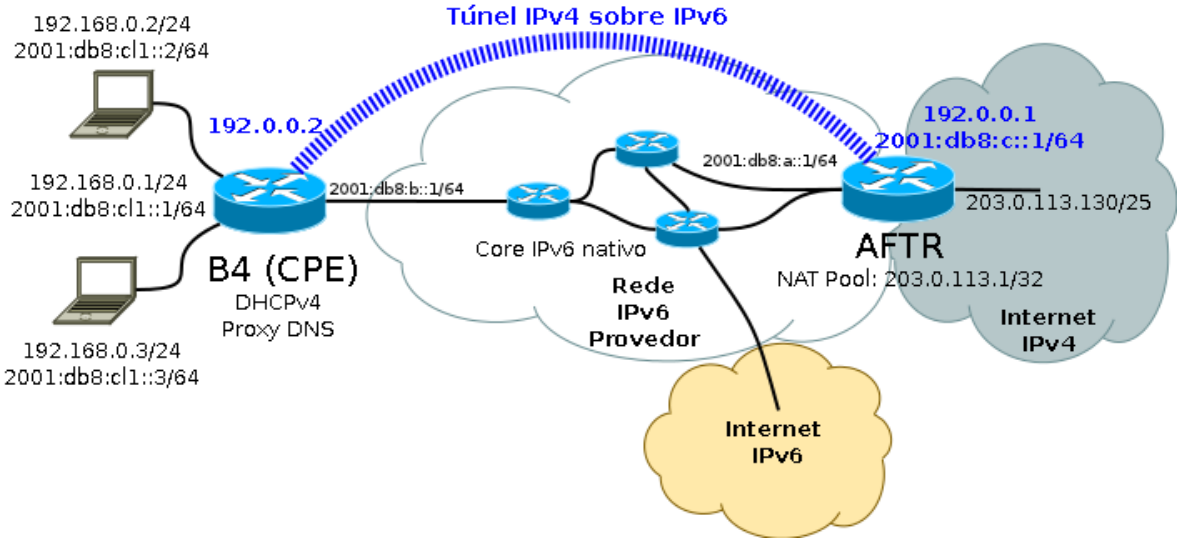
Introdução Teórica

O Dual Stack Lite (Pilha dupla simplificada), é uma técnica padronizada pela **RFC 6333**. Ela pode ser aplicada em situações em que o provedor já oferece IPv6 nativo para seus usuários. Sua implementação necessita de um equipamento denominado AFTR (Address Family Transition Router), que implementa um CGN (Carrier Grade NAT), um NAT de grande porte, na rede do provedor. Entre o AFTR e o CPE Customer Premise Equipment) do usuário utiliza-se um túnel IPv4 sobre IPv6 para transportar o tráfego IPv4. No contexto do DS-Lite, o CPE do usuário é chamado de B4 (DS-Lite Basic Bridging BroadBand). Nas extremidades desses túneis são usados endereços da faixa 192.0.0.0/29, especialmente reservada para este fim. Para o CPE do usuário e os demais equipamentos da rede do usuário são utilizados IPs da **RFC 1918**, e não há problema se diferentes usuários utilizarem faixas de IPs repetidas, dado que o AFTR identifica os diferentes túneis com base no IPv6 de origem dos pacotes encapsulados.

É importante frisar alguns pontos:

- O AFTR usa CGN, mas não força o usuário a utilizar duplo NAT. Ou seja, AFTR realiza a função de NAT, de forma concentrada, para cada um dos dispositivos de cada usuário.
- O DS-Lite utiliza endereços privados na faixa 192.0.0.0/29 para as extremidades dos túneis v4 sobre v6, evitando a utilização desnecessária de endereços IPv4 na infraestrutura do provedor.

A figura abaixo exemplifica o funcionamento do DS-Lite:



Roteiro Experimental

Experiência 2 - Implantação do DS-Lite

1. A principal meta dessa experiência é a simulação da implantação da técnica DS-Lite em uma rede de um ISP (Internet Service Provider). E, com isso, fazer com que o aluno comece a se familiarizar com suas funcionalidades.

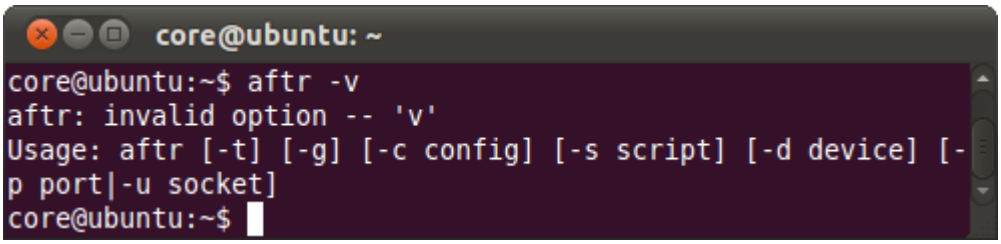
2. Caso você esteja utilizando a máquina virtual fornecida pelo Nic.br pule para o passo 4. Caso contrário, o programa AFTR deve ser instalado com a execução dos seguintes passos:
    b. Acesse a URL a seguir para baixar a última versão do AFTR: http://www.isc.org/software/aftr (acessado 02/05/2012). Nesse site, você deve baixar o arquivo ".tar.gz" que contém o código fonte do programa.
    c. Mova o arquivo baixado para a pasta "/home/core" da VM.
    d. Abra o Terminal e utilize os seguintes comandos para fazer a instalção:

```
$ tar xf aftr-1.1.tar.gz
$ cd aftr-1.1
$ chmod +x configure
$ ./configure
$ make
$ sudo ln -s aftr /usr/bin/aftr
```

lembre-se de trocar o número da versão (no caso 1.1) para a versão baixada.

e. para testar o funcionamento, digite o comando a baixo no terminal e verifique se a saída é equivalente à mensagem mostrada:

```
$ aftr -v
```

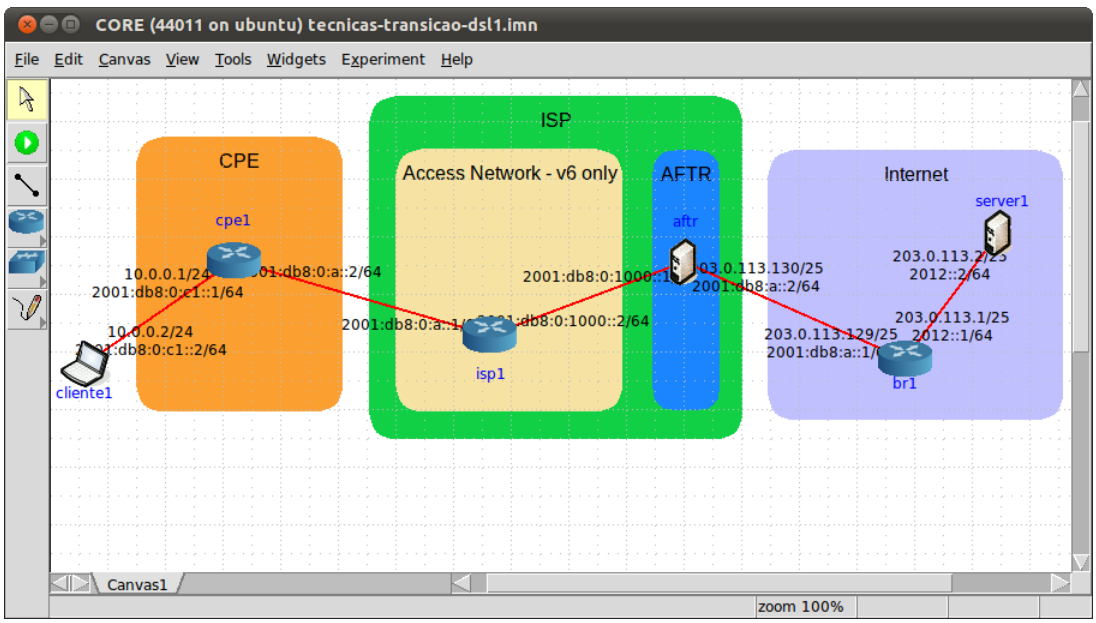


3. Caso não esteja utilizando a Máquina Virtual fornecida pelo NIC.br, o programa Wireshark que é utilizado para se obter uma melhor visualização das informações dos pacotes trafegados em uma rede também deve ser instalado. Para tanto, abra um Terminal da VM e execute o comando:

```
$ sudo apt-get install wireshark
```

Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

4. Com as instalações do AFTR concluídas, inicie o CORE e abra o arquivo “**tecnicas-transicao-dsl1.imn**”. A seguinte topologia deve aparecer:



Essa topologia ilustra a situação em que um determinado ISP possui uma rede unicamente IPv6 funcional e deseja implementar a técnica DS-Lite para que seus clientes passem a ter acesso à Internet IPv4.

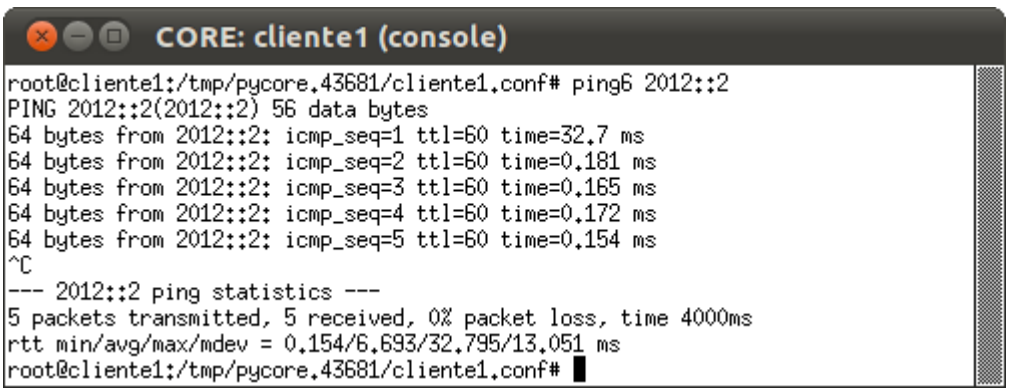
Na situação inicial da simulação a rede foi configurada com rotas estáticas de forma que todas as máquinas pudessem se conectar via IPv6. Porém, na prática, o provedor pode utilizar quais quer outras técnicas para distribuir IPv6 para sua

infra-estrutura e clientes.

5. Verifique a conectividade da máquina cliente:
   b. Inicie a simulação:
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Start.
   c. Espere até que o CORE termine a inicialização da simulação e abra o terminal da máquina cliente através de um duplo-clique sobre a máquina 'cliente1'.
   d. Utilize o comando "ping6" para testar a conectividade com a máquina 'server1' na Internet:

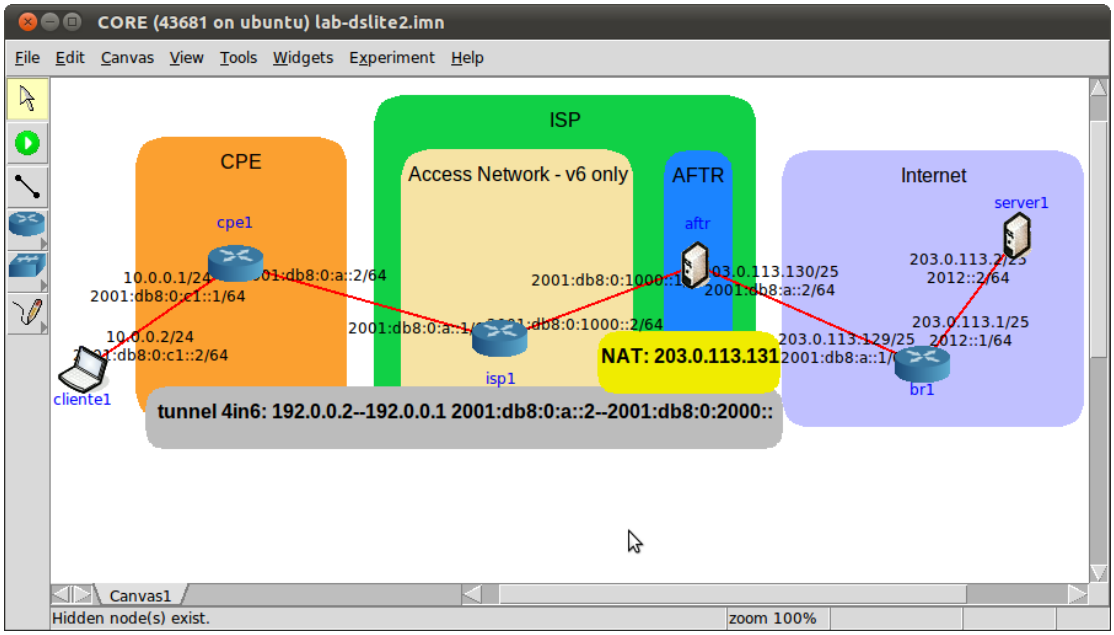
```
# ping6 2012::2
```

   O resultado deve ser:



   Pressione "Ctrl+C" para parar as requisições.

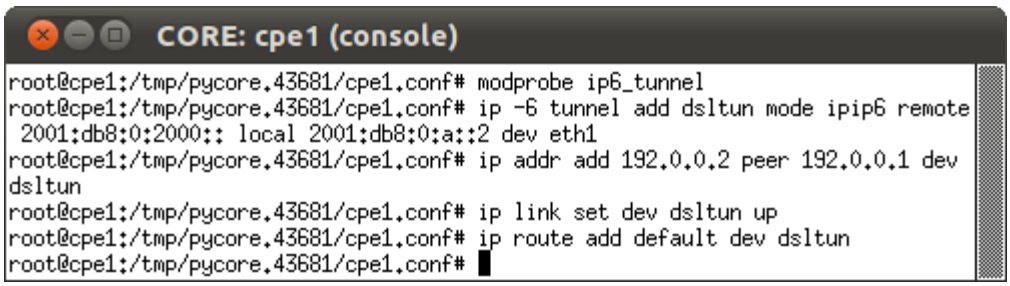
6. A partir desse passo será iniciada a implantação da técnica de transição DS Lite na rede simulada. O servidor de borda do ISP será configurado como um roteador AFTR de modo a realizar a tarefa de NATv4 sobre a rede IPv6 do servidor. Enquanto isso, o CPE será configurado como um B4. Isso é necessário para se possa fechar um túnel que encapsule os pacotes IPv4 na rede IPv6 do ISP. A figura a seguir exemplifica essa situação:



7. A configuração será iniciada pelo CPE. Ainda com o CORE executando a simulação, acesse o terminal da máquina 'cpe1' (duplo clique sobre ela) e rode os seguintes comandos:

```
# modprobe ip6_tunnel
# ip -6 tunnel add dsltun mode ipip6 remote 2001:db8:0:2000:: local
2001:db8:0:a::2 dev eth1
# ip addr add 192.0.0.2 peer 192.0.0.1 dev dsltun
# ip link set dev dsltun up
# ip route add default dev dsltun
```

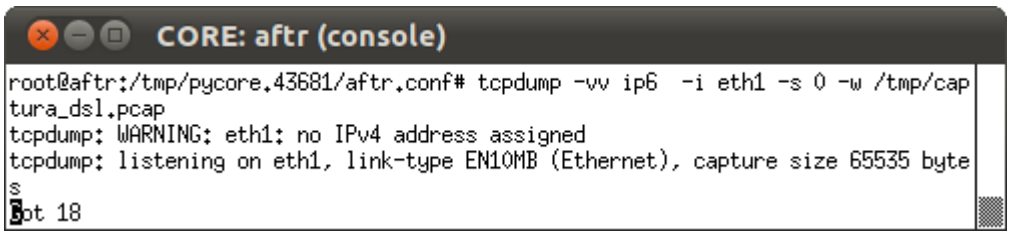
O terminal deve ficar assim:



Essa sequência de comandos cria uma nova interface de rede virtual chamada “dsltun” que encapsula todos os pacotes IPv4 vindos da rede local do cliente em pacotes IPv6 e os envia para o endereço 2001:db8:0:2000:: que será configurado ponta de saída túnel na máquina AFTR.

Para testar, abra o terminal da máquina ‘aftr’ e inicie uma captura dos pacotes que chegam em sua interface eth1. Para isso, utilize o comando a seguir:

```
# tcpdump -vv ip6  -i eth1 -s 0 -w /tmp/captura_dsl.pcap
```



Com o tcpdump escutando a interface eth1, acesse o terminal da máquina 'cliente1' e realize um *ping v4* para a máquina ‘server1’ na Internet:

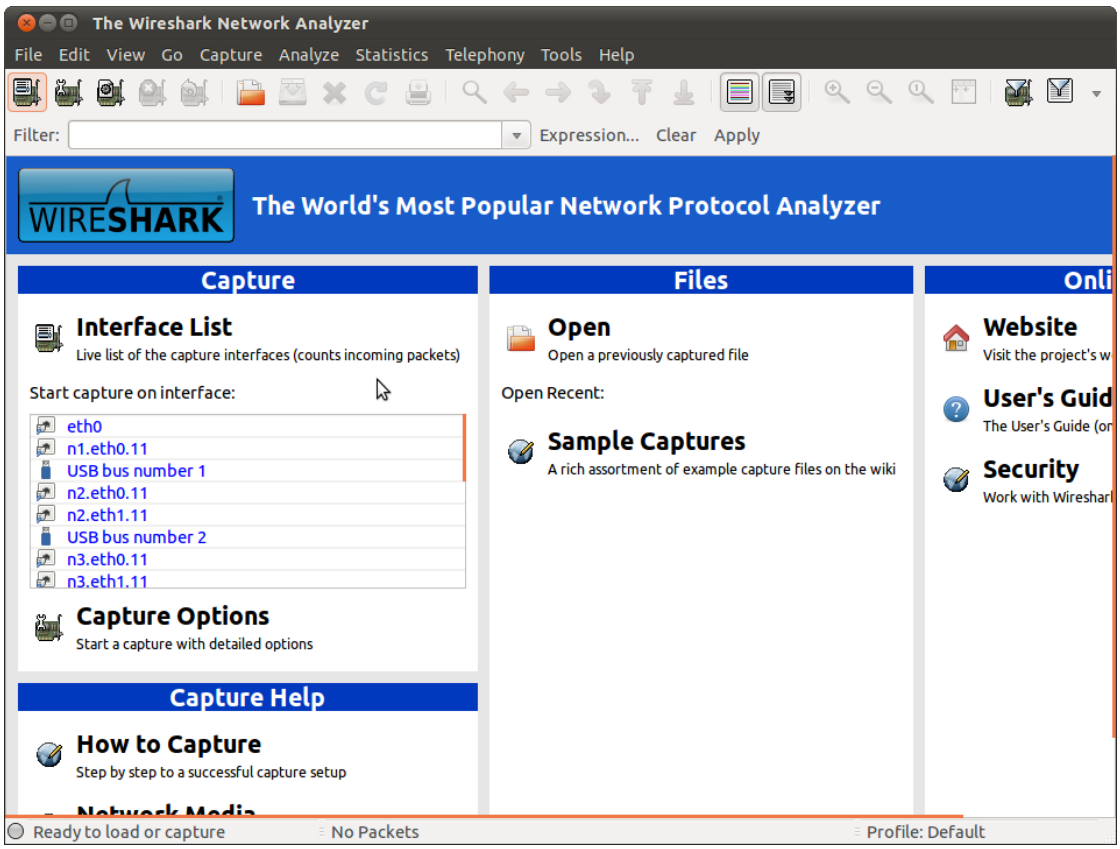
```
# ping 203.0.113.2
```

A tela deve ser similar à seguinte:

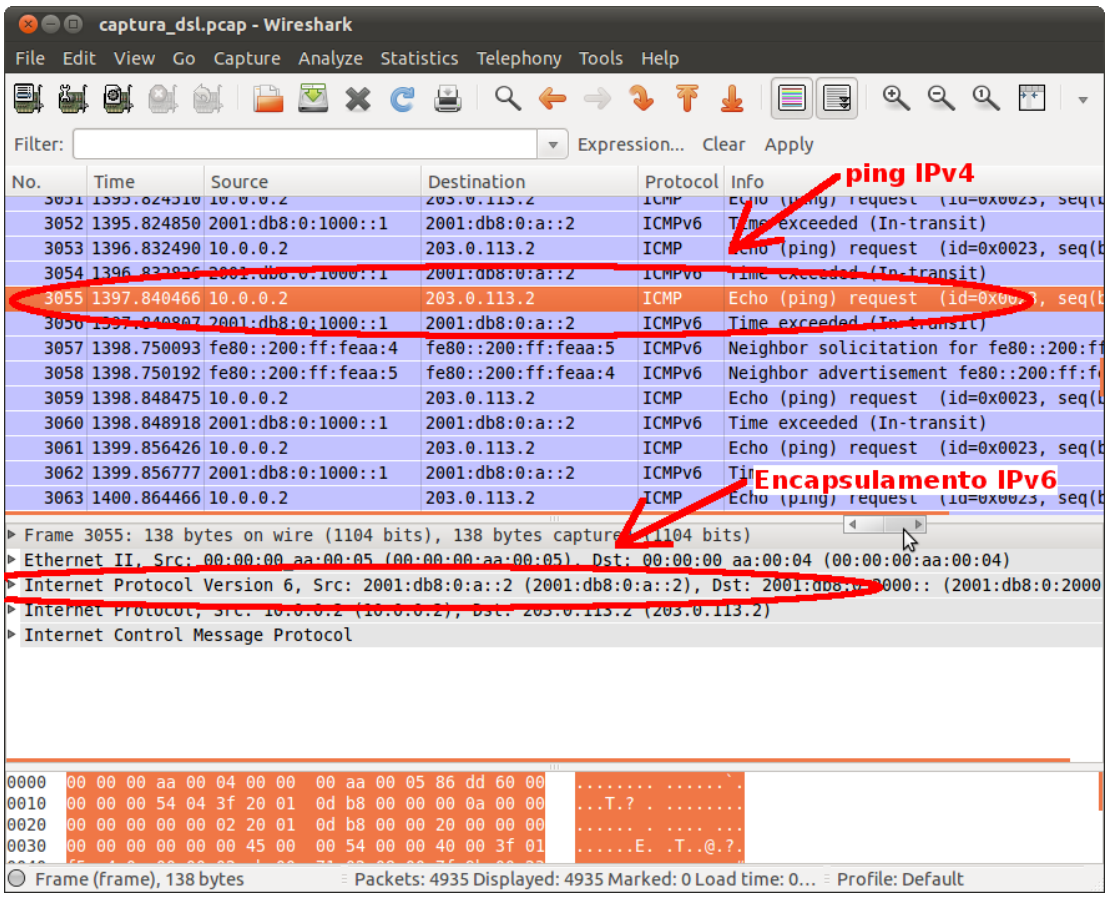
Volte ao terminal do ‘aftr’ e pare a execução do tcpdump com a sequência Ctrl+C. Para analisar os pacotes capturados, utilize o programa Wireshark previamente instalado. Para abri-lo, inicie um terminal na máquina virtual e utilize o comando:

```
$ sudo wireshark
```

Clique no botão ‘OK’ em ambas as janelas de aviso que aparecerem no canto superior esquerdo da tela. Essa é a tela inicial:

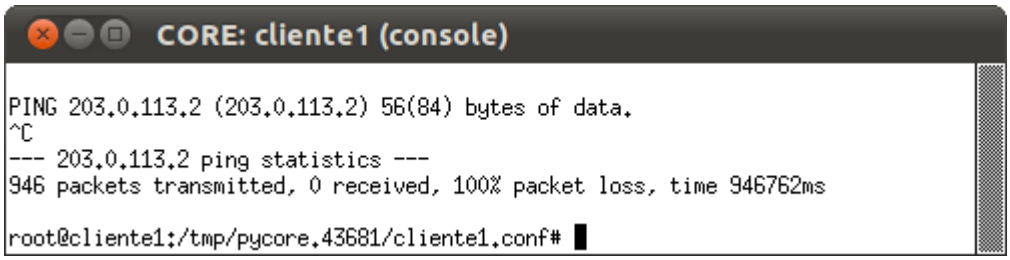

The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply
WIRESHARK
The World's Most Popular Network Protocol Analyzer
Capture
Interface List
Live list of the capture interfaces (counts incoming packets)
Start capture on interface:
eth0
n1.eth0.11
USB bus number 1
n2.eth0.11
n2.eth1.11
USB bus number 2
n3.eth0.11
n3.eth1.11
Capture Options
Start a capture with detailed options
Capture Help
How to Capture
Step by step to a successful capture setup
Files
Open
Open a previously captured file
Open Recent:
Sample Captures
A rich assortment of example capture files on the wiki
Website
User's Guid
Security
Ready to load or capture
No Packets
Profile: Default

Para analisar o arquivo de captura criado, abra o arquivo /tmp/captura_dsl.pcap com o menu File>Open:



Verifique os pacotes capturados e note que pela interface capturada passam pacotes ICMPv4 da origem 10.0.0.2 para o destino 203.0.113.2 encapsulados com cabeçalhos IPv6.

Volte ao CORE e pare a execução do *ping* pressionando Ctrl+C na janela do terminal da máquina 'cliente1'. Observe que que de fato houve 100% de perda de pacotes em razão da ponta de saída do túnel ainda não ter sido configurada:



8. Com o túnel 4in6 funcionando por parte do CPE, o passo seguinte consiste na configuração da máquina AFTR que fechará o túnel e realizará a função de NAT. Abra o terminal da máquina aftr e siga os passos:

b. No terminal dessa mesma máquina, utilize o comando "`nano aftr-script`" para cria um arquivo chamado "aftr-script". Com o editor de texto *nano* aberto, copie o conteúdo conteúdo abaixo e cole com a utilização do atalho Shift+Insert. Mas, caso esteja utilizando a Máquina Virtual fornecida, basta copiar o arquivo previamente criado para a pasta base do 'aftr' com a utilização do comando "`cp /home/core/Desktop/transicao/DS-Lite/aftr-script .`"

```
#!/bin/sh

aftr_start() {
    set -x
    ip link set tun0 up
    ip addr add 192.0.0.1 peer 192.0.0.2 dev tun0
    ip route add 203.0.113.131/32 dev tun0
    ip -6 addr add fe80::1 dev tun0
    ip -6 route add 2001:db8:0:2000::/64 dev tun0
    arp -i eth0 -s 203.0.113.131 0a:0b:0c:0d:0e:f0 pub
}

aftr_stop() {
    set -x
    ip link set tun0 down
}

case "$1" in
start)
       aftr_start
       ;;
stop)
       aftr_stop
       ;;
*)
       echo "Usage: $0 start|stop"
       exit 1
       ;;
esac

exit 0
```

Depois de criado o arquivo, aperte Ctrl+X para sair do nano e 'Y' para confirmar a criação do arquivo.

Após criado o arquivo, utilize o comando 'chmod +x aftr-script' para habilitar a execução do arquivo.

Nesse arquivo é importante observar que:

- os endereços 192.0.0.1 e 192.0.0.2 são especificamente designados pela **RFC 6333** para a configuração das interfaces de túnel nas implementações do DS-Lite;
- 203.0.113.131 é o endereço público utilizado para a realização do NAT na rede interna do ISP. E, deve ser diferente do endereço utilizado na interface física do servidor AFTR;

- 2001:db8:0:2000:: é um endereço arbitrariamente escolhido para fechar o túnel no servidor AFTR. Esse endereço não pode ser utilizado nas interfaces de rede do servidor ou por qualquer outro equipamento na rede.

c. Crie um arquivo chamado "aftr.conf" com o conteúdo a seguir. Caso esteja utilizando a VM fornecida, basta copiar o arquivo previamente criado para a pasta base do 'aftr' com a utilização do comando '`cp /home/core/Desktop/transicao/DS-Lite/aftr.conf1 ./aftr.conf`' no terminal dessa mesma máquina.

```
default tunnel mss on
defmtu 1450
address endpoint 2001:db8:0:2000::
address icmp 203.0.113.131
pool 203.0.113.131
acl6 ::0/0
```

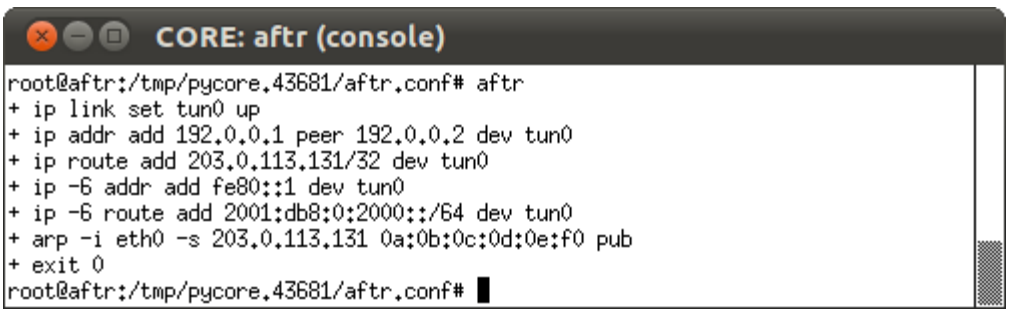
Novamente:

- 203.0.113.131 é o endereço público utilizado para a realização do NAT na rede interna do ISP. E, deve ser diferente do endereço utilizado na interface física do servidor AFTR;
- 2001:db8:0:2000:: é um endereço arbitrariamente escolhido para fechar o túnel no servidor AFTR. Esse endereço não pode ser utilizado nas interfaces de rede do servidor ou por qualquer outro equipamento na rede.

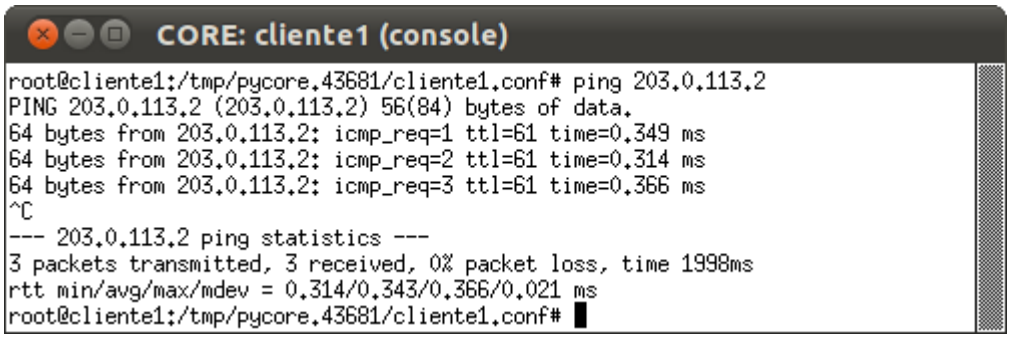
d. Inicie o serviço aftr:

```
# aftr
```

A saída deve ser:



9. Com isso, implantação da técnica de está finalizada. Para testar seu funcionamento, utilize o comando "*ping 203.0.113.2*" a partir da máquina cliente 'cliente1' para verificar que ela possui conectividade IPv4 com a Internet:



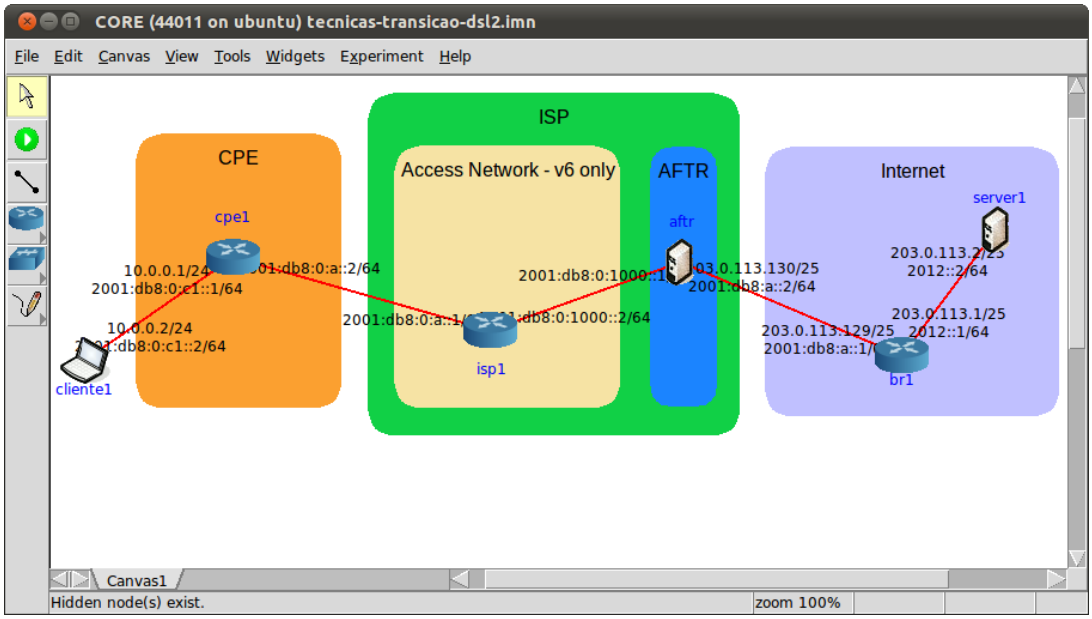
Se desejar, utilize o Wireshark para analisar os diversos equipamentos da rede simulada para verificar como o pacote "*ping*" é encapsulado e desencapsulado em seu trajeto.

Ao finalizar os testes, acesse a máquina 'aftr' e utilize o comando 'killall aftr' para matar o processo 'aftr' que foi iniciado. E, pare a simulação clicando no botão  .

Experiência 4 - Adição de novos clientes na rede do ISP

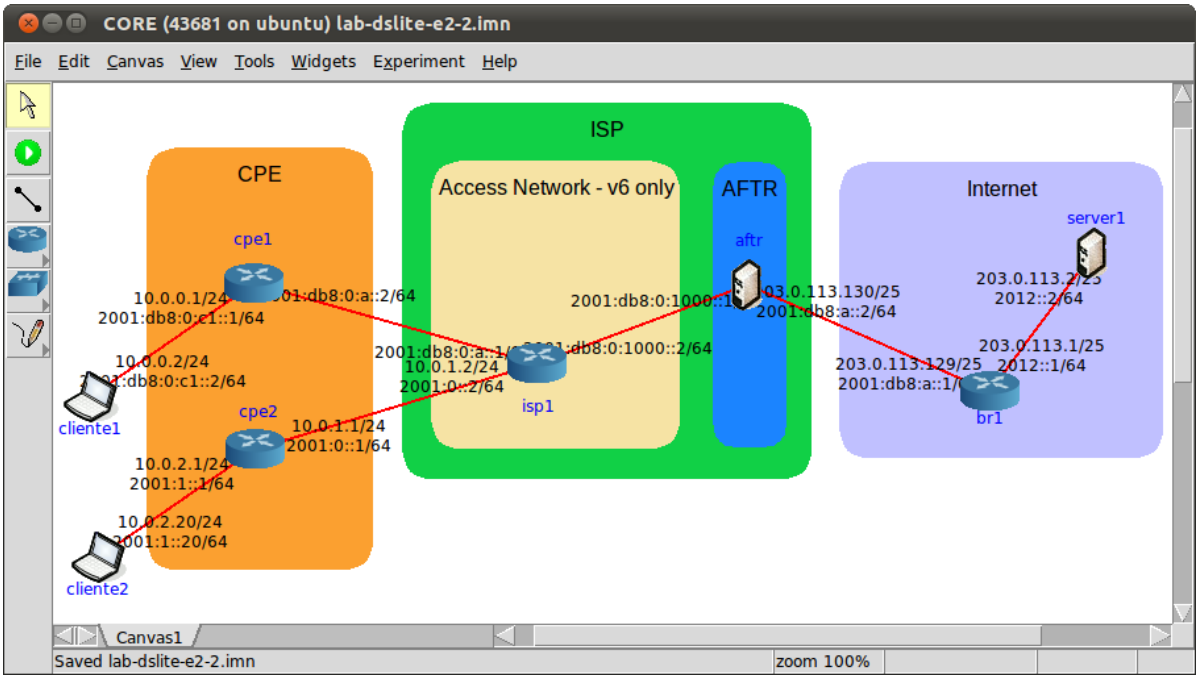
1. O objetivo dessa segunda experiência é observar o comportamento do sistema AFTR com a adição de novos clientes à rede do ISP e, com isso, mostrar que a solução pode ser expandida sem muitas dificuldades.

2. Nela, considera-se que o aluno já tenha realizado a experiência inicial da técnica DS-Lite. O arquivo "**tecnicas-transicao-dsl2.imn**" será utilizado e está pré-configurado com o túnel da experiência anterior.
   Ao abrir o arquivo, a seguinte topologia deve aparecer:



   Para testá-la, clique no botão  para iniciar a simulação e siga as instruções do item 9 da primeira experiência.

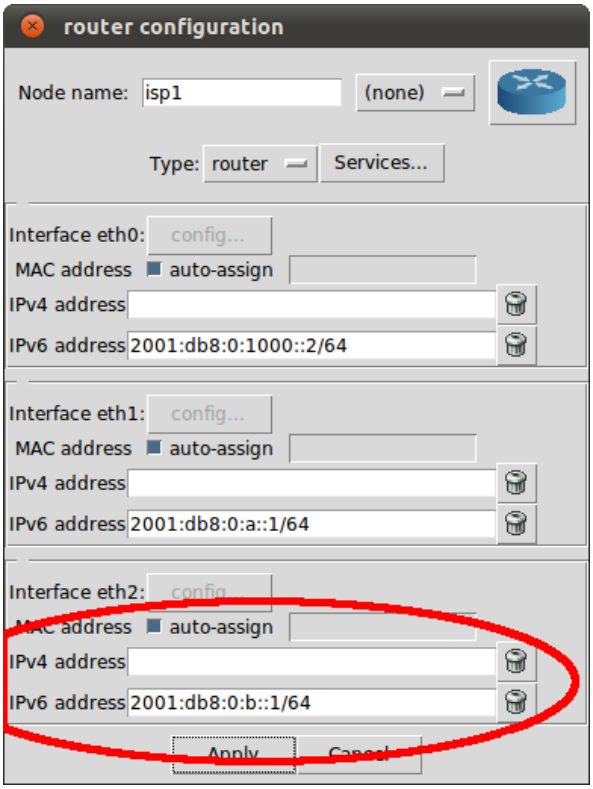
3. O primeiro passo é adicionar um novo cliente à essa rede.
   a. Volte ao modo de edição da rede, pare a execução da simulação clicando no botão  ;
   b. Clique no botão "network-layer virtual nodes"  , selecione a opção "*router*"  para adicionar um novos roteadores, clique na área de desenho para adicionar um à topologia;
   c. Clique novamente no botão "network-layer virtual nodes", selecione a opção "PC"  e clique na área de desenho para adicioná-lo;
   d. Utilize a ferramenta "link tool"  para ligar os equipamentos da seguinte forma:



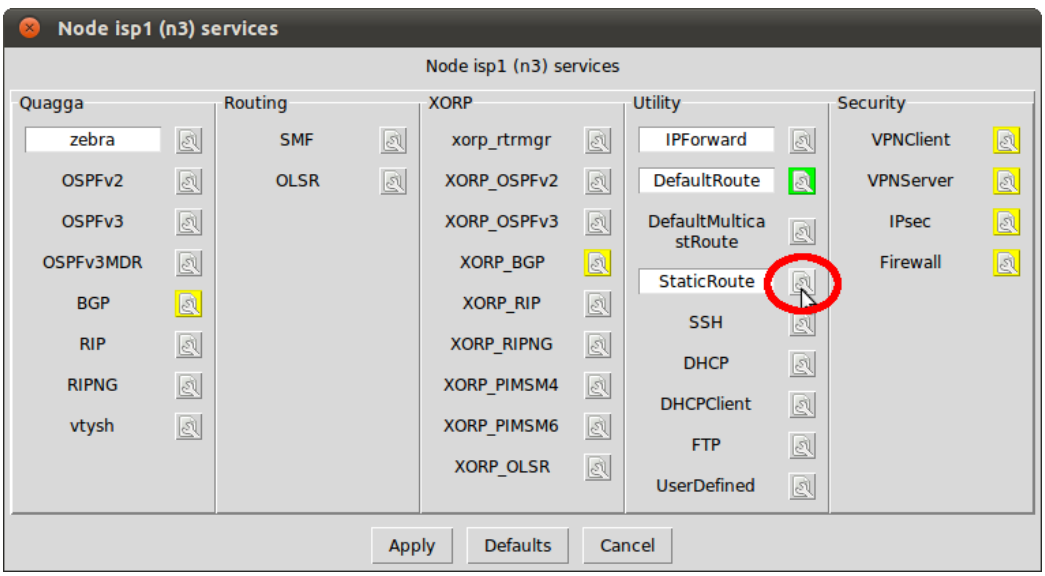
Importante: para que os comandos que serão passados funcionem da forma correta, crie inicialmente o link entre os roteadores 'cpe2' e 'isp1' para que o CORE utilize o nome *eth0* nessa interface roteador 'cpe2'.

4. A seguir deve-se configurar essa nova sub-rede:

   a. Utilize a ferramenta “selection tool”  para abrir a janela de configurações do roteador 'isp1' com um duplo clique sobre ele. Nessa janela, edite a interface *eth2* de forma que o campo “*IPv4 address*” fique vazio e o campo “IPv6 address” contenha o endereço “2001:db8:0:b::1/64”:

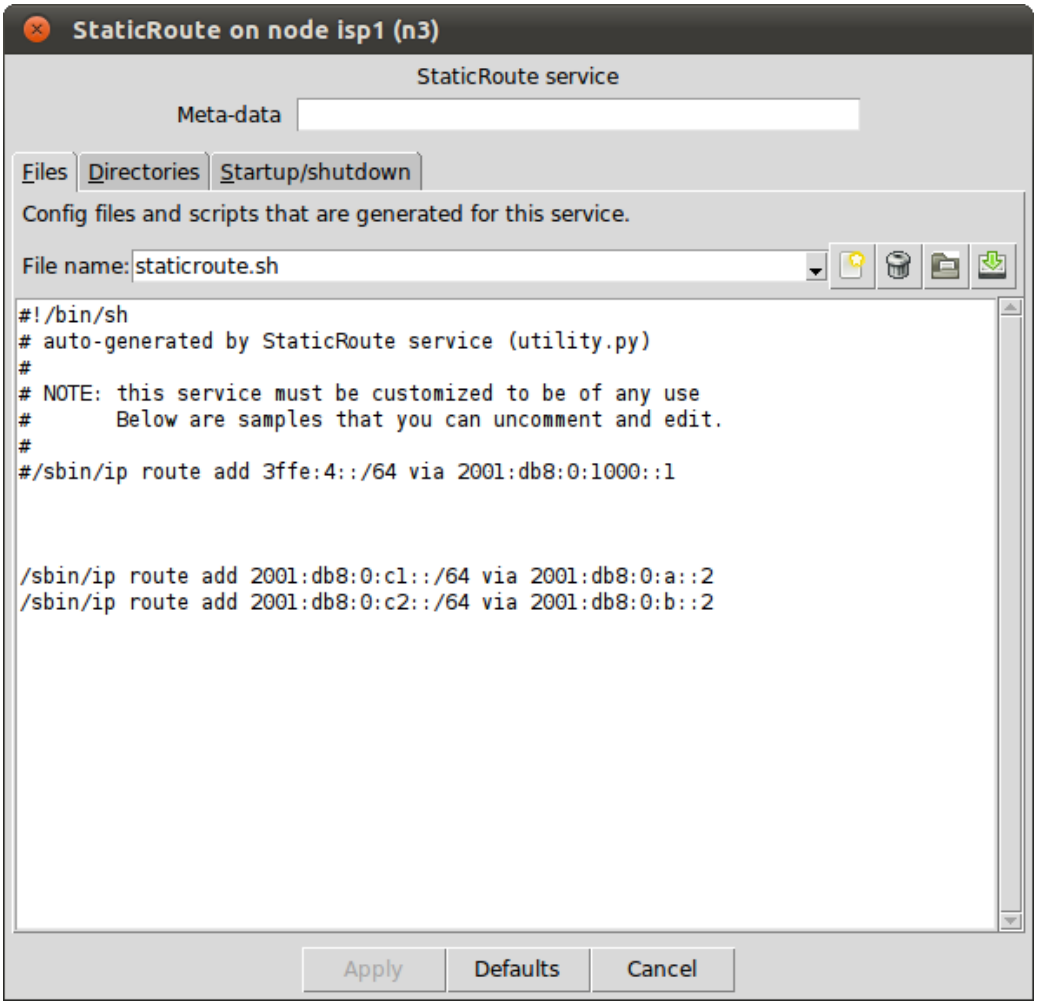


b. Ainda nessa janela, clique no botão serviços ("Services...") para configurar as novas rotas desse roteador. Na janela que abrir clique no botão de configuração de rotas estáticas como indicado na figura:
   - Obs: Numa implementação mais realista da rede de um ISP, as configurações de rotas poderiam ser feitas de forma automática com a utilização de protocolos como RIP ou OSPF, porém, nessa experiência elas serão configuradas com rotas estáticas.

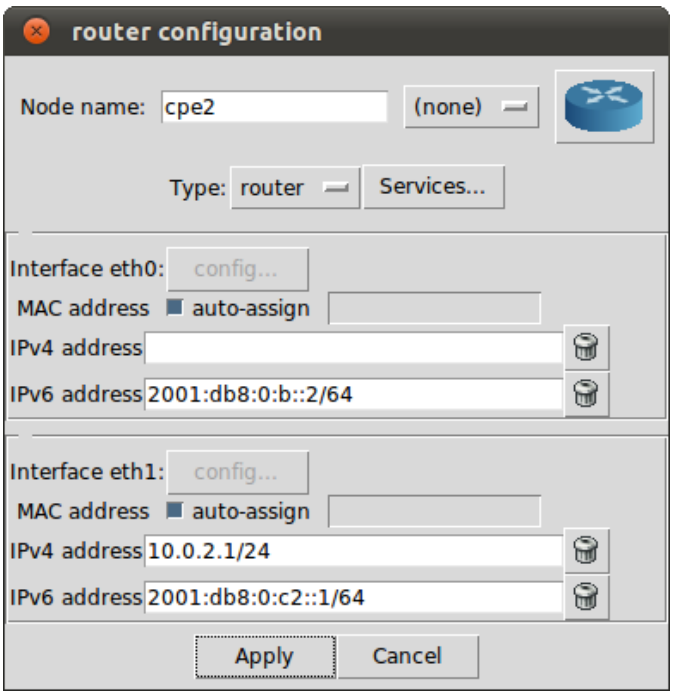


Adicione a seguinte linha arquivo e confirme (botão "Apply") a configuração em todas as janelas:

```
/sbin/ip route add 2001:db8:0:c2::/64 via 2001:db8:0:b::2
```



c. No roteador 'cpe2' configure o endereço “2001:db8:0:b::2/64” na interface eth0 e os endereços “10.0.2.1/24” e “2001:db8:0:c2::1” na interface eth1:

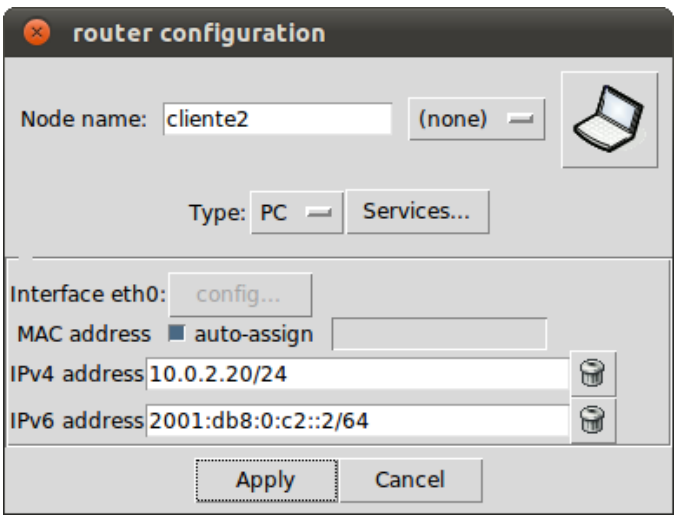


Na janela de serviços ative a opção de rota default clicando em “Default Route” e nas configurações, substitua o conteúdo do arquivo por:

```
#!/bin/sh
ip route add default via 2001:db8:0:b::1
```

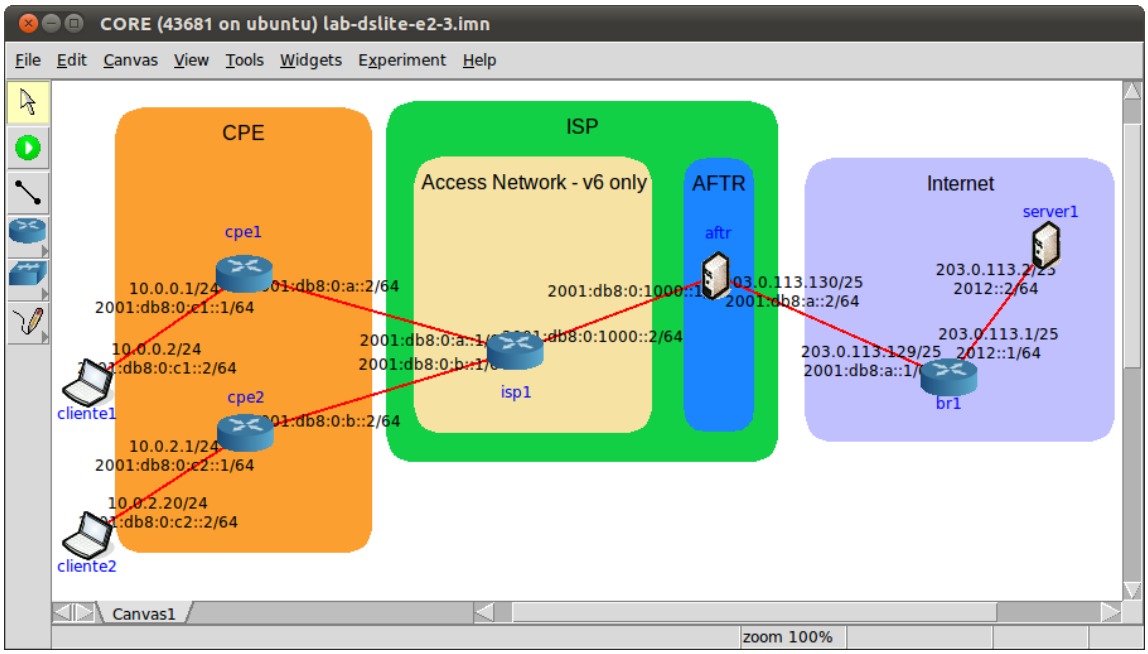
Ao fim, confirme (“Apply”) todas as alterações.

d. E, no PC 'cliente2' configure os seguintes endereços:



Obs: apesar de nessa simulação estarmos configurando a rede IPv4 do cliente com endereços estáticos, numa situação real esses endereços normalmente seriam configurados com a utilização de um serviço DHCPv4 no CPE do cliente.

Ao fim a rede deve ficar assim:



5. Com a rede IPv6 configurada, simulação deve ser iniciada para que se possa prosseguir com a configuração DS-Lite e prover conectividade IPv4 ao novo cliente 'cliente2'.
    a. Antes de iniciar configuração do túnel verifique, com a utilização dos comandos '*ping6 2012::2*' e '*ping 203.0.133.2*' no terminal do computador do novo cliente, que existe conectividade IPv6 com a Internet mas não IPv4.
    Os resultados devem ser os senguites:

```
CORE: cliente2 (console)
root@cliente2:/tmp/pycore.43681/cliente2.conf# ping6 2012::2
PING 2012::2(2012::2) 56 data bytes
64 bytes from 2012::2: icmp_seq=1 ttl=60 time=36.0 ms
64 bytes from 2012::2: icmp_seq=2 ttl=60 time=0.207 ms
64 bytes from 2012::2: icmp_seq=3 ttl=60 time=0.165 ms
64 bytes from 2012::2: icmp_seq=4 ttl=60 time=0.178 ms
^C
--- 2012::2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.165/9.151/36.055/15.533 ms
root@cliente2:/tmp/pycore.43681/cliente2.conf#
```

```
CORE: cliente2 (console)
root@cliente2:/tmp/pycore.43681/cliente2.conf# ping 203.0.113.2
PING 203.0.113.2 (203.0.113.2) 56(84) bytes of data.
From 10.0.2.1 icmp_seq=1 Destination Net Unreachable
From 10.0.2.1 icmp_seq=2 Destination Net Unreachable
From 10.0.2.1 icmp_seq=3 Destination Net Unreachable
From 10.0.2.1 icmp_seq=4 Destination Net Unreachable
^C
--- 203.0.113.2 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 2998ms

root@cliente2:/tmp/pycore.43681/cliente2.conf#
```

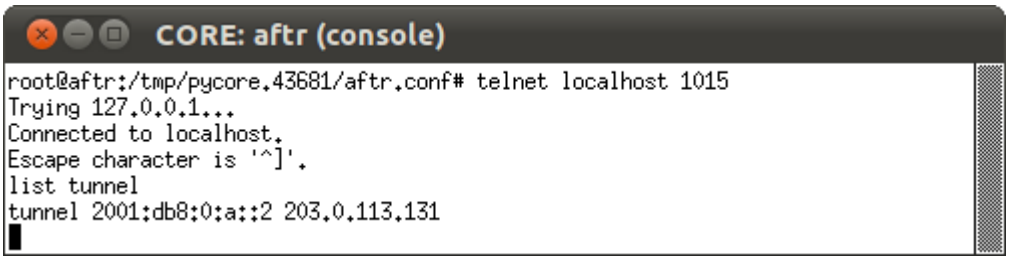
Faça o mesmo no cliente anterior 'cliente1' para verificar que ele ainda possui

conectividade IPv4 e IPv6.

b. Abra o terminal do host 'aftr' que provê o serviço AFTR e, nele digite o seguinte comando para abrir o shell do programa aftr:

```
# telnet localhost 1015
```

Depois de se conectar à interface do serviço AFTR utilize o comando '*list tunnel*' para listar os túneis que já estão ativos. O resultado deve ser o seguinte:



O que mostra que existe apenas o túnel para o 'cpe1' do cliente previamente configurado.

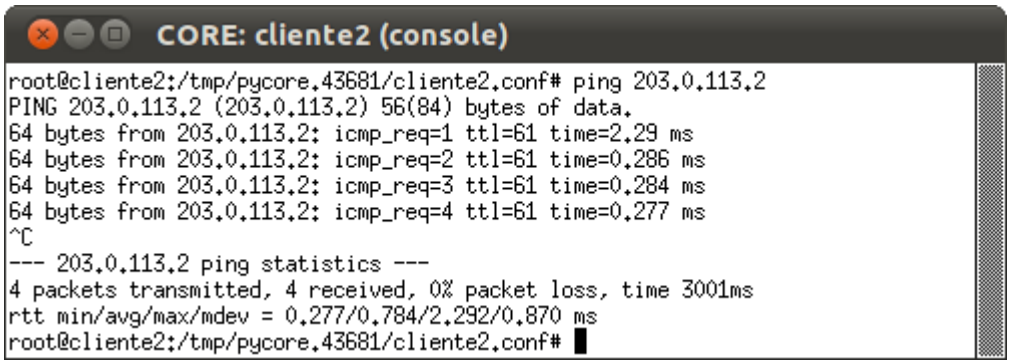
Importante: O túnel pode não estar listado por ser criado dinamicamente. Caso isso aconteça, utilize o comando "ping 203.0.113.2" no terminal da máquina 'cliente1' para reestabelecer o túnel.

c. Sem fechar a janela do terminal 'aftr', acesse o terminal do roteador 'cpe2'. Utilize os seguintes comandos para configurar o novo túnel 4in6:

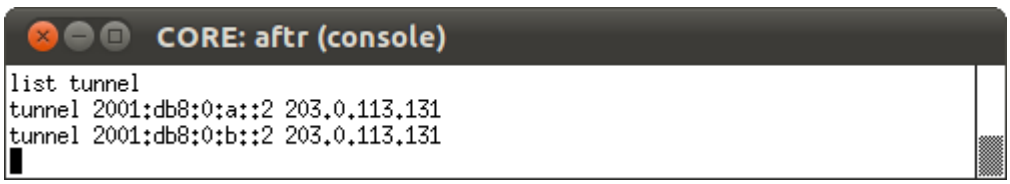
```
# modprobe ip6_tunnel
# ip -6 tunnel add dsltun mode ipip6 remote 2001:db8:0:2000:: local
2001:db8:0:b::2 dev eth0
# ip addr add 192.0.0.2 peer 192.0.0.1 dev dsltun
# ip link set dev dsltun up
# ip route add default dev dsltun
```

Note que todos os endereços a exceção do endereço da interface local do roteador e da ponta local do túnel são os mesmos configurados no CPE da experiência anterior.

6. Com isso, o túnel já deve estar configurado e funcionando. Repita o comando '*ping 203.0.113.2*' para verificar que o cliente 'cliente2' passou a ter conectividade IPv4 com a Internet. O Resultado deve parecido com:



No terminal do AFTR, utilize o comando '*list tunnel*' para verificar que um novo túnel 4in6 foi automaticamente criado:

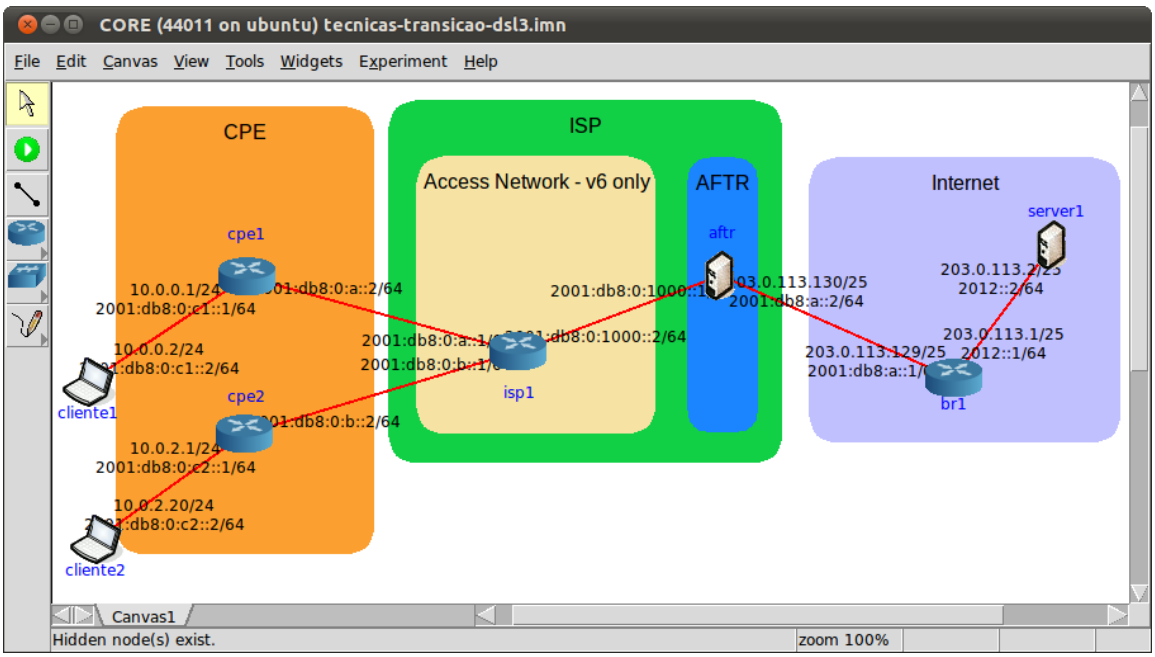


Se desejar, utilize o Wireshark para escutar algumas interfaces da rede e verificar como é o encapsulamento dos pacotes no túnel. Depois que acabar os testes, termine a simulação clicando no botão  .

Experiência 5 - Adição da técnica A+P à rede DS-Lite

1. Nessa experiência pretende-se aplicar a técnica A+P /Port Range Router a uma rede previamente configurada com a técnica DS-Lite com o objetivo de mostrar a integração dessas técnicas e de apresentar uma solução que permita o controle de algumas portas IPv4 por parte dos clientes da operadora. Facilitando, assim, a utilização de algumas aplicações como servidores Web.
2. Para essa terceira experiência considera-se que o aluno já tenha realizado as experiências anteriores da técnica DS-Lite. O arquivo "tecnicas-transicao-dsl3.imn" será utilizado por já estar configurado com os túneis da segunda experiência apresentada.

   Ao abrir o arquivo, a seguinte rede deve aparecer:



3. Na experiência, a máquina "aftr", "cpe1" e "cpe2" serão configuradas com a técnica A+P com o intuito de permitir o compartilhamento de um mesmo IPv4 público por mais de um cliente do ISP.

   Para começar, inicie a simulação da rede com o botão  .

4. Com a simulação em andamento, a máquina "aftr" terá de ser reconfigurada com as configurações da técnica A+P:
   a. Acesse o terminal da máquina "aftr" e edite o arquivo aftr-script utilizando o comando "*nano aftr-script*". Nele, adicione as linhas abaixo antes do final da função "aftr_start()" para fazer com que esse servidor responda pelo endereço IPv4 203.0.113.133 que será distribuído pra os clientes:

```
ip route add 203.0.113.133/32 dev tun0
arp -i eth0 -s 203.0.113.133 0a:0b:0c:0d:0e:f2 pub
```

Feche o *nano* utilizando o atalho Ctrl+X.

b. Ainda no terminal da máquina “aftr”, edite o arquivo aftr.conf com o comando “*nano aftr.conf*”. Troque o conteúdo desse arquivo pela configuração a seguir. Caso esteja utilizando a VM fornecida, basta sobrescrever o arquivo com o comando ‘`cp /home/core/Desktop/transicao/DS-Lite/aftr.conf2 ./aftr.conf`’ no terminal dessa mesma máquina.

```
autotunnel off
bucket tcp size 11
bucket udp size 9
bucket icmp size 4
decay 1 .983
decay 5 .996
decay 15 .998
default fragment equal off
default fragment lifetime 31
default fragment ipv6 maxcount 1025
default fragment in maxcount 1026
default fragment out maxcount 1027
default hold lifetime 121
default nat lifetime tcp 601
default nat lifetime closed 121
default nat lifetime udp 301
default nat lifetime icmp 31
default nat lifetime retrans 11
default pool tcp 10-9998
default pool udp 11-9999
default pool echo 12-10000
default tunnel auto on
default tunnel mss on
default tunnel mtu 1310
default tunnel toobig off
default tunnel toobig strict
default tunnel fragment ipv6 maxcount 17
default tunnel fragment ipv4 maxcount 65
default tunnel nat tcp maxcount 201
default tunnel nat udp maxcount 202
default tunnel nat icmp maxcount 51
default tunnel nat tcp rate 51
default tunnel nat udp rate 21
default tunnel nat icmp rate 6
defmss off
defmtu 1320
deftoobig off
deftoobig strict
eqfrag on
eqfrag off
quantum 21

default tunnel mss on
defmtu 1450
address endpoint 2001:db8:0:2000::
```

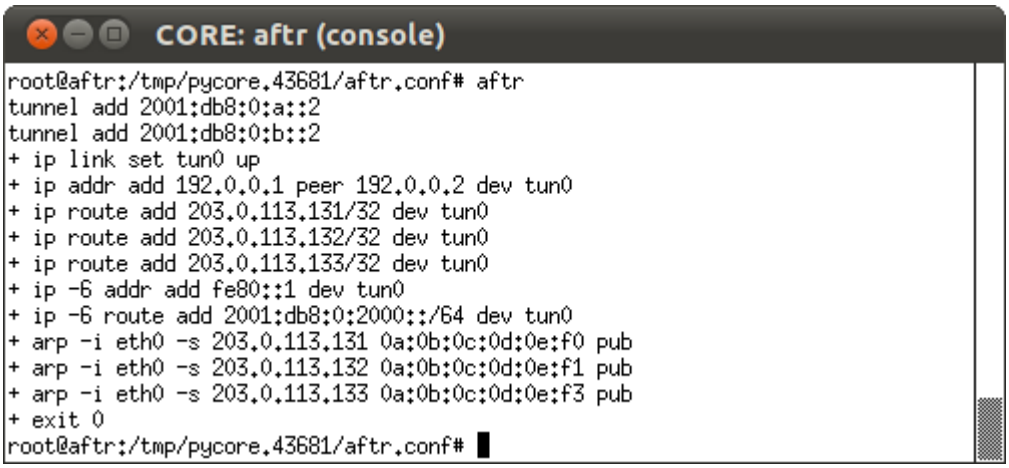
```
address icmp 203.0.113.131
pool 203.0.113.131
pool 203.0.113.133 echo 32000-64000


acl6 ::0/0


prr 2001:db8:0:a::2 tcp 203.0.113.133 10000
prr 2001:db8:0:a::2 udp 203.0.113.133 10000
prr 2001:db8:0:a::2 tcp 203.0.113.133 10001
prr 2001:db8:0:a::2 udp 203.0.113.133 10001
prr 2001:db8:0:a::2 tcp 203.0.113.133 10002
prr 2001:db8:0:a::2 udp 203.0.113.133 10002
prr 2001:db8:0:a::2 tcp 203.0.113.133 10003
prr 2001:db8:0:a::2 udp 203.0.113.133 10003

prr 2001:db8:0:b::2 tcp 203.0.113.133 10010
prr 2001:db8:0:b::2 udp 203.0.113.133 10010
prr 2001:db8:0:b::2 tcp 203.0.113.133 10011
prr 2001:db8:0:b::2 udp 203.0.113.133 10011
prr 2001:db8:0:b::2 tcp 203.0.113.133 10012
prr 2001:db8:0:b::2 udp 203.0.113.133 10012
prr 2001:db8:0:b::2 tcp 203.0.113.133 10013
prr 2001:db8:0:b::2 udp 203.0.113.133 10013
```

Com a linha “pool 203.0.113.133 echo 32000-64000” esse arquivo configura o endereço 203.0.113.133 como parte do pool de endereços que serão utilizados para fazer o NAT da rede interna do ISP. E, as linhas “prr” designam as portas TCP e UDP entre 10000 e 10003 para serem utilizadas pelo cliente 1 e as portas de 10010 a 10013 para o cliente 2.

Note que cada uma das portas deve ser definida individualmente. Porém, como alternativa para que essa configuração estática não fique extremamente grande, as portas podem ser distribuídas dinamicamente com a utilização da interface *shell* da aplicação AFTR acessível via telnet porta 1015.

Ao concluir a configuração do arquivo, saia e salve-o com o comando Ctrl+X.

c. Para finalizar a configuração da máquina “aftr” utilize o comando “aftr” no mesmo terminal que estava sendo utilizado. A saída deve ser:



5. O passo seguinte da implantação da técnica A+P é configuração das máquinas “cpe1” e “cpe2”. Agora, além de ficarem encarregadas da criação dos túneis IPv4 em IPv6, elas também serão responsáveis pela tradução dos endereços IPv4 públicos providos pelo ISP para os endereços do NAT local da rede do cliente.
Nessa etapa, o programa *iptables* é utilizado para desempenhar tal tradução. Abra o terminal correspondente à máquina “cpe1” e utilize os comandos a seguir. Mas, caso esteja utilizando a VM fornecida, execute o script ‘`iptables-cpe1.sh`’ com o comando ‘sh /home/core/Desktop/transicao/DS-Lite/iptables-cpe1.sh’.

```
# iptables -t nat -A POSTROUTING -p tcp -s 10.0.0.0/24 -j SNAT --to-source
203.0.113.133:10001-10003
# iptables -t nat -A POSTROUTING -p udp -s 10.0.0.0/24 -j SNAT --to-source
203.0.113.133:10001-10003
# iptables -t nat -A POSTROUTING -p icmp -s 10.0.0.0/24 -j SNAT
--to-source 203.0.113.133:10001-10003

# iptables -t nat -A PREROUTING -p tcp -d 203.0.113.133 --dport 10000 -j DNAT
--to-destination 10.0.0.2:22
# iptables -t nat -A PREROUTING -p udp -d 203.0.113.133 --dport 10000 -j DNAT
--to-destination 10.0.0.2:22
# iptables -t nat -A PREROUTING -p icmp -d 203.0.113.133 -j DNAT
--to-destination 10.0.0.2
```

Isso serve para configurar o *iptables* para mapear os pacotes TCP e UDP que chegam pela porta de 10000 na porta 22 correspontente ao serviço de ssh do “cliente1” e, também, configura os pacotes que saem da rede interna para utilizarem as portas de 10001 a 10003 que foram configuradas no AFTR para formar o túnel com este CPE.
Os comandos utilizados no “cpe2” são equivalentes, sendo a porta 10010 mapeada para o servidor ssh do cliente 2 e as portas de 10011 a 10012 como portas de saída. Caso esteja utilizando a VM fornecida, execute o script ‘`iptables-cpe2.sh`’ com a linha ‘`sh /home/core/Desktop/transicao/DS-Lite/iptables-cpe1.sh`’. Caso

contrário, execute os seguites comandos:

```
# iptables -t nat -A POSTROUTING -p tcp -s 10.0.2.0/24 -j SNAT --to-source
203.0.113.133:10011-10013
# iptables -t nat -A POSTROUTING -p udp -s 10.0.2.0/24 -j SNAT --to-source
203.0.113.133:10011-10013
# iptables -t nat -A POSTROUTING -p icmp -s 10.0.2.0/24 -j SNAT
--to-source 203.0.113.133:10011-10013

# iptables -t nat -A PREROUTING -p tcp -d 203.0.113.133 --dport 10010 -j DNAT
--to-destination 10.0.2.20:22
# iptables -t nat -A PREROUTING -p udp -d 203.0.113.133 --dport 10010 -j DNAT
--to-destination 10.0.2.20:22
# iptables -t nat -A PREROUTING -p icmp -d 203.0.113.133 -j DNAT
--to-destination 10.0.2.20
```

6. Para testar o funcionamento da técnica, abra o terminal da máquina “cliente1” e execute o comando abaixo para iniciar uma captura de pacotes com o programa *tcpdump*:

```
# tcpdump -vv -i eth0 -s 0 -w /tmp/captura_a+p1.pcap
```

Sem fechá-lo, abra o terminal do “cliente2” e digite:

```
# tcpdump -vv -i eth0 -s 0 -w /tmp/captura_a+p2.pcap
```

Por fim, inicie também um *tcpdump* no terminal da máquina server1:

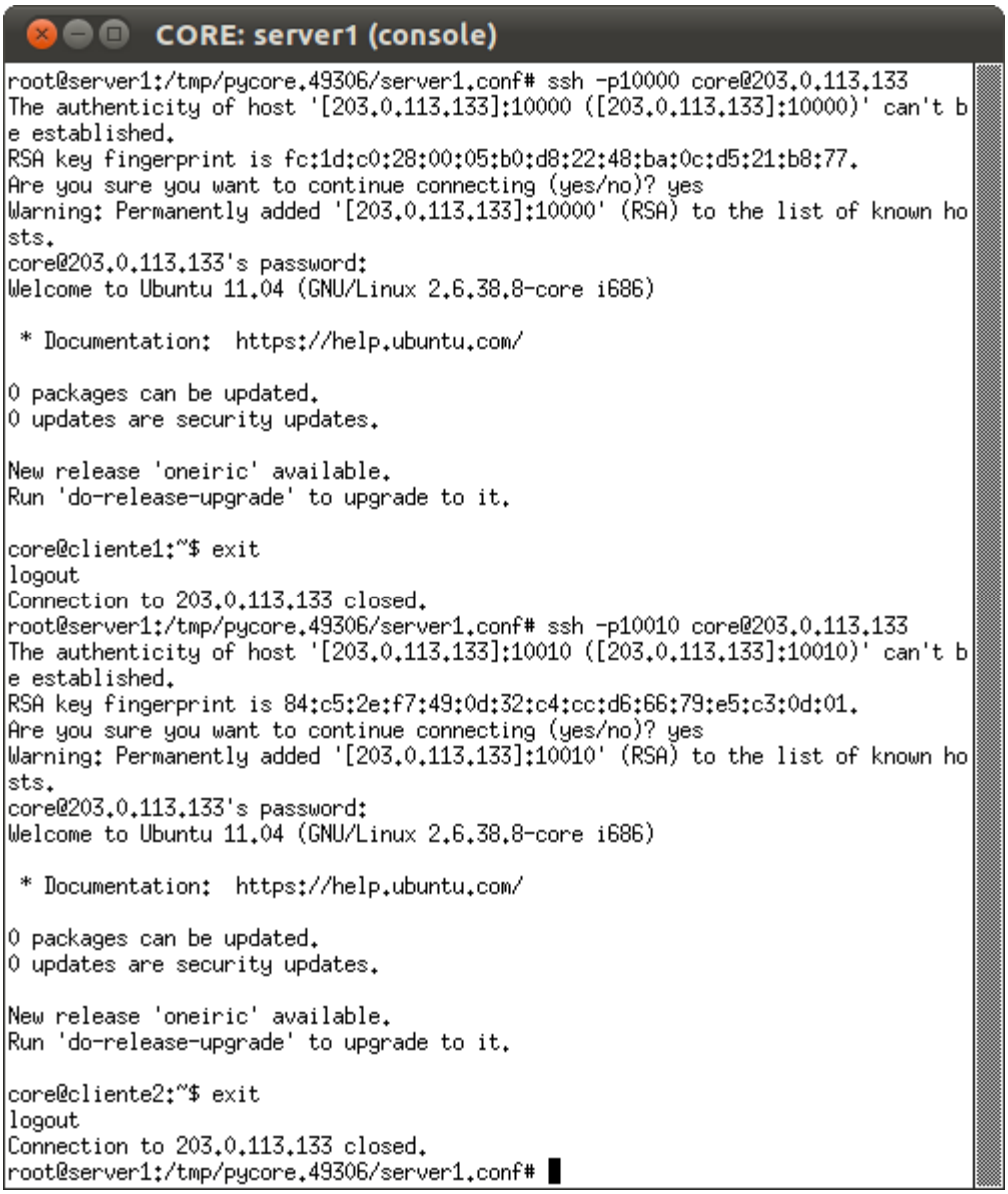
```
# tcpdump -vv -i eth0 -s 0 -w /tmp/captura_server1.pcap
```

Abra um novo terminal da máquina “server1” e digite os comandos abaixo para solicitar conexões SSH nas portas 10000 e 10010 para o endereço 203.0.113.133. A seguir, volte aos terminais das máquinas “cliente1” e “cliente2” e pare as execuções dos *tcpdumps* com o comando Ctrl+C.

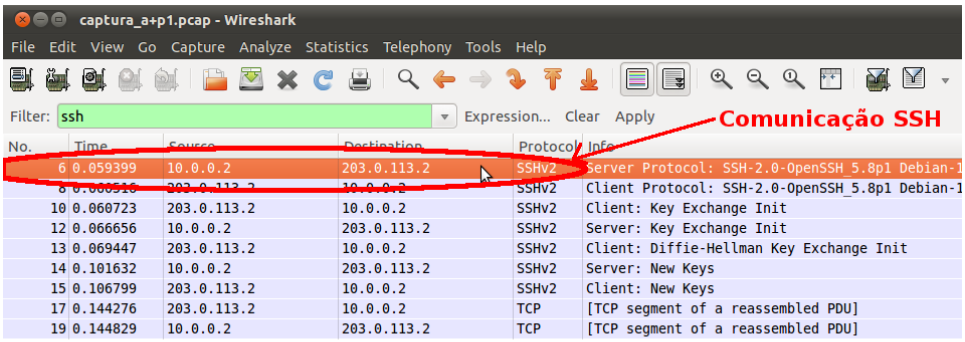
```
# ssh -p10000 core@203.0.113.133
# exit
# ssh -p10010 core@203.0.113.133
# exit
```

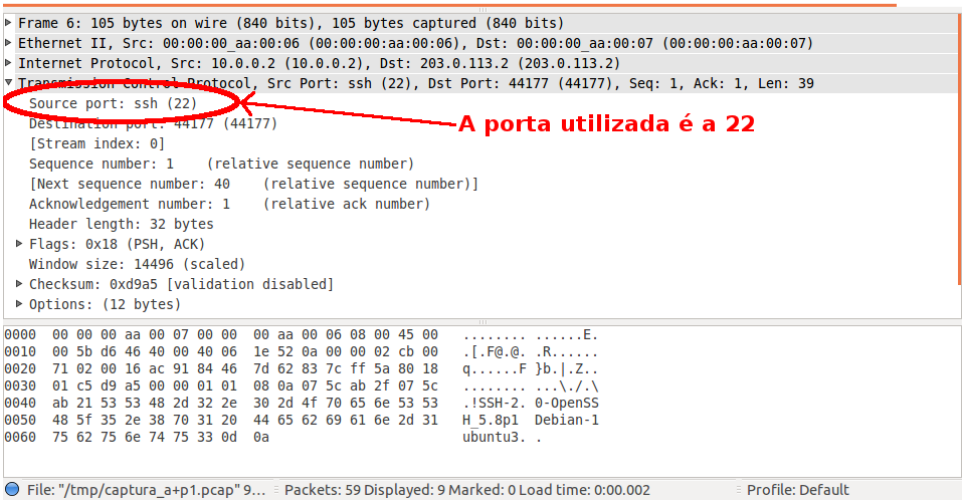
Importante: Depois de cada chamada do comando *ssh* será requisitada a aceitação da chave publica do servidor e a senha do usuário. Aceite a chave, digitando ‘yes’ e, depois, ‘core’ quando a senha for solicitada.

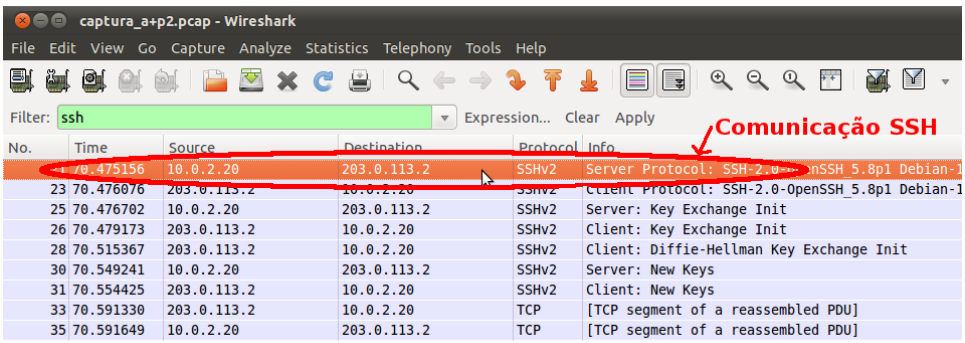
O resultado deve ser:

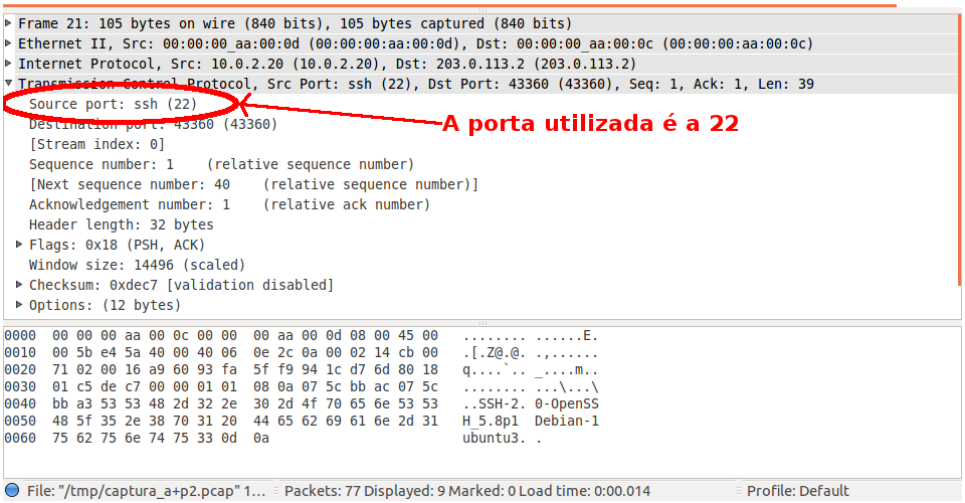


Isso mostra que os clientes 1 e 2 aceitaram os pedidos de conexão SSH na porta 22. Para verificar que as requisições de fato chegaram aos clientes, abra os arquivos "/tmp/captura_a+p1.pcap" e "/tmp/captura_a+p2.pcap" no Wireshark. E note que a pesar de a conexão ter sido feita com as portas 10000 e 10010 do IP 203.0.113.133 elas chegaram corretamente às portas dos servidores SSH configurados em cada um dos clientes. E, para melhor visualizar a conversa no Wireshark, digite "SSH" na barra de Filtros ("*Filter:*"):

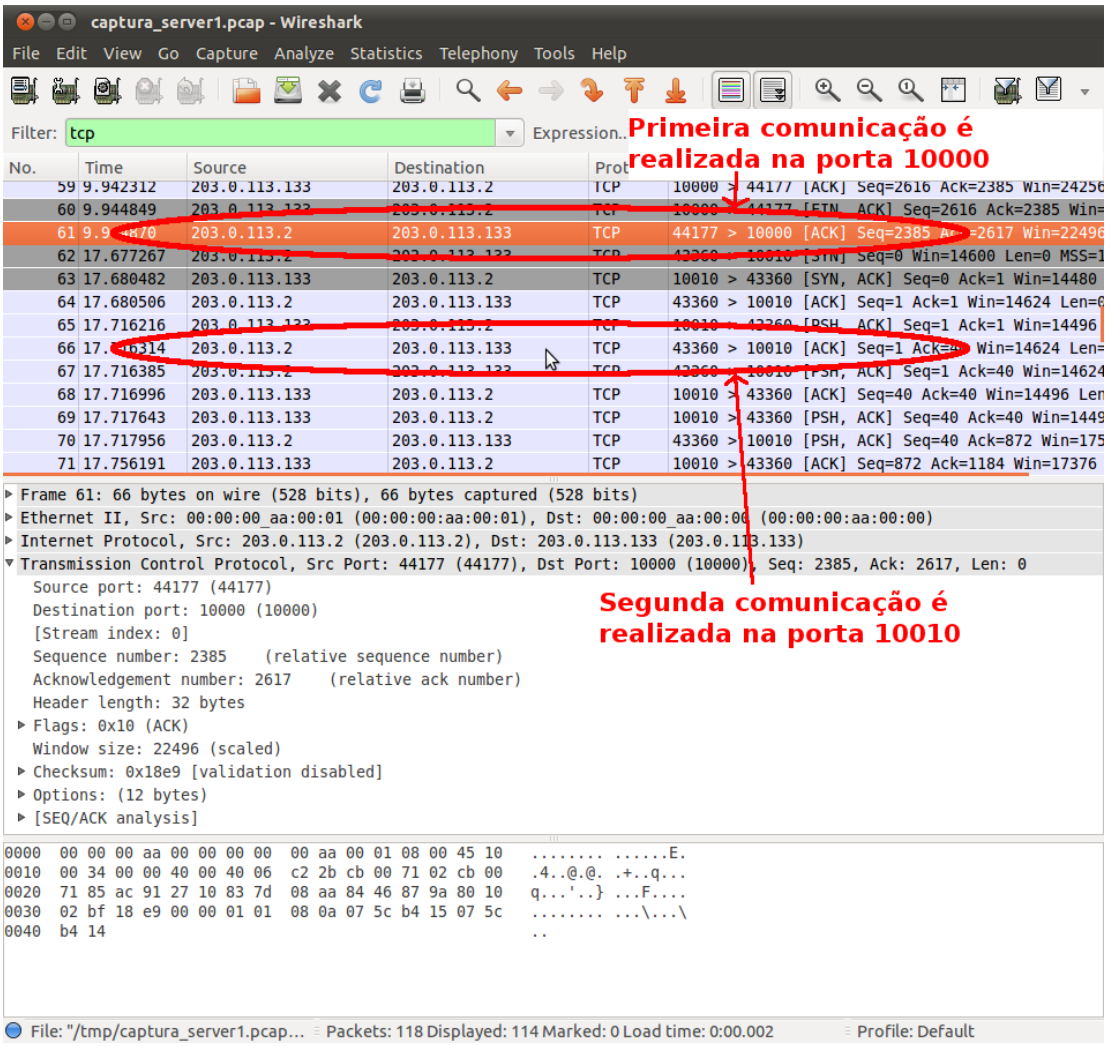

captura_a+p1.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter: ssh
Expression... Clear Apply
Comunicação SSH
No. Time Source Destination Protocol Info
6 0.059399 10.0.0.2 203.0.113.2 SSHv2 Server Protocol: SSH-2.0-OpenSSH_5.8p1 Debian-1
8 0.060510 203.0.113.2 10.0.0.2 SSHv2 Client Protocol: SSH-2.0-OpenSSH_5.8p1 Debian-1
10 0.060723 203.0.113.2 10.0.0.2 SSHv2 Client: Key Exchange Init
12 0.066656 10.0.0.2 203.0.113.2 SSHv2 Server: Key Exchange Init
13 0.069447 203.0.113.2 10.0.0.2 SSHv2 Client: Diffie-Hellman Key Exchange Init
14 0.101632 10.0.0.2 203.0.113.2 SSHv2 Server: New Keys
15 0.106799 203.0.113.2 10.0.0.2 SSHv2 Client: New Keys
17 0.144276 203.0.113.2 10.0.0.2 TCP [TCP segment of a reassembled PDU]
19 0.144829 10.0.0.2 203.0.113.2 TCP [TCP segment of a reassembled PDU]


Frame 6: 105 bytes on wire (840 bits), 105 bytes captured (840 bits)
Ethernet II, Src: 00:00:00_aa:00:06 (00:00:00:aa:00:06), Dst: 00:00:00_aa:00:07 (00:00:00:aa:00:07)
Internet Protocol, Src: 10.0.0.2 (10.0.0.2), Dst: 203.0.113.2 (203.0.113.2)
Transmission Control Protocol, Src Port: ssh (22), Dst Port: 44177 (44177), Seq: 1, Ack: 1, Len: 39
Source port: ssh (22)
Destination port: 44177 (44177)
A porta utilizada é a 22
[Stream index: 0]
Sequence number: 1 (relative sequence number)
[Next sequence number: 40 (relative sequence number)]
Acknowledgement number: 1 (relative ack number)
Header length: 32 bytes
Flags: 0x18 (PSH, ACK)
Window size: 14496 (scaled)
Checksum: 0xd9a5 [validation disabled]
Options: (12 bytes)
0000 00 00 00 aa 00 07 00 00 00 aa 00 06 08 00 45 00 ....E.
0010 00 5b d6 46 40 00 40 06 1e 52 0a 00 00 02 cb 00 .[.F@.@. .R.....
0020 71 02 00 16 ac 91 84 46 7d 62 83 7c ff 5a 80 18 q.....F }b.|.Z..
0030 01 c5 d9 a5 00 00 01 01 08 0a 07 5c ab 2f 07 5c ....\./.\
0040 ab 21 53 53 48 2d 32 2e 30 2d 4f 70 65 6e 53 53 .!SSH-2. 0-OpenSS
0050 48 5f 35 2e 38 70 31 20 44 65 62 69 61 6e 2d 31 H_5.8p1 Debian-1
0060 75 62 75 6e 74 75 33 0d 0a ubuntu3. .
File: "/tmp/captura_a+p1.pcap" 9... Packets: 59 Displayed: 9 Marked: 0 Load time: 0:00.002 Profile: Default


captura_a+p2.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter: ssh
Expression... Clear Apply
Comunicação SSH
No. Time Source Destination Protocol Info
21 70.475156 10.0.2.20 203.0.113.2 SSHv2 Server Protocol: SSH-2.0-OpenSSH_5.8p1 Debian-1
23 70.476076 203.0.113.2 10.0.2.20 SSHv2 Client Protocol: SSH-2.0-OpenSSH_5.8p1 Debian-1
25 70.476702 10.0.2.20 203.0.113.2 SSHv2 Server: Key Exchange Init
26 70.479173 203.0.113.2 10.0.2.20 SSHv2 Client: Key Exchange Init
28 70.515367 203.0.113.2 10.0.2.20 SSHv2 Client: Diffie-Hellman Key Exchange Init
30 70.549241 10.0.2.20 203.0.113.2 SSHv2 Server: New Keys
31 70.554425 203.0.113.2 10.0.2.20 SSHv2 Client: New Keys
33 70.591330 203.0.113.2 10.0.2.20 TCP [TCP segment of a reassembled PDU]
35 70.591649 10.0.2.20 203.0.113.2 TCP [TCP segment of a reassembled PDU]


Frame 21: 105 bytes on wire (840 bits), 105 bytes captured (840 bits)
Ethernet II, Src: 00:00:00_aa:00:0d (00:00:00:aa:00:0d), Dst: 00:00:00_aa:00:0c (00:00:00:aa:00:0c)
Internet Protocol, Src: 10.0.2.20 (10.0.2.20), Dst: 203.0.113.2 (203.0.113.2)
Transmission Control Protocol, Src Port: ssh (22), Dst Port: 43360 (43360), Seq: 1, Ack: 1, Len: 39
Source port: ssh (22)
Destination port: 43360 (43360)
A porta utilizada é a 22
[Stream index: 0]
Sequence number: 1 (relative sequence number)
[Next sequence number: 40 (relative sequence number)]
Acknowledgement number: 1 (relative ack number)
Header length: 32 bytes
Flags: 0x18 (PSH, ACK)
Window size: 14496 (scaled)
Checksum: 0xdec7 [validation disabled]
Options: (12 bytes)
0000 00 00 00 aa 00 0c 00 00 00 aa 00 0d 08 00 45 00 ....E.
0010 00 5b e4 5a 40 00 40 06 0e 2c 0a 00 02 14 cb 00 .[.Z@.@. .,.....
0020 71 02 00 16 a9 60 93 fa 5f f9 94 1c d7 6d 80 18 q....`.. _....m..
0030 01 c5 de c7 00 00 01 01 08 0a 07 5c bb ac 07 5c ....\...\
0040 bb a3 53 53 48 2d 32 2e 30 2d 4f 70 65 6e 53 53 ..SSH-2. 0-OpenSS
0050 48 5f 35 2e 38 70 31 20 44 65 62 69 61 6e 2d 31 H_5.8p1 Debian-1
0060 75 62 75 6e 74 75 33 0d 0a ubuntu3. .
File: "/tmp/captura_a+p2.pcap" 1... Packets: 77 Displayed: 9 Marked: 0 Load time: 0:00.014 Profile: Default

Para visualizar os pacotes na interface do servidor, abra o arquivo "/tmp/captura_server1.pcap". Como as portas utilizadas para a comunição não são padrão do serviço *SSH*, a visualização das conversas fica facilitada se antes de abrir o arquivo a opção View->Name Resolution->Enable for Transport Layer for desabilitada. Ainda assim, o nome do filtro deve ser mudado de "SSH" para "TCP". Note, nesse arquivo, que de fato a comunição com o cliente1 é feita através da porta 10000 e, com o cliente2, pela porta 10010:



Para finalizar o experimento, pare a execução da simulação clicando no botão  .

IPv6 - Laboratório de relay 6to4

Objetivo

Túneis 6to4 realizados automaticamente, e normalmente sem conhecimento do usuário, pelos sistemas operacionais Windows XP, Vista e 7, bem como por alguns equipamentos de rede, podem levar a uma experiência pior para os usuários, dado que os dados trafegarão por relays públicos 6to4, normalmente fora do país. A conexão será menos estável e apresentará delays maiores.

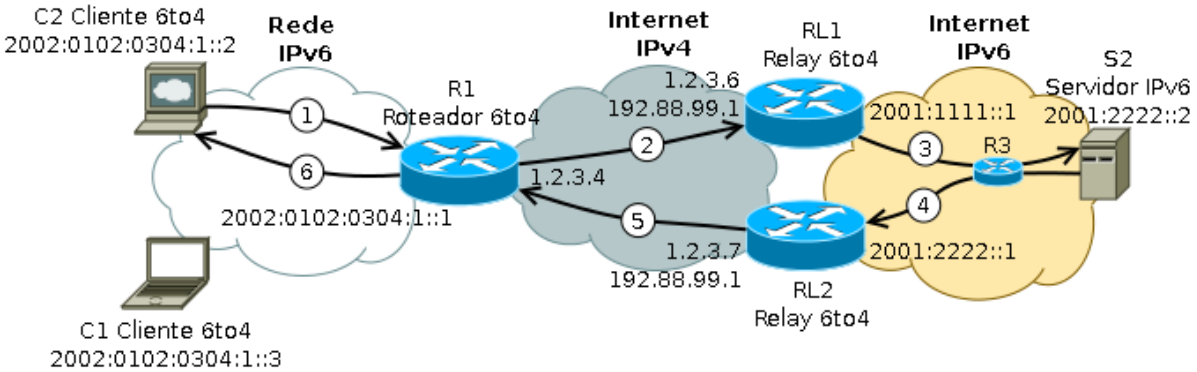
Nesse laboratório mostra-se como ativar um relay 6to4 local, num provedor de serviços que já tenha o IPv6 habilitado, a fim de mitigar o problema descrito anteriormente, eliminando a necessidade dos pacotes de resposta viajarem longas distâncias, passando por relays potencialmente não confiáveis.

Para o presente exercício será utilizado a topologia descrita no arquivo:
tecnicas-transicao-relay6to4.imn.

Introdução Teórica

O 6to4 (**RFC 3056**) é umas das técnicas de transição mais antigas em uso. Com ajuda de relays pilha dupla distribuídos na Internet, abertos, instalados de forma colaborativa por diversas redes, qualquer rede IPv4 pode obter conectividade IPv6, através de túneis 6in4 automáticos. O endereçamento 6to4, utiliza o prefixo de endereço global **2002:wwxx:yyzz::/48**, onde wwxx:yyzz é o endereço IPv4 público do cliente convertido para hexadecimal.

A figura abaixo demonstra o fluxo dos pacotes em uma rede 6to4. É importante notar que não existe a necessidade de os pacotes irem e voltarem pelo mesmo relay 6to4.



Dentre os problemas que afetam o 6to4, pode-se citar problemas de qualidade com relays

públicos e problemas de segurança. Alie-se a isso o fato de que diversos sistemas operacionais suportam túneis 6to4 de forma automática, entre eles o Windows XP, o Windows Vista e o Windows 7. O fato dos sistemas operacionais ativarem os túneis 6to4 sem intervenção ou conhecimento dos usuários traz algumas consequências sérias. Uma delas é que firewalls ou outras medidas de segurança em redes corporativas podem ser inadvertidamente contornadas. Outra, é que, via túnel, os pacotes podem seguir caminhos mais longos, trazendo uma experiência pior para o usuário, em comparação àquela que ele teria se estivesse simplesmente usando IPv4. Um agravante é que não há relays públicos 6to4 no Brasil, ocasionando a ida do pacote para localidades distantes como América do Norte ou Europa, mesmo que a origem e o destino estejam no país.

Provedores de conteúdos e serviços na Internet podem sofrer com a questão, pois ao implantar o IPv6 em um servidor Web, por exemplo, usuários que antes acessavam-no bem via IPv4, podem passar a fazê-lo de forma lenta e instável, via IPv6 obtido automaticamente por meio de um túnel automático 6to4.

É recomendável agir para mitigar esse problema, principalmente porque existe uma medida bastante simples e efetiva, que pode ser utilizada. Deve-se lembrar que o caminho de ida do 6to4 pode ser diferente do caminho de volta. Isto permite que um relay 6to4 seja criado em um servidor Web, ou em uma rede, com o objetivo de responder as requisições recebidas via 6to4. O relay não deve ser público, apenas servirá para responder às requisições dirigidas ao serviço advindas de clientes 6to4. A implementação deste relay não irá reduzir o tempo gasto para receber pacotes 6to4, mas garante que os pacotes 6to4 de resposta saiam da rede com destino ao originador da requisição, já encapsulados em IPv4, e isto dará a vantagem do tempo de resposta ser consideravelmente reduzido, já que não será necessário o pacote ir até o relay localizado no exterior. Esta redução pode melhorar bastante a experiência de acesso de um usuário que utilize 6to4 para acessar um serviço qualquer.

Roteiro Experimental

Experiência 6 - Relay 6to4

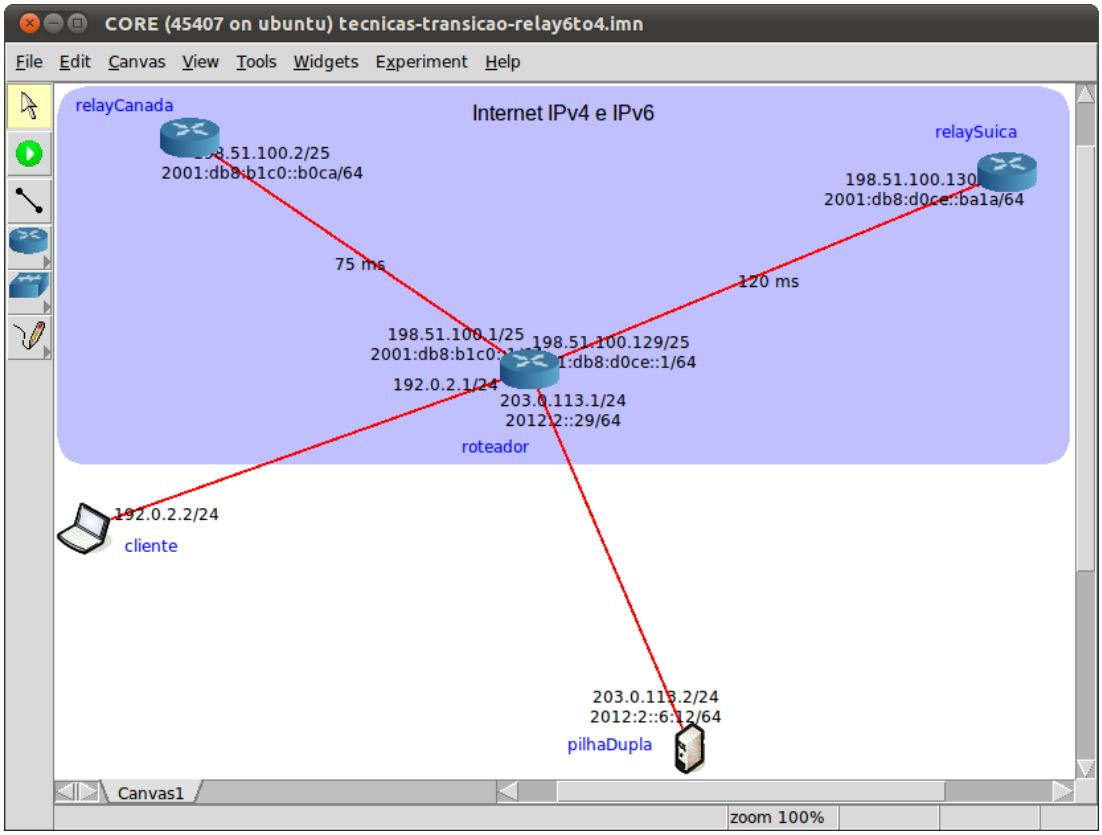
1. Para fazer algumas verificações durante o experimento também será necessária a utilização do programa Wireshark que realiza a verificação dos pacotes que são enviados na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

   Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

2. Inicie o CORE e abra o arquivo “tecnicas-transicao-relay6to4.imn”. A seguinte

topologia inicial de rede deve aparecer deve aparecer:



O objetivo dessa topologia de rede é representar de forma simplificada o ambiente na Internet necessário para perceber o problema e experimentar a técnica de mitigação proposta.

A máquina pilhaDupla representa um servidor qualquer na Internet. Um servidor Web, por exemplo. Esse servidor usa IPv4 e IPv6. A máquina cliente representa um usuário doméstico com Windows, cujo provedor oferece apenas IPv4. Nessa situação um cliente real estabeleceria um túnel 6to4 automático (muito embora a máquina virtual da experiência use na verdade Linux, e o túnel tenha sido configurado por um script quando a experiência foi iniciada). Os relays representam respectivamente um relay 6to4 público na Suiça e outro no Canadá, situação não muito diferente daquela que pode ser encontrada na prática, na Internet.

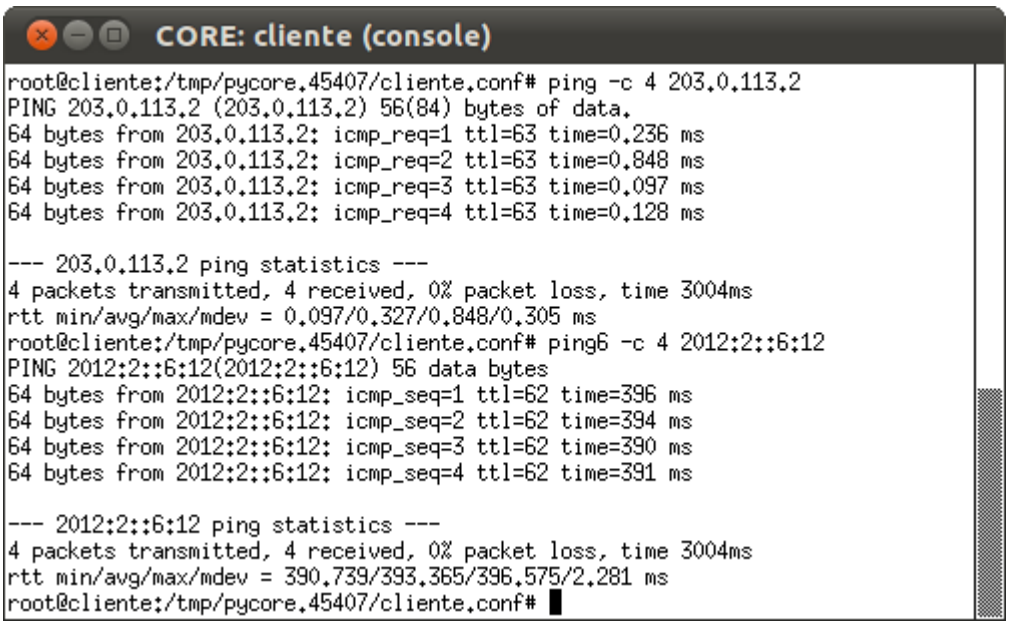
3. Verifique o estado inicial dos nós da topologia.
   a. Inicie a simulação:
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do cliente, através do duplo-clique.

c. Verifique o tempo total de ida e volta entre o cliente e o servidor pilha dupla através do seguinte comando:

```
# ping -c 4 203.0.113.2
# ping6 -c 4 2012:2::6:12
```

O resultado deve ser:



Note que da forma que a rede está configurada, os pacotes do cliente até o servidor passam pelo relayCanada. Numa situação real, eles seriam direcionados para o relay mais próximo ao cliente, por meio do endereço anycast 192.88.99.1. Os pacotes do servidor para o cliente passam, por sua vez, pelo relayCanada. Na Internet real os pacotes seriam direcionados ao relay mais próximo ao servidor, que buscaria quem anuncia a rota para 2002::/16. Note que, na experiência, foram configurados delays de 75ms e 120ms nos enlaces do relayCanada e do relaySuica, respectivamente, para simular os atrasos encontrados na Internet, causados pela distância.

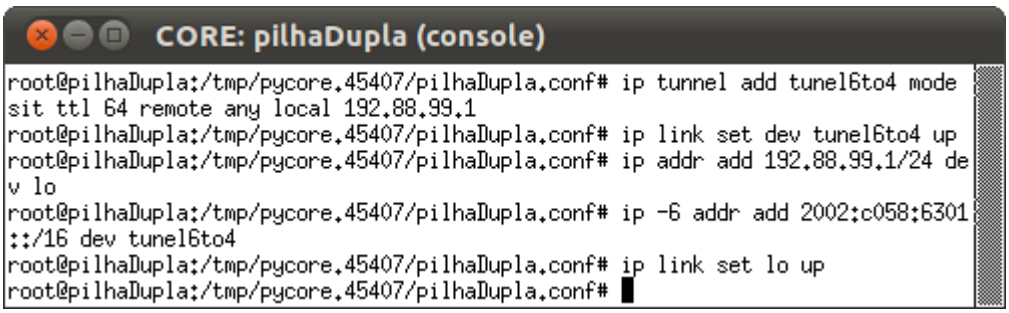
O teste deste item é uma representação da diferença de tempo de ida e volta entre cliente e pilhaDupla comparando o tempo gasto quando utilizado IPv4 e IPv6 via 6to4. Cabe ressaltar que esta experiência simula cliente e servidor próximos fisicamente, por isso os delays IPv4 são baixos.

4. Configure o relay 6to4 no servidor pilha dupla.

a. Abra o terminal de pilhaDupla através do duplo-clique e utilize os seguintes comandos para a configuração:

```
# ip tunnel add tunel6to4 mode sit ttl 64 remote any
  local 192.88.99.1
# ip link set dev tunel6to4 up
# ip addr add 192.88.99.1/24 dev lo
# ip -6 addr add 2002:c058:6301::/16 dev tunel6to4
# ip link set lo up
```

O resultado deve ser:



A configuração acima cria um *relay* 6to4 no próprio servidor pilhaDupla. É atribuído o endereço 192.88.99.1 à interface de *loopback* de pilhaDupla e criado um túnel que utiliza o protocolo 6in4, representado por **sit** no Linux. Os pacotes cujos endereços de destino pertencem a 2002::/16 são encapsulados por pilhaDupla, através do túnel criado. Desse modo, elimina-se o uso de um *relay* 6to4 público, uma vez que o pacote IPv6 já é encapsulado na rede de origem e transmitido desta forma via Internet IPv4.

Caso esta configuração seja utilizada em servidores reais, é necessário verificar com o provedor *upstream* se são utilizados filtros que não permitem *spoofing* (e.g., envio de pacotes cujo endereço de origem não pertence à rede interna do ISP). Se o cliente for um Sistema Autônomo, com blocos IPs próprios, é provável que o *upstream* não faça esse filtro. Se usar IPs do próprio *upstream*, é provável que sim.

5. Verifique a mudança efetuada.

a. Abra o terminal de pilhaDupla, através do duplo-clique.

b. Utilize o seguinte comando para iniciar a captura de pacotes de pilhaDupla:

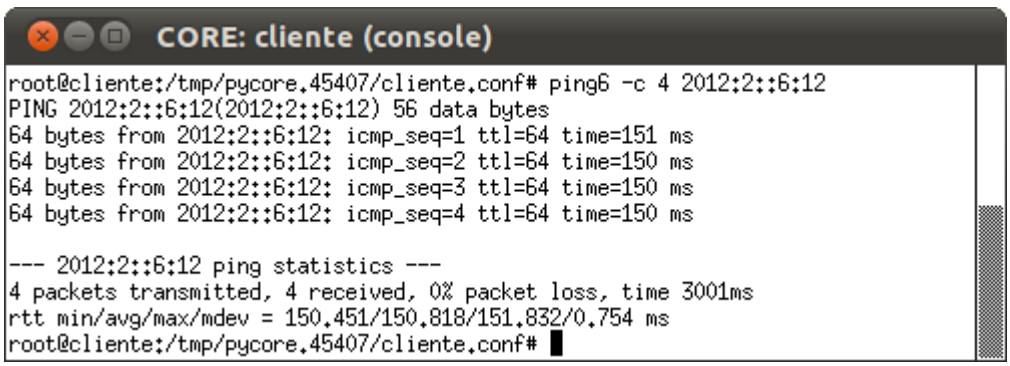
```
# tcpdump -i eth0 -s 0 -w /tmp/captura_relay6to4.pcap
```

O resultado deve ser:

c. No terminal de cliente, utilize novamente o seguinte comando:

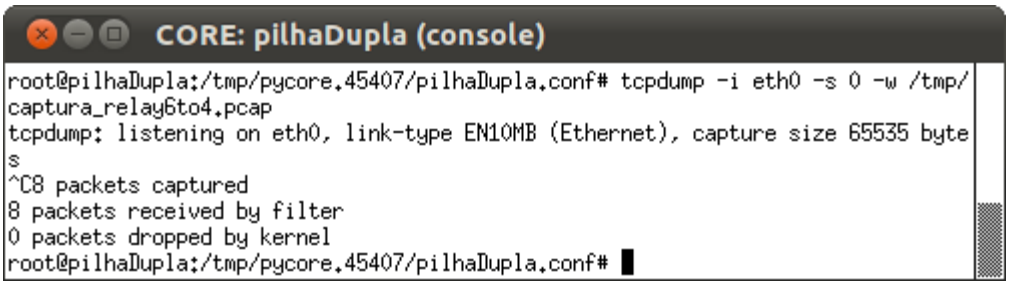
```
# ping6 -c 4 2012:2::6:12
```

O resultado deve ser:



d. No terminal do pilhaDupla, encerre a captura de pacotes através da sequência Ctrl+C.

O resultado deve ser:



Comparando o tempo de ida e volta gasto neste passo com o passo anterior à configuração, podemos notar que os pacotes do servidor para o cliente não passam mais por um relay público. Neste experimento, podemos verificar isso numericamente de forma simples, por causa da redução de 240 ms. Isso significa que os pacotes não estão mais trafegando pelo relaySuica, o que acontecia em seu caminho de volta, do servidor pilhaDupla, até o cliente.

Observe que não é possível efetuar nenhuma melhoria do tempo percorrido de um pacote originado de cliente para pilhaDupla quando a técnica 6to4 é utilizada. A

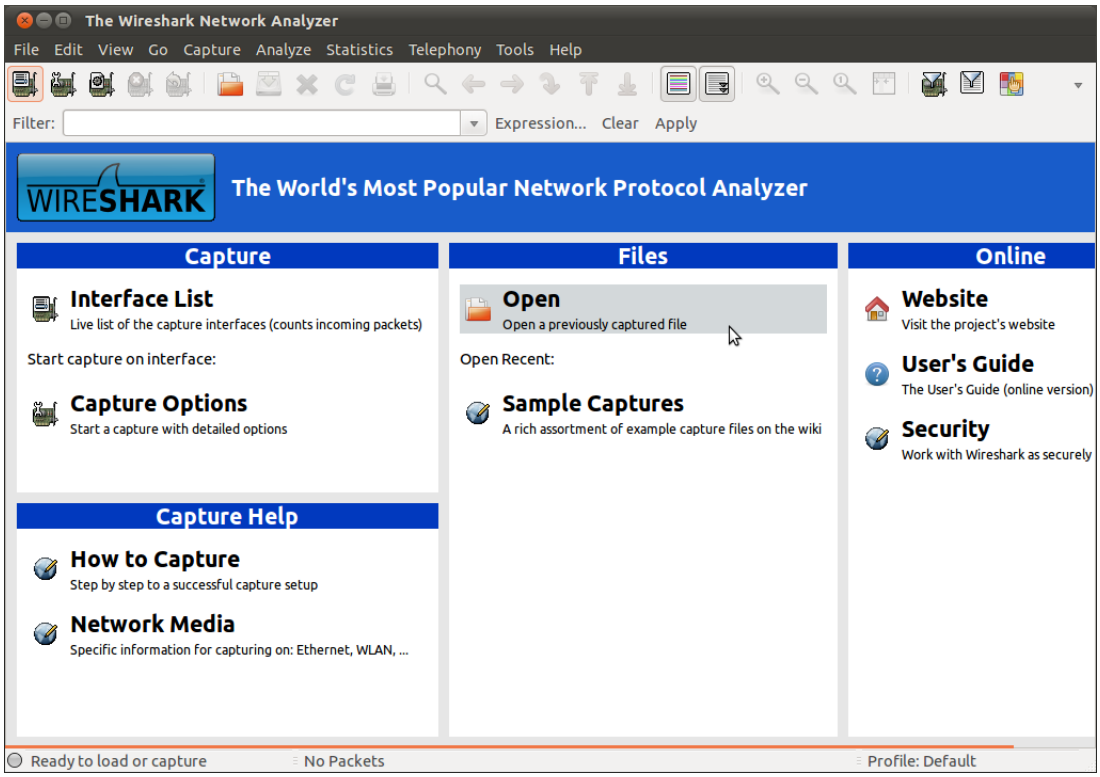
configuração aqui apresentada visa reduzir o tempo de gasto entre pilhaDupla e cliente, dado que o uso do endereço 2002::/16 indica que o cliente possui trânsito IPv4 nativo. Contudo, numa situação real isso é minimizado porque no modelo cliente servidor, usado por exemplo na Web, as requisições são pequenas, enquanto as respostas carregam uma grande quantidade de informação.

6. Encerre a simulação:

   a. aperte o botão  ; ou
   b. utilize o menu Experiment > Stop.

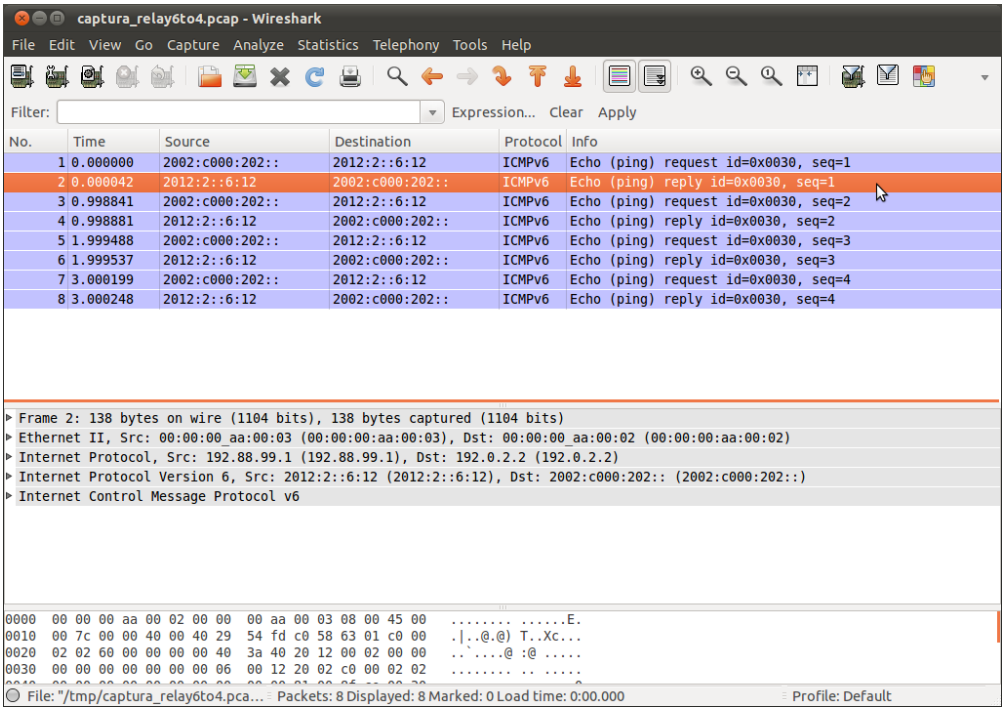
7. Para verificar os pacotes capturados, utilizaremos o programa Wireshark. Para abri-lo, inicie o programa wireshark. Uma maneira é através de um terminal na máquina virtual com o comando:

```
$ wireshark
```

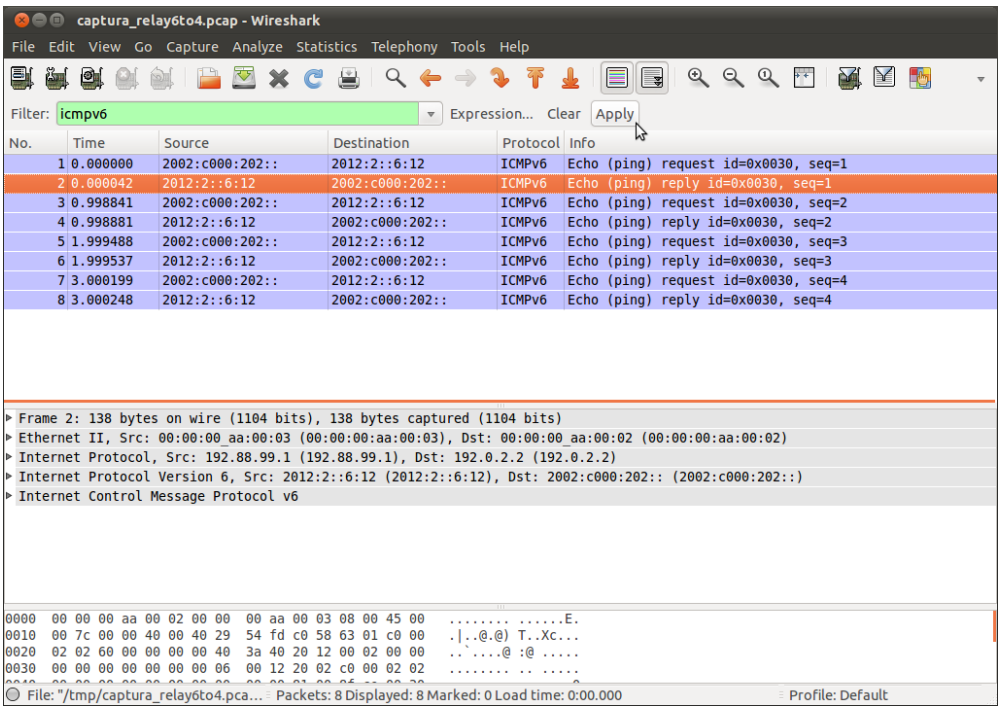


Esse programa tem como principais funcionalidades a captura e análise de pacotes transmitidos por uma interface de rede. Através de seu uso, é possível melhor visualizar os pacotes que trafegam pela rede. Verifique o arquivo de captura previamente obtido.

a. Abra o arquivo `/tmp/captura_relay6to4.pcap` com o menu File>Open:

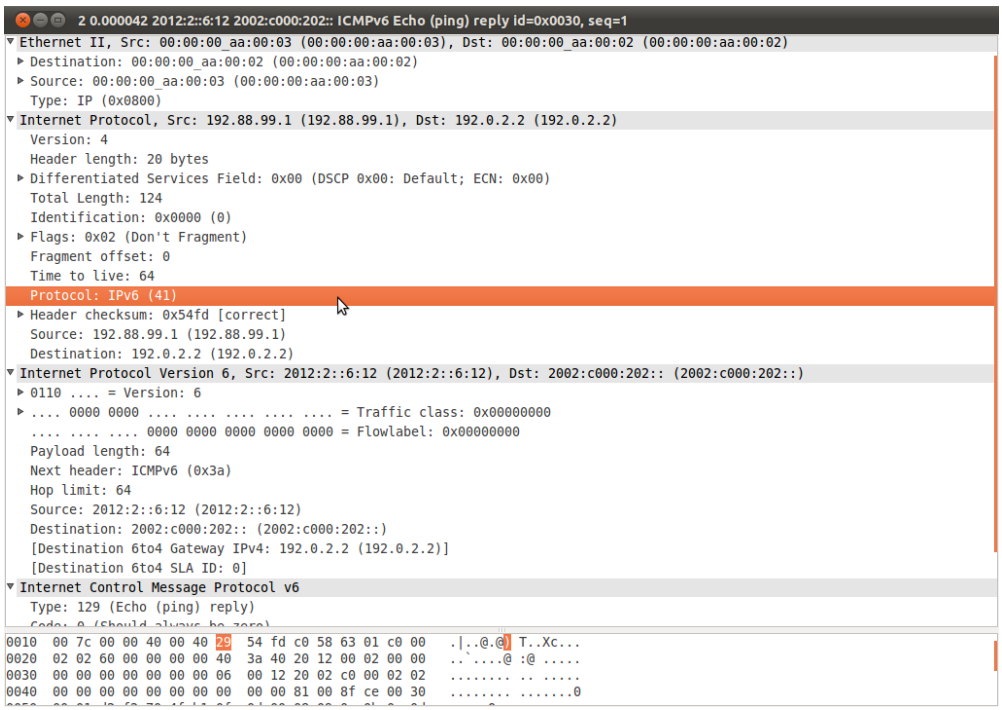


b. Coloque o filtro icmpv6 para visualizar apenas os pacotes que queremos observar:



Note que as versões mais atuais do Wireshark são inteligentes o suficiente para mostrar os pacotes como do tipo ICMPv6, mesmo eles estando encapsulados em pacotes IPv4.

Analise os pacotes *echo reply* veja se os dados contidos nos pacotes conferem com a teoria, prestando atenção à forma com que os pacotes IPv6 foram encapsulados em pacotes IPv4, principalmente quanto ao endereço IPv4 de origem.



Campos importantes:

- Type (camada Ethernet): indica que a mensagem utiliza o protocolo IPv4.
- Protocol (camada IPv4): indica que a mensagem encapsula o protocolo IPv6 (protocolo número 41 - 6in4).
- Destination (camada IPv4): o destino é o endereço IPv4 de cliente (192.0.2.2).
- Source (camada IPv4): a fonte é o endereço *anycast* IPv4 de pilhaDupla (192.88.99.1).
- Destination (camada IPv6): o destino é o endereço IPv6 de cliente (2002:c000:202::).
- Source (camada IPv6): a fonte é o endereço IPv6 de pilhaDupla (2012:2::6:12).

IPv6 - Laboratório de 6rd

Objetivo

A técnica IPv6 Rapid Deployment (6rd) foi desenvolvida pela ISP francesa Free, levando apenas seis semanas desde o início do projeto até o fornecimento de IPv6 para seus clientes. Essa técnica de tunelamento permite implantar o IPv6 para usuários em uma rede que suporta apenas IPv4. O 6rd, contudo, deve ser suportado pelos equipamentos dos clientes (CPEs).

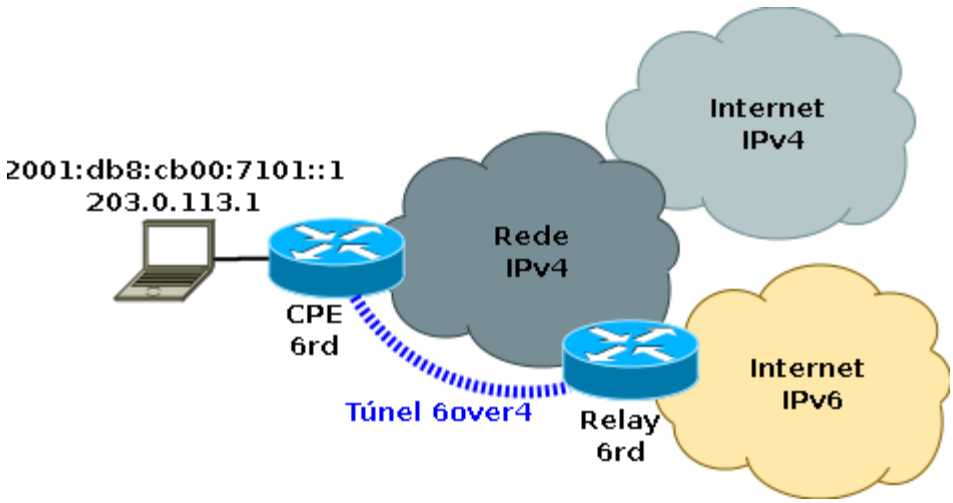
Esse laboratório é dividido em duas partes:

1. Configuração manual do *relay* e de um CPE, fornecendo uma rede /64; e
2. Configuração manual de um CPE, fornecendo uma rede /56.

Para o presente exercício serão utilizadas as topologias descrita nos arquivos: **tecnicas-transicao-6rd.imn** e **tecnicas-transicao-6rd-multiplosCPEs.imn**.

Introdução Teórica

Para explicar o funcionamento do 6rd vamos analisar a topologia da rede com a sua implementação:



Analisando a figura das nuvens, é possível notar que o 6rd depende basicamente de dois componentes:

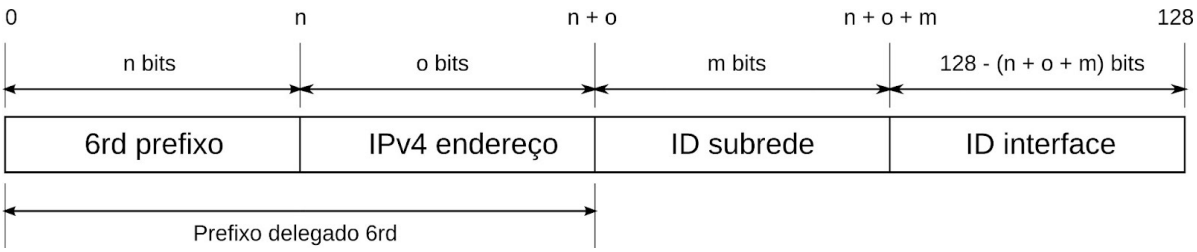
- CPE 6rd: instalado como interface entre a rede da operadora e do usuário.
- *Relay* 6rd: instalado na interface entre a rede IPv4 da operadora e a Internet IPv6.

O CPE 6rd é um CPE tradicional (xDSL modem, cable modem, 3G modem etc), cujo

software foi modificado para permitir o uso do 6rd. O uso de um CPE 6rd dificulta a implementação da técnica, uma vez que requer a substituição, lógica ou física, de equipamentos em campo. Tal modificação nos CPEs normalmente é viável nos casos em que o provedor gerencia remotamente o equipamento, sendo capaz de fazer *upgrades* em seu *firmware*.

O *relay* 6rd é um equipamento que vai encapsular e desencapsular pacotes para trafegarem corretamente nas redes IPv4 e IPv6.

O CPE 6rd atribui ao usuário um endereço IPv4, como um CPE normal. Entretanto um endereço IPv6 também é atribuído ao usuário. Este endereço IPv6 é um endereço IPv6 público válido, mas é construído de maneira específica, baseado no endereço IPv4:



No 6rd, o tamanho ***n*** do prefixo e o tamanho ***o*** do endereço IPv4, que formam o prefixo delegado 6rd, são escolhas do provedor de acesso. Para permitir que a auto-configuração de endereço *stateless* funcione é necessário que o tamanho deste prefixo ***n + o*** seja menor que 64 bits.

Normalmente utiliza-se ***n***=32, ***o***=32 e ***m***=0. Pode-se, contudo, aumentar o número de bits utilizados por ***n*** para além de 32, forçando o endereço IPv4 a utilizar parte dos 64 bits menos significativos, o que impede o funcionamento da auto-configuração *stateless*. Para evitar que isto ocorra, o endereço IPv4 pode ocupar menos de 32 bits. Tal configuração é possível se os endereços IPv4 fizerem parte de uma mesma rede, pois pode-se omitir o prefixo da rede. Por exemplo, se todos os endereços IPv4 forem do tipo 198.51.0.0/16, os 16 bits que representam os números 198 e 51 podem ser omitidos e a representação do endereço IPv4 necessitará somente de 16 bits, ao invés dos 32 bits necessários para representar o endereço completo.

Roteiro Experimental

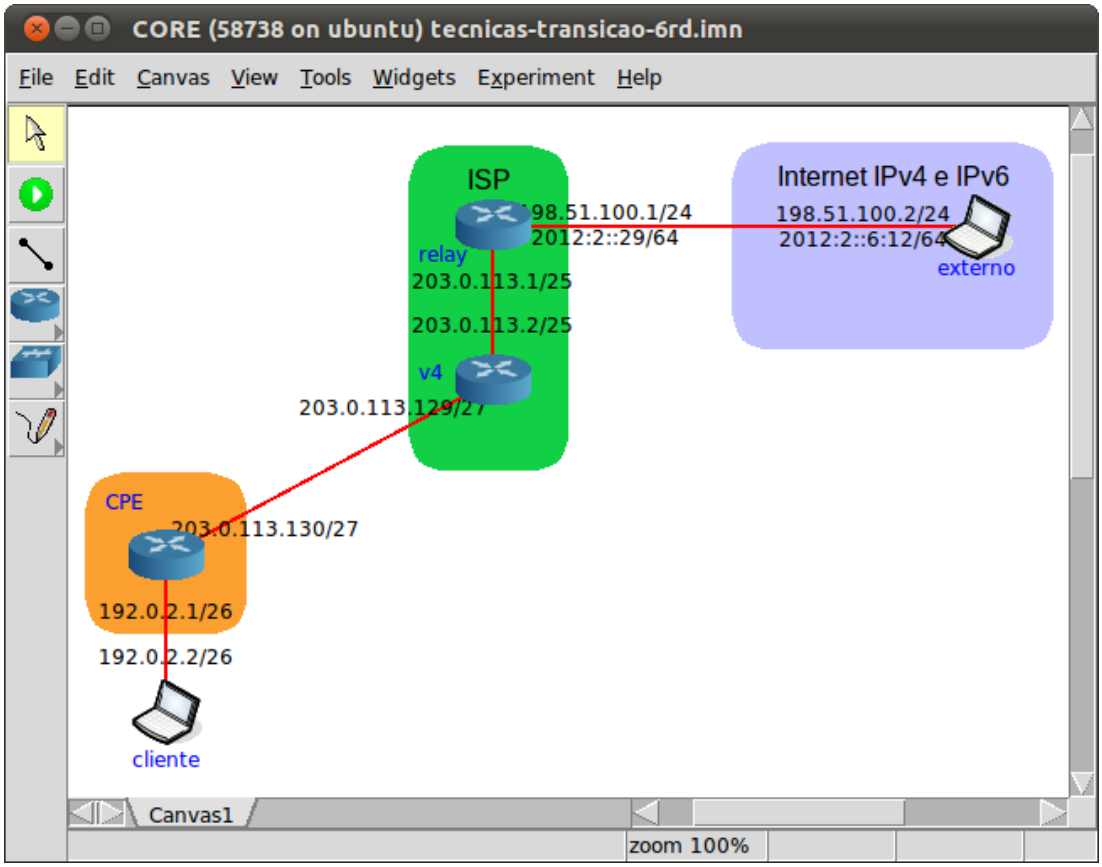
Experiência 7 - Configuração 6rd no relay e em um CPE (/64)

1. Para fazer algumas verificações durante o experimento também será necessária a utilização do programa Wireshark que realiza a verificação dos pacotes que são enviados na rede. Na máquina virtual, utilize um Terminal para rodar o comando:

```
$ sudo apt-get install wireshark
```

Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

2. Inicie o CORE e abra o arquivo “tecnicas-transicao-6rd.imn”. A seguinte topologia inicial de rede deve aparecer:



O objetivo dessa topologia de rede é representar o mínimo necessário para que a implantação do 6rd seja percebida, uma vez que pelo roteador v4 entre o CPE e o *relay* transporta apenas IPv4. A rede foi configurada com rotas estáticas de forma que todas as máquinas possam conectar-se via IPv4, assim como conectar-se via IPv6 entre o *relay* e o servidor externo.

Neste experimento, o cliente receberá uma rede /64, sendo o prefixo composto pelo prefixo IPv6 do ISP e o endereço IPv4 do CPE.

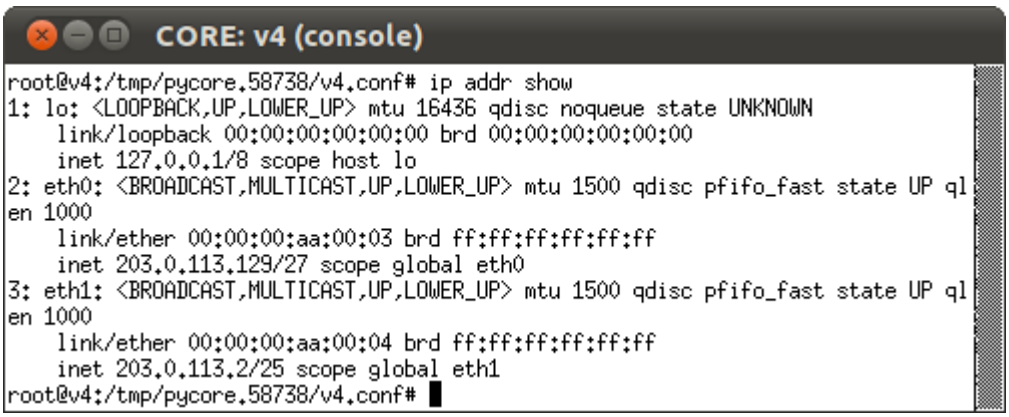
3. Verifique a configuração dos nós da topologia.
   a. Inicie a simulação:
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Start.

   b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do roteador v4, através do duplo-clique.

c. Verifique que o roteador v4 não suporta IPv6 através do seguinte comando:

```
# ip addr show
```

O resultado deve ser:

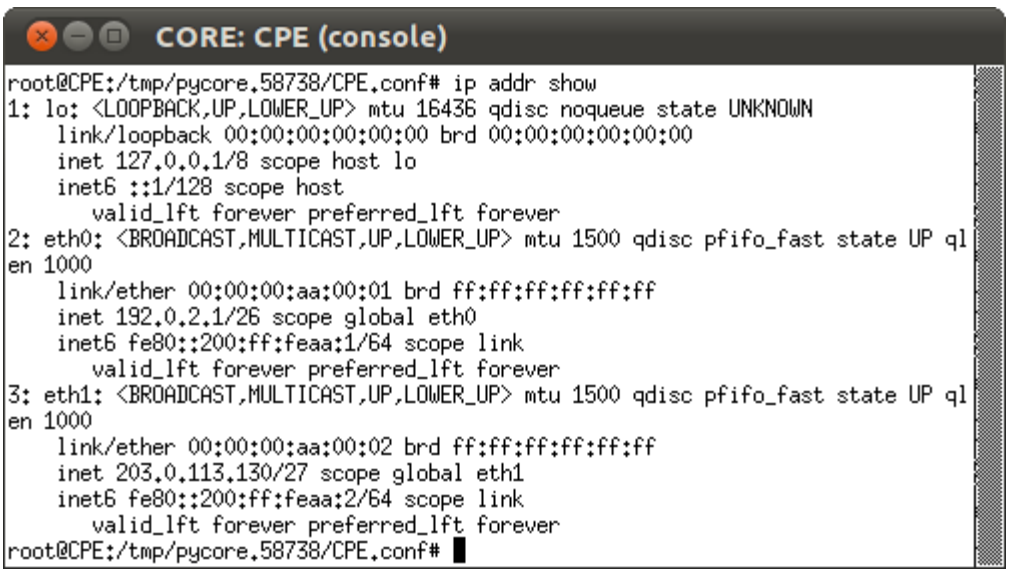


Pode observar-se que não há endereços IPv6 nas interfaces, nem mesmo endereços do tipo *link local*, o que indica que o roteador não suporta o protocolo.

d. Abra o terminal do CPE, através do duplo-clique e verifique a configuração através do comando:

```
# ip addr show
```

O resultado deve ser:



Pode observar-se que, no contexto do IPv6, há somente os endereços *link*

local e o *loopback* configurados.

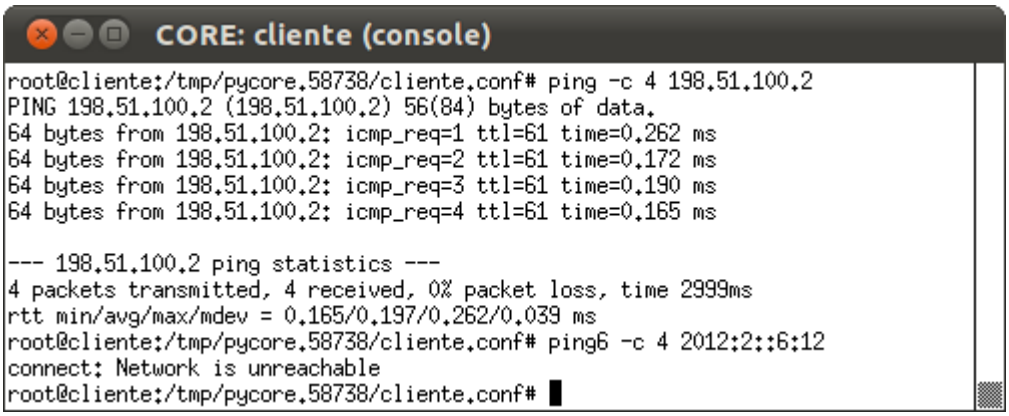
4. Verifique a conectividade entre cliente e externo.

   a. Abra o terminal de cliente, através do duplo-clique.

   b. Utilize os seguintes comandos para verificar a conectividade:

```
# ping -c 4 198.51.100.2
# ping6 -c 4 2012:2::6:12
```

   O resultado deve ser:



   Observe que há conectividade IPv4, mas não há ainda conectividade IPv6. A configuração de 6rd proverá essa conectividade.

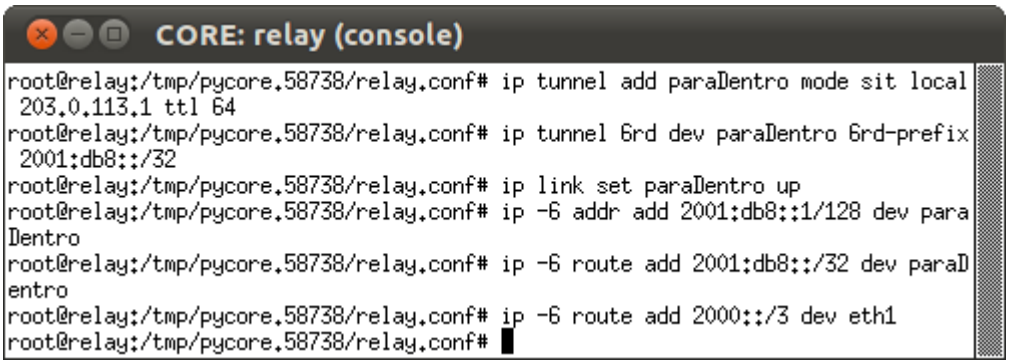
5. Configure o 6rd no *relay* e no CPE e um endereço IPv6 no cliente.

   No momento de preparo deste laboratório, a configuração do CPE e do *relay* é suportado para o 6rd no Linux a partir do kernel 2.6.33 e em equipamentos Cisco e Juniper.

   a. Abra o terminal de *relay*, através do duplo-clique, e utilize os seguintes comandos para a configuração:

```
# ip tunnel add paraDentro mode sit local 203.0.113.1 ttl 64
# ip tunnel 6rd dev paraDentro 6rd-prefix 2001:db8::/32
# ip link set paraDentro up
# ip -6 addr add 2001:db8::1/128 dev paraDentro
# ip -6 route add 2001:db8::/32 dev paraDentro
# ip -6 route add 2000::/3 dev eth1
```

O resultado deve ser:

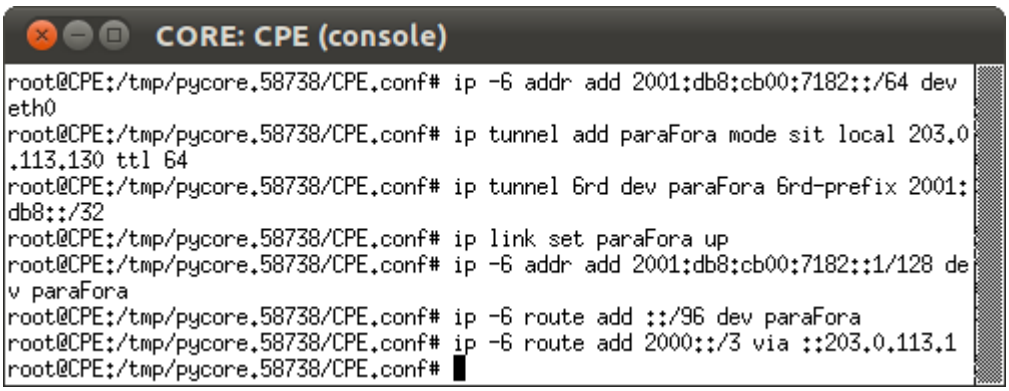


O *relay* possui conectividade IPv4 e IPv6 para a Internet, aqui representada pelo nó externo, enquanto a rede interna está devidamente configurada somente para o uso de IPv4. A configuração acima cria o túnel paraDentro, cujo nome refere-se à rede interna do ISP. O túnel associa-se a um endereço IPv4 da rede interna do ISP em sua criação (203.0.113.1), utilizando o protocolo 41, representada no Linux pelo tipo **sit**. O prefixo IPv6 fornecido ao ISP (2001:db8::/32) é utilizado como parâmetro de configuração do túnel configurado para o 6rd. Após inicializar o túnel, são configuradas as rotas: uma para a rede interna através do túnel e outra para a Internet, representada pela interface eth1.

b. Abra o terminal de CPE através do duplo-clique e utilize os seguintes comandos para a configuração:

```
# ip -6 addr add 2001:db8:cb00:7182::/64 dev eth0
# ip tunnel add paraFora mode sit local 203.0.113.130 ttl 64
# ip tunnel 6rd dev paraFora 6rd-prefix 2001:db8::/32
# ip link set paraFora up
# ip -6 addr add 2001:db8:cb00:7182::1/128 dev paraFora
# ip -6 route add ::/96 dev paraFora
# ip -6 route add 2000::/3 via ::203.0.113.1
```

O resultado deve ser:



O CPE possui conectividade IPv4 dentro da rede do ISP, conforme citado no item anterior. A configuração inicialmente atribui um endereço na interface de rede local (eth0) referente ao prefixo IPv6 do CPE, sendo tal prefixo composto pelo prefixo IPv6 do ISP (2001:db8::/32) e o endereço IPv4 do CPE convertido em hexadecimal (203.0.113.130 → cb 00 71 82), resultando no prefixo 2001:db8:cb00:7182::/64.

Em seguida, é configurado o túnel paraFora, cujo nome refere-se à rede interna do ISP, por onde se trafegam os pacotes destinados à Internet. O túnel associa-se a um endereço IPv4 da rede interna do ISP em sua criação (203.0.113.130), utilizando o protocolo 41, representada no Linux pelo tipo **sit**.

O prefixo IPv6 fornecido ao ISP é 2001:db8::/32 e é utilizado como parâmetro de configuração do túnel configurado para o 6rd. Após inicializar o túnel, são configuradas as rotas: uma para viabilizar o uso do túnel em IPv4 (::/96) e outro para a Internet, através do encapsulamento para o relay, via IPv4 (::203.0.113.1).

c. Abra o terminal de cliente, através do duplo-clique, e utilize os seguintes comandos para a configuração:

```
# ip -6 addr add 2001:db8:cb00:7182::b1c0/64 dev eth0
# ip -6 route add default via 2001:db8:cb00:7182::
```

O resultado deve ser:

Neste experimento, o endereço IPv6 do cliente é configurado manualmente, pois o objetivo é configurar o 6rd na rede do ISP. Para uma configuração mais robusta, o CPE deve ser configurado para prover a autoconfiguração. Essa funcionalidade prevista no protocolo IPv6 é apresentada na experiência da categoria **Funcionalidades** sob o título **Autoconfiguração de endereços Stateless**.

Observe que neste experimento, os valores apresentados na introdução teórica de ***n***, ***o*** e ***m*** são, respectivamente, 32, 32 e 0 bits, de modo que o maior prefixo IPv6 do ISP foi utilizado juntamente com o endereço IPv4 integral do CPE.

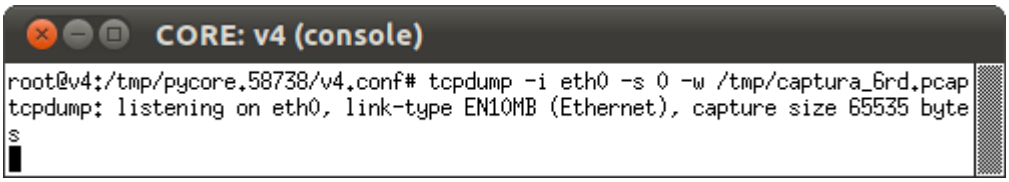
6. Verifique a conectividade via IPv6 entre cliente e externo.

   a. Abra o terminal do roteador v4, através do duplo-clique.

   b. Utilize o seguinte comando para iniciar a captura de pacotes do roteador:

```
# tcpdump -i eth0 -s 0 -w /tmp/captura_6rd.pcap
```

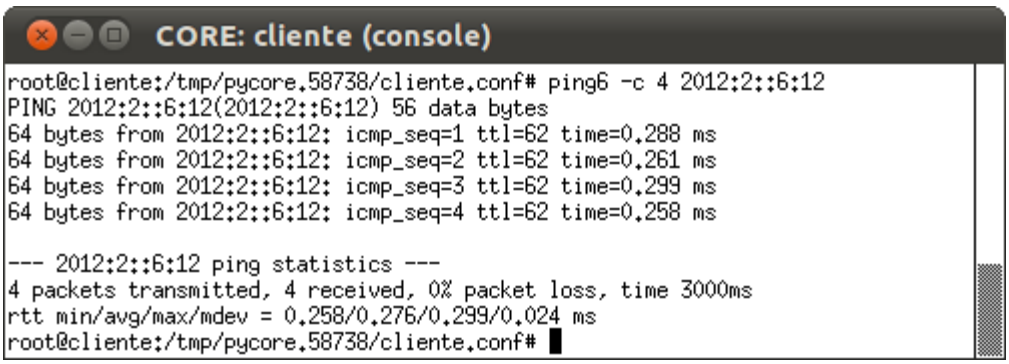
   O resultado deve ser:



   c. No terminal de cliente, utilize novamente o seguinte comando:

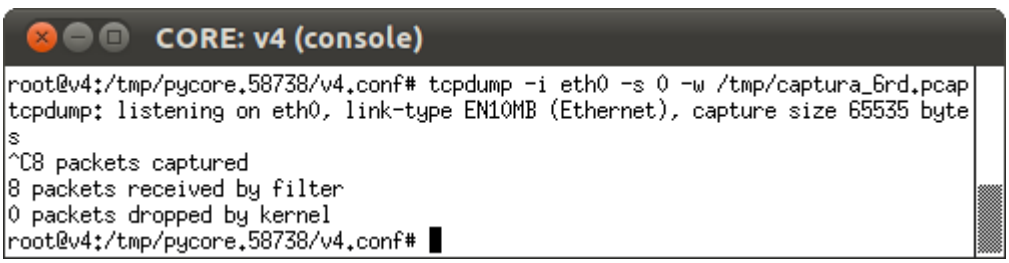
```
# ping6 -c 4 2012:2::6:12
```

   O resultado deve ser:



   d. No terminal do roteador v4, encerre a captura de pacotes através da sequência Ctrl+C.

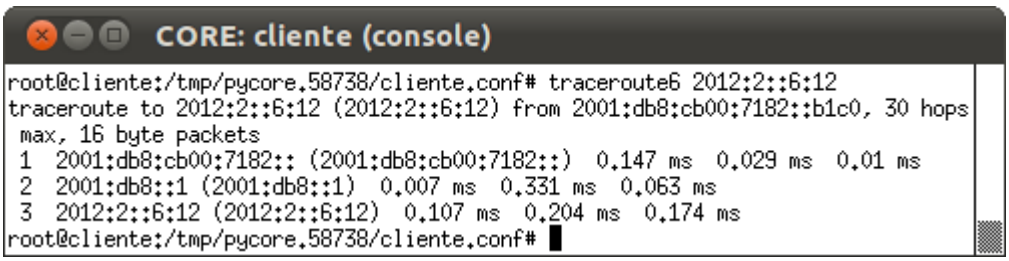
   O resultado deve ser:



e. No terminal de cliente, utilize o seguinte comando:

```
# traceroute6 2012:2::6:12
```

O resultado deve ser:

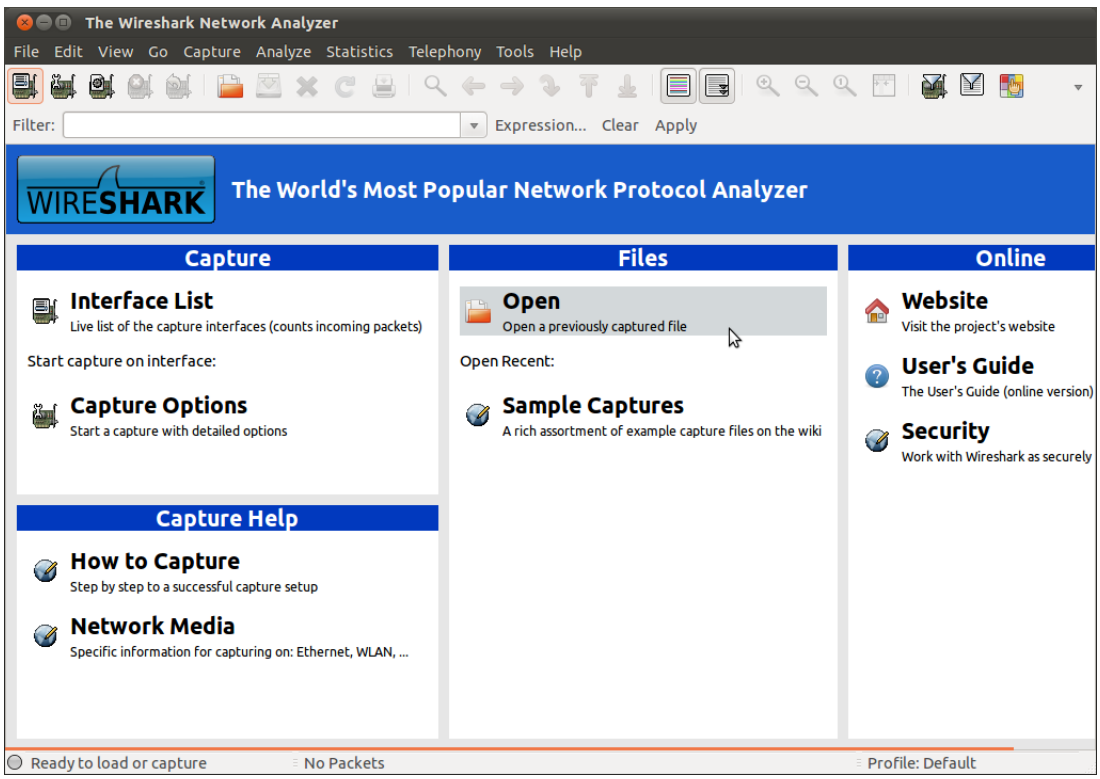


7. Encerre a simulação:

a. aperte o botão  ; ou
b. utilize o menu Experiment > Stop.

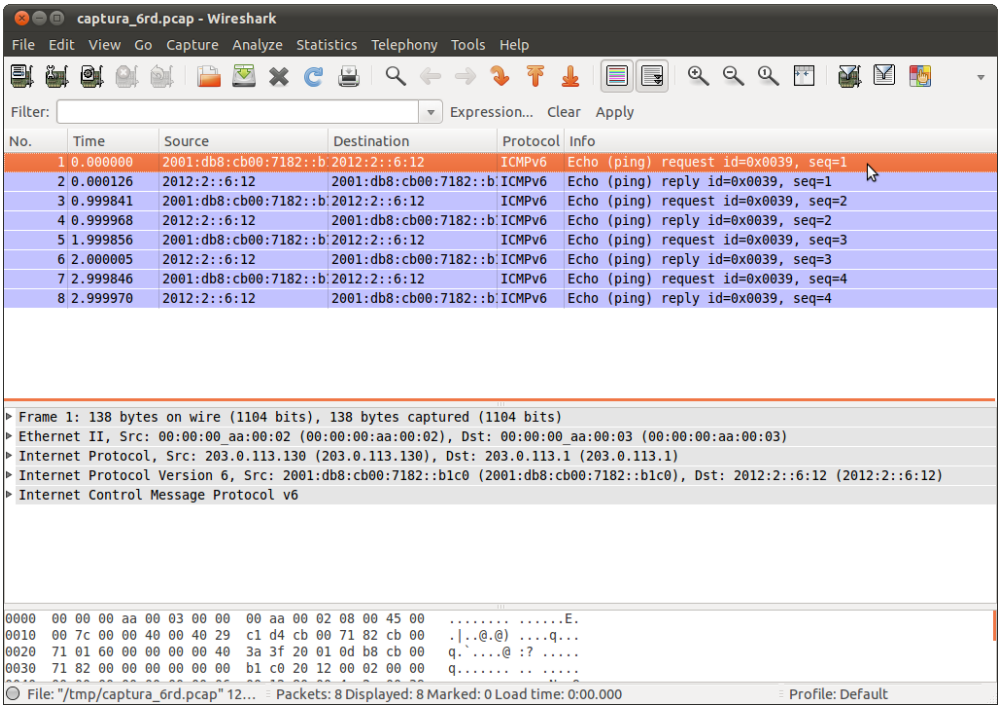
8. Para verificar os pacotes capturados, utilizaremos o programa Wireshark. Para abri-lo, inicie o programa wireshark. Uma maneira é através de um terminal na máquina virtual com o comando:

```
$ wireshark
```

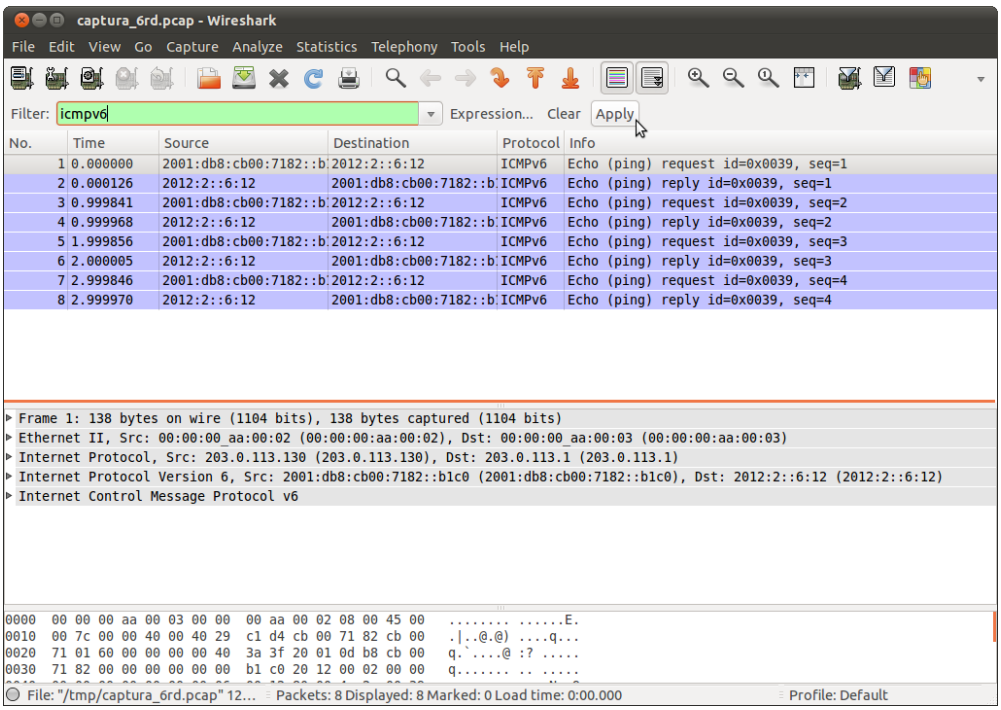


Esse programa tem como principais funcionalidades a captura e análise de pacotes transmitidos por uma interface de rede. Através de seu uso, é possível melhor visualizar os pacotes que trafegam pela rede. Verifique o arquivo de captura previamente obtido.

a. Abra o arquivo `/tmp/captura_6rd.pcap` com o menu File>Open:

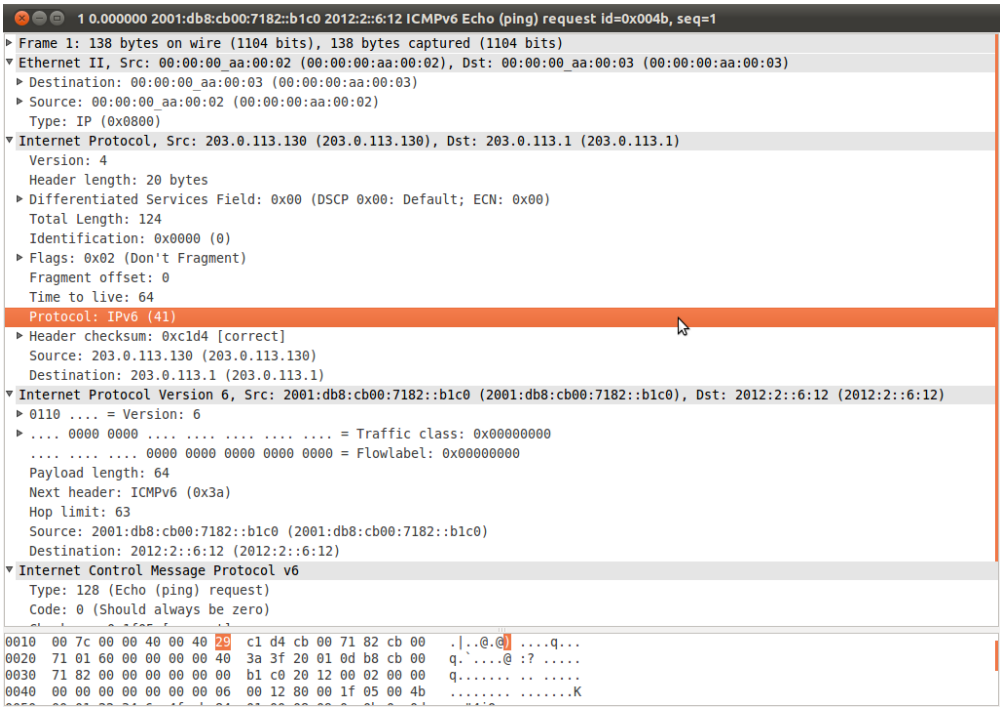


b. Coloque o filtro icmpv6 para visualizar apenas os pacotes que queremos observar:



Note que as versões mais atuais do Wireshark são inteligentes o suficiente para mostrar os pacotes como do tipo ICMPv6, mesmo eles estando encapsulados em pacotes IPv4.

Analise os pacotes *echo request* e *echo reply* veja se os dados contidos nos pacotes conferem com a teoria, prestando atenção à forma com que os pacotes IPv6 foram encapsulados em pacotes IPv4.

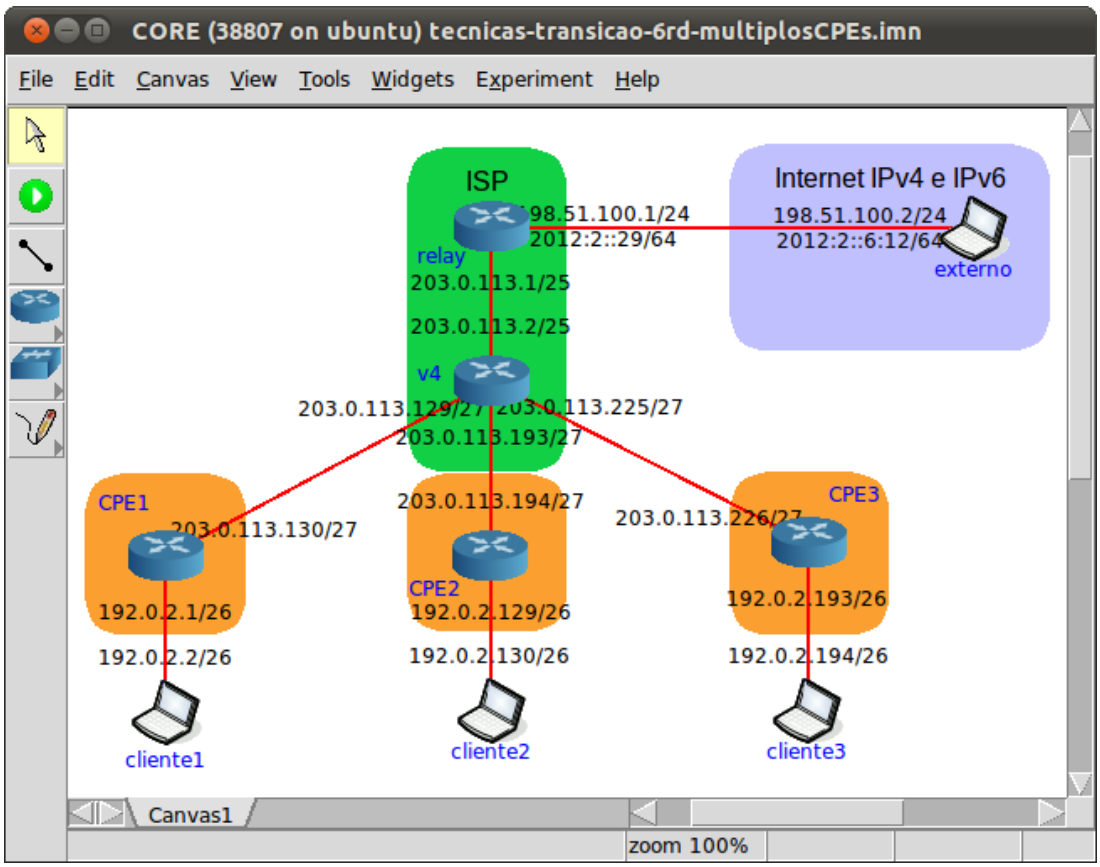


Campos importantes:

- Type (camada Ethernet): indica que a mensagem utiliza o protocolo IPv4.
- Protocol (camada IPv4): indica que a mensagem encapsula o protocolo IPv6 (protocolo número 41 - 6in4).
- Destination (camada IPv4): o destino é o endereço IPv4 do *relay* (203.0.113.1).
- Source (camada IPv4): a fonte é o endereço IPv4 do CPE (203.0.113.130).
- Destination (camada IPv6): o destino é o endereço IPv6 de externo (2012:2::6:12).
- Source (camada IPv6): a fonte é o endereço IPv6 do cliente (2001:db8:cb00:7182::b1c0).

Experiência 8 - Configuração 6rd em um CPE (/56)

1. No CORE, abra o arquivo “tecnicas-transicao-6rd-multiplos-CPEs.imn”. A seguinte topologia inicial de rede deve aparecer deve aparecer:



Comparando com a topologia do experimento anterior, foram adicionados dois conjuntos de CPEs e clientes. Neste experimento, os CPEs 2 e 3 já estão configurados, bem como o *relay*.

Em contraste com o experimento anterior, o ISP divulgará um prefixo menor (2001:db8:cab0::/48) e o cliente receberá um prefixo maior (/56). Isso é realizado através de uma propriedade do 6rd, na qual o prefixo IPv4 da rede interna do ISP é omitido.

Neste experimento, o cliente receberá uma rede /56, sendo o prefixo composto pelo prefixo IPv6 divulgado pelo ISP e pelos octetos finais do endereço IPv4 do CPE.

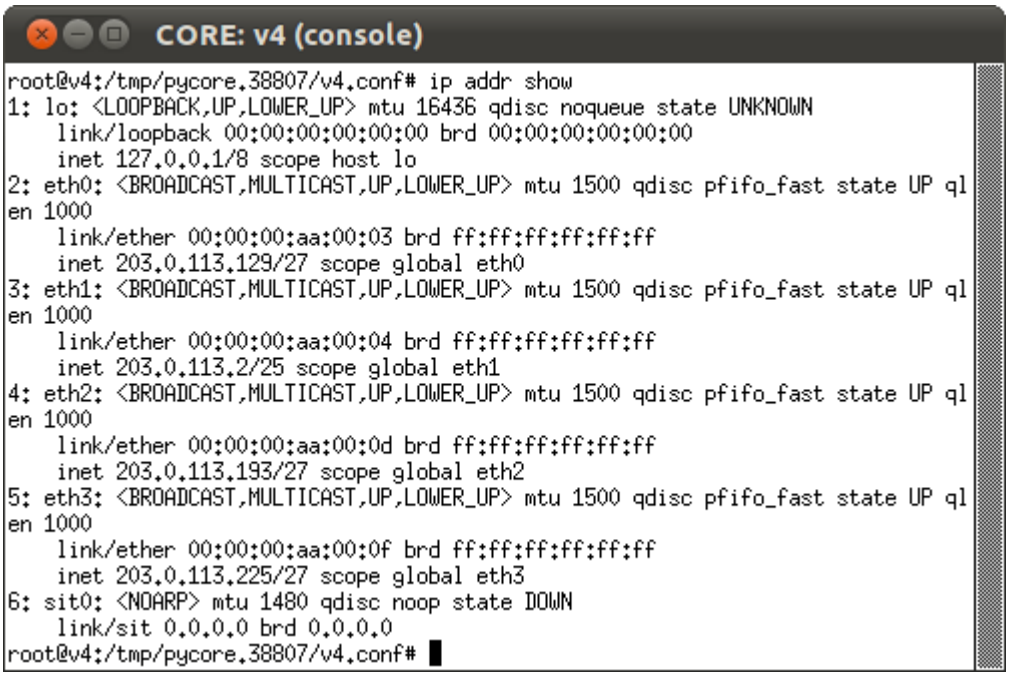
2. Verifique a configuração dos nós da topologia.
   a. Inicie a simulação:
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Start.

b. Espere até que o CORE termine a inicialização da simulação e abra o terminal do roteador, através do duplo-clique.

c. Verifique que o roteador v4 não suporta IPv6 através do seguinte comando:

```
# ip addr show
```

O resultado deve ser:

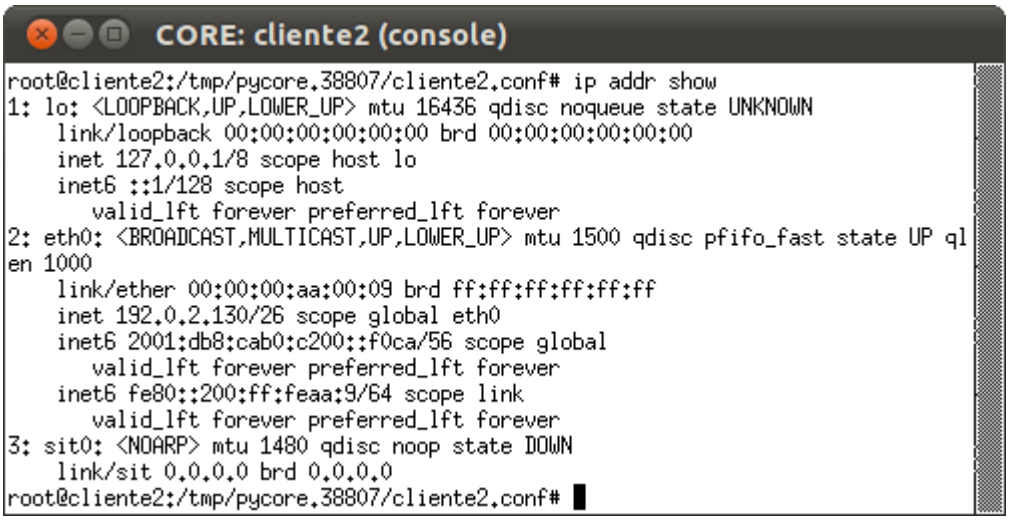


Pode observar-se que não há endereços IPv6 nas interfaces, nem mesmo endereços do tipo *link local*, o que indica que o roteador não suporta o protocolo.

d. Abra o terminal de cliente2, através do duplo-clique e verifique a configuração através do comando:

```
# ip addr show
```

O resultado deve ser:

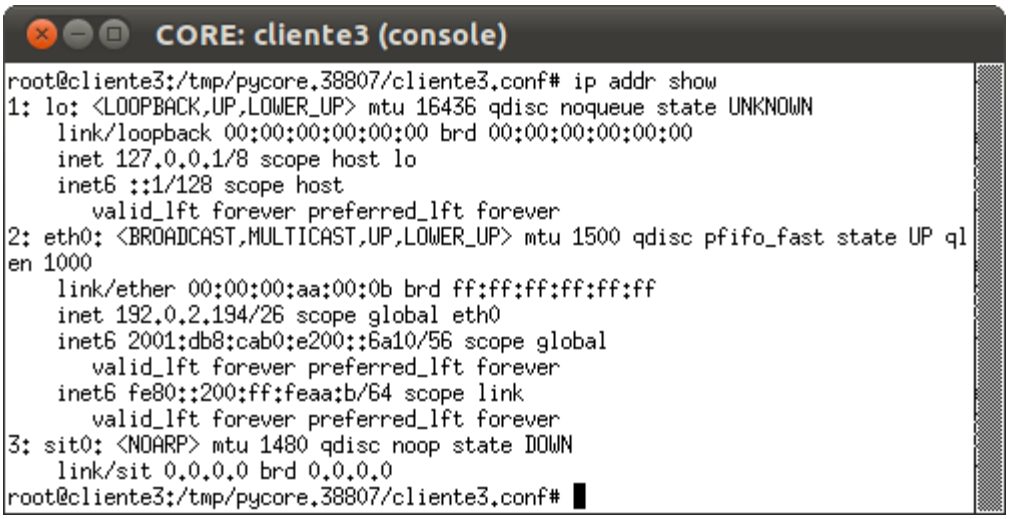


Pode observar-se que o endereço 2001:db8:cab0:c200::f0ca/56 está configurado na interface que comunica com o CPE2.

e. Abra o terminal de cliente3, através do duplo-clique e verifique a configuração através do comando:

```
# ip addr show
```

O resultado deve ser:

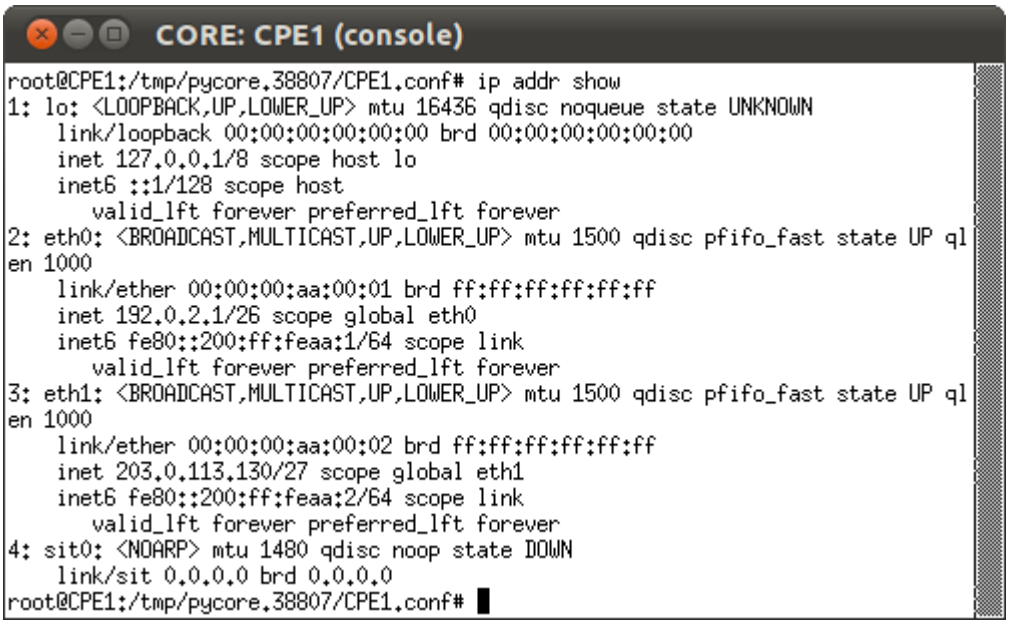


f. Pode observar-se que o endereço 2001:db8:cab0:e200::6a10/56 está configurado na interface que comunica com o CPE3.

g. Abra o terminal de CPE1, através do duplo-clique e verifique a configuração através do comando:

```
# ip addr show
```

O resultado deve ser:



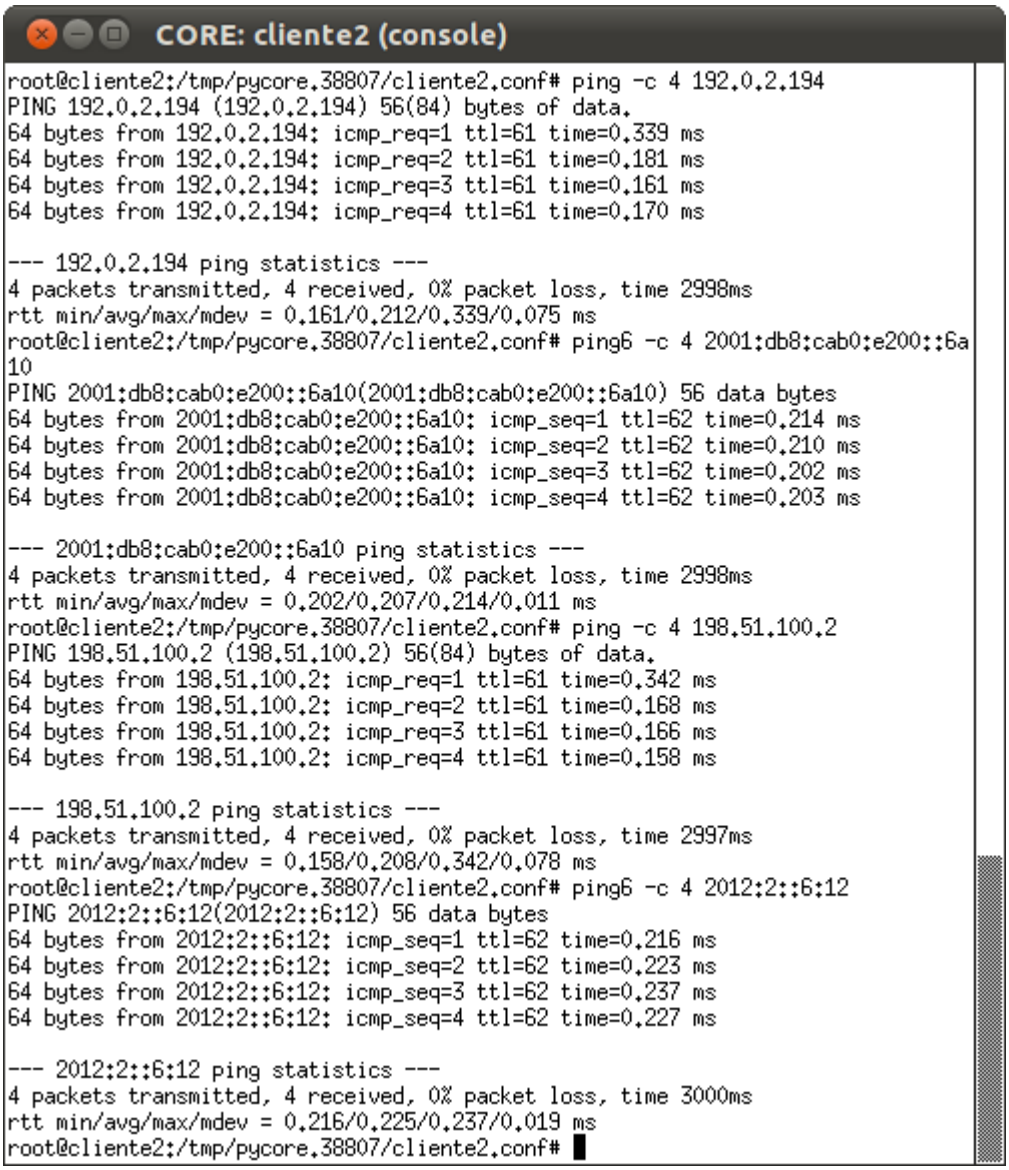
Pode observar-se que, no contexto do IPv6, há somente os endereços *link local* e o *loopback* configurados.

3. Verifique a conectividade entre os clientes.

a. Utilize os seguintes comandos em cliente2 para verificar a conectividade com cliente3 e externo:

```
# ping -c 4 192.0.2.194
# ping6 -c 4 2001:db8:cab0:e200::6a10
# ping -c 4 198.51.100.2
# ping6 -c 4 2012:2::6:12
```

O resultado deve ser:

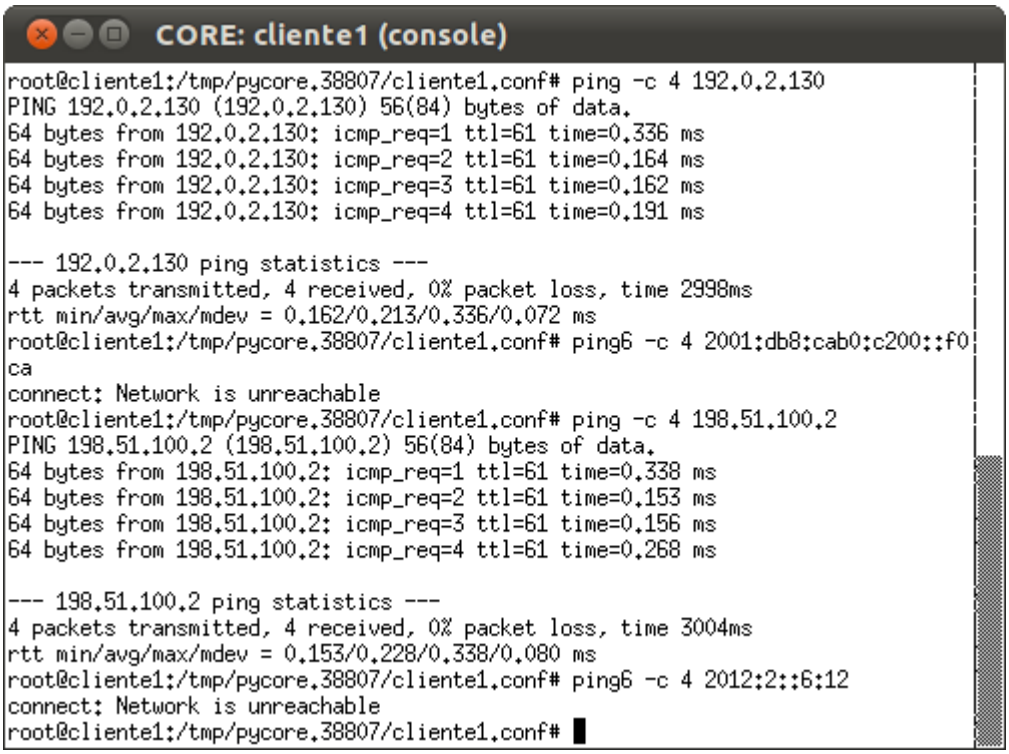


Observe que há conectividade IPv4 e IPv6 partindo de cliente2 tanto internamente quanto externamente em relação à rede do ISP.

b. Utilize os seguintes comandos em cliente1 para verificar a conectividade com cliente2 e externo:

```
# ping -c 4 192.0.2.130
# ping6 -c 4 2001:db8:cab0:c200::f0ca
# ping -c 4 198.51.100.2
# ping6 -c 4 2012:2::6:12
```

O resultado deve ser:



Observe que há conectividade IPv4, mas não há ainda conectividade IPv6. A configuração de 6rd proverá essa conectividade.

4. Configure o 6rd no CPE1 e um endereço IPv6 no cliente1.

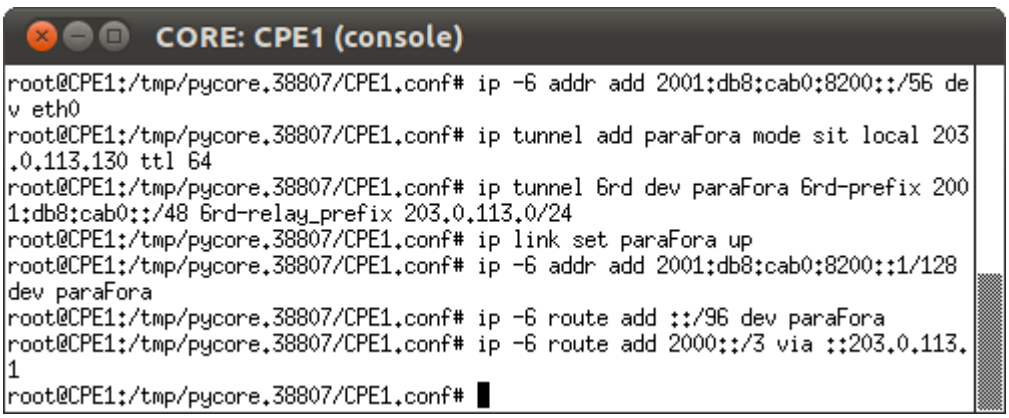
   a. Neste experimento, o *relay* já está configurado. Os seguintes comandos encontram-se nesse roteiro a título de comparação da configuração utilizada em relação ao experimento anterior.

```
# ip tunnel add paraDentro mode sit local 203.0.113.1 ttl 64
# ip tunnel 6rd dev paraDentro 6rd-prefix 2001:db8:cab0::/48
6rd-relay_prefix 203.0.113.0/24
# ip link set paraDentro up
# ip -6 addr add 2001:db8:cab0::1/128 dev paraDentro
# ip -6 route add 2001:db8:cab0::/48 dev paraDentro
# ip -6 route add 2000::/3 dev eth1
```

b. Abra o terminal de CPE1 através do duplo-clique e utilize os seguintes comandos para a configuração:

```
# ip -6 addr add 2001:db8:cab0:8200::/56 dev eth0
# ip tunnel add paraFora mode sit local 203.0.113.130 ttl 64
# ip tunnel 6rd dev paraFora 6rd-prefix 2001:db8:cab0::/48
6rd-relay_prefix 203.0.113.0/24
# ip link set paraFora up
# ip -6 addr add 2001:db8:cab0:8200::1/128 dev paraFora
# ip -6 route add ::/96 dev paraFora
# ip -6 route add 2000::/3 via ::203.0.113.1
```

O resultado deve ser:

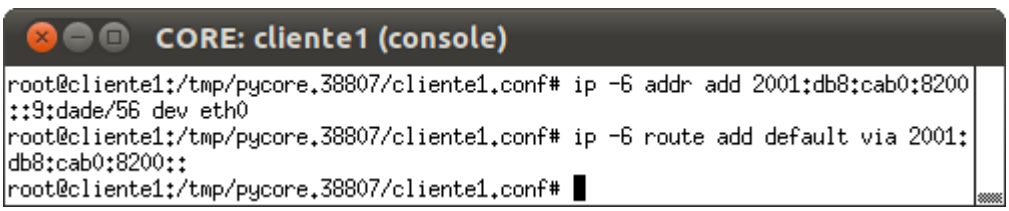


Na configuração apresentada, são destacadas as diferenças quando comparadas com o mesmo passo do experimento anterior. O prefixo IPv6 do CPE1 é composto pelo prefixo IPv6 divulgado pelo ISP (2001:db8:cab0::/48) e o restante do endereço IPv4 do CPE1 após a remoção da prefixo IPv4 da rede interna do ISP convertido em hexadecimal (203.0.113.130/24 → 130 decimal → 82 hexadecimal), resultando no prefixo 2001:db8:cab0:8200::/56.

c. Abra o terminal de cliente1, através do duplo-clique, e utilize os seguintes comandos para a configuração:

```
# ip -6 addr add 2001:db8:cab0:8200::9:dade/56 dev eth0
# ip -6 route add default via 2001:db8:cab0:8200::
```

O resultado deve ser:



Observe que neste experimento, os valores apresentados na introdução teórica de ***n***, ***o*** e ***m*** são, respectivamente, 48, 8 e 8 bits, de modo que somente uma parte do prefixo IPv6 do ISP foi utilizado juntamente com parte do endereço IPv4 do CPE1. Nesta configuração, é atribuido ao cliente um prefixo IPv6 /56, permitindo a criação de 256 subredes /64 (2^8 = 256).

5. Verifique a conectividade via IPv6 partindo de cliente1.

a. Abra o terminal de cliente1, através do duplo-clique, e utilize os seguintes comandos para verificar a conectividade com cliente2, cliente3 e externo:

```
# ping6 -c 2 2001:db8:cab0:c200::f0ca
# ping6 -c 2 2001:db8:cab0:e200::6a10
# ping6 -c 2 2012:2::6:12
# traceroute6 2001:db8:cab0:e200::6a10
# traceroute6 2012:2::6:12
```

O resultado deve ser:

IPv6 - Laboratório de NAT64 e DNS64

Objetivo

O NAT64 e o DNS64 em conjunto permitem que, em uma rede, hosts somente IPv6 acessem servidores somente IPv4 na Internet, por meio da tradução dos protocolos. Nesse laboratório o objetivo é configurar o NAT64 e o DNS64 para verificar o funcionamento da técnica.

Para o presente exercício será utilizado a topologia descrita no arquivo: **tecnicas-transicao-nat64.imn**.

Ao contrário dos outros experimentos desenvolvidos pela equipe IPv6.br, o arquivo de topologia deste experimento requer o uso da máquina virtual que serve de host para o CORE como parte integrante do experimento em si, devido a incompatibilidade do software utilizado em relação à técnica de emulação da rede do CORE. Também é importante destacar que este o arquivo de topologia não funciona no VMware Player. recomendamos a utilização do VirtualBox 4.1.10 ou superior.

Introdução Teórica

Neste módulo prático, testaremos o funcionamento da técnica de tradução NAT64 (**RFC 6146**), aplicável para nós somente IPv6 acessarem a Internet IPv4, utilizando uma técnica stateful de tradução de pacotes IPv6 em IPv4. Ela necessita de uma técnica auxiliar para a conversão do DNS, chamada de DNS64 (**RFC 6147**). São sistemas distintos, mas que trabalham em conjunto para permitir a comunicação entre as redes IPv6 e IPv4.

Roteiro Experimental

Experiência 9 - NAT64 e DNS64

Os passos a seguir, ensinam como instalar o módulo do kernel do Linux desenvolvido pelo projeto ecdysis, disponível no endereço http://ecdysis.viagenie.ca/download/ecdysis-nf-nat64-20101117.tar.gz, necessário para o funcionamento do NAT64. Porém, para fins didáticos, estes primeiros passos já foram realizados no cenário disponível no arquivo “tecnicas-transicao-nat64.imn,” sendo necessário apenas a sua configuração, que será detalhada mais a frente, a partir do item 3.

1. Para realizar o download, primeiro faça um pequeno cadastro no site do projeto ecdysis e, em seguida, você receberá por e-mail o link para baixar os arquivos necessários para instalação. Após a realização do download, descompacte o arquivo e prossiga com a instalação via terminal do seguinte modo:

```
$ tar xvzf ecdysis-nf-nat64-20101117.tar.gz
$ cd ecdysis-nf-nat64-20101117
$ make
$ sudo make install
```

Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

Se você utiliza uma versão de kernel no Linux superior a 2.6.38, utilize os seguintes comandos, ao invés do apresentado anteriormente:

```
$ tar xvzf ecdysis-nf-nat64-20101117.tar.gz
$ cd ecdysis-nf-nat64-20101117
$ wget
http://www.viagenie.ca/pipermail/ecdysis-discuss/attachments/20110725/7f43f0b9/
attachment.el | patch
$ make
$ sudo make install
```

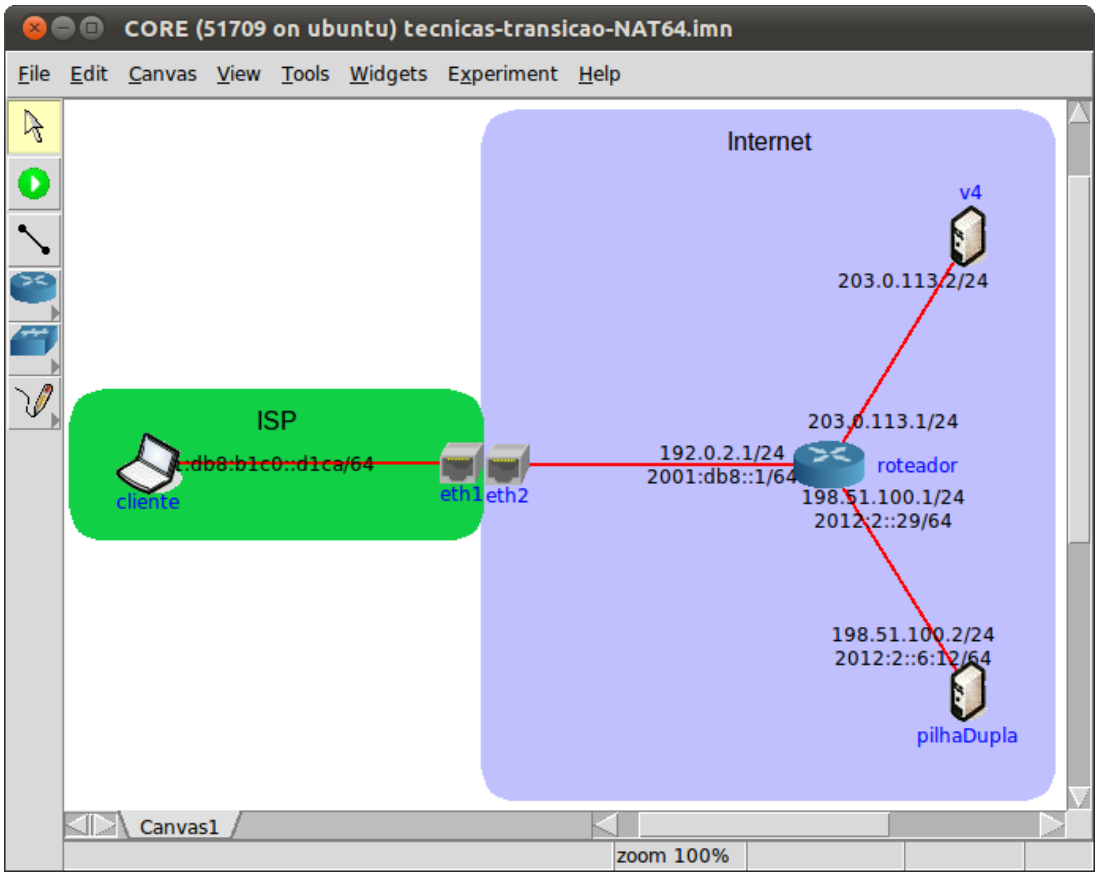
Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

2. Para este experimento, também é necessário o uso do BIND, que é uma implementação do protocolo DNS (Domain Name System), com versão superior a 9.8. Na máquina virtual, utilize um Terminal para rodar os comandos:

```
$ wget ftp://ftp.isc.org/isc/bind9/9.8.1-P1/bind-9.8.1-P1.tar.gz
$ tar xvzf bind-9.8.1-P1.tar.gz
$ cd xvzf bind-9.8.1-P1
$ ./configure
$ make
$ sudo make install
```

Antes da instalação será solicitada a senha do usuário core. Digite “core” para prosseguir com a instalação.

3. Inicie o CORE e abra o arquivo “tecnicas-transicao-nat64.imn”, localizada no diretório do desktop “Transição/NAT64”. A seguinte topologia inicial de rede deve aparecer:



O objetivo dessa topologia de rede é representar o mínimo necessário para que o uso conjunto de NAT64 e DNS64 seja entendido.

4. Verifique a configuração dos nós da topologia.
   a. Inicie a simulação:
      i. aperte o botão  ; ou
      ii. utilize o menu Experiment > Start.

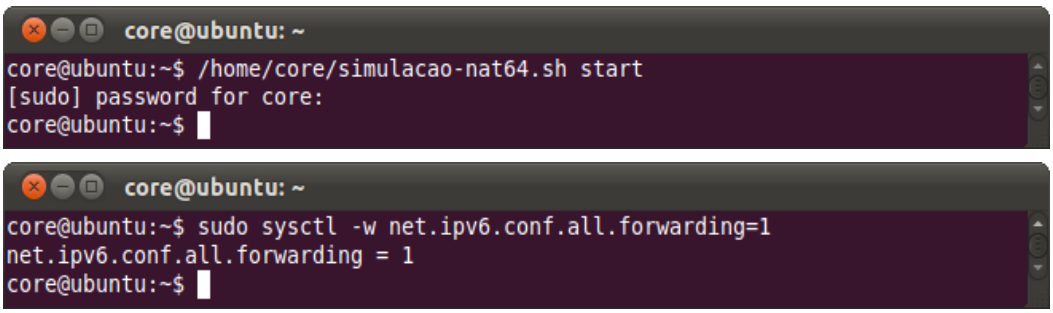
   b. Espere até que o CORE termine a inicialização da simulação.

   c. Na máquina virtual (fora do CORE), abra o Terminal e rode o seguinte comando:

```
# /home/core/simulacao-nat64.sh start
# sudo sysctl -w net.ipv6.conf.all.forwarding=1
```

Caso necessário, digite “core” para prosseguir com a configuração.

O resultado deve ser:



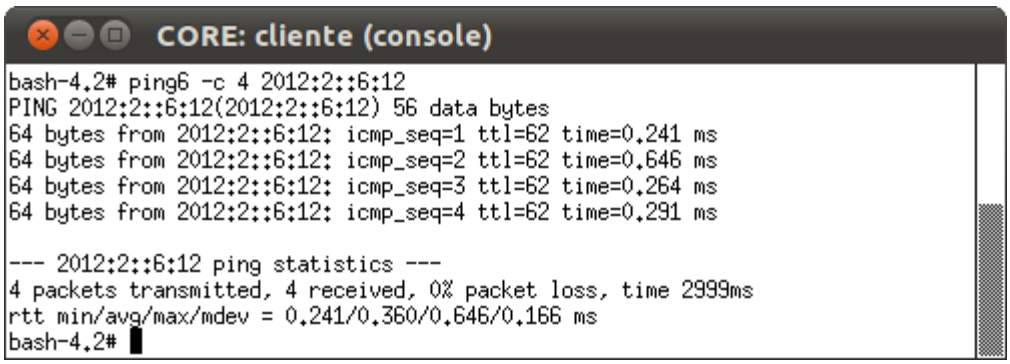
Na topologia deste experimento, a máquina virtual é utilizada como nó responsável pelo DNS64 e NAT64 e provê roteamento para o nó cliente. Esse script interliga a máquina virtual que serve de base para o CORE com a emulação que é executada neste, identificando interfaces de rede, setando bridges e rotas estáticas.

5. Verifique a conectividade entre cliente e pilhaDupla.

Abra o terminal do cliente através do duplo-clique e utilize o seguinte comando:

```
# ping6 -c 4 2012:2::6:12
```

O resultado deve ser:



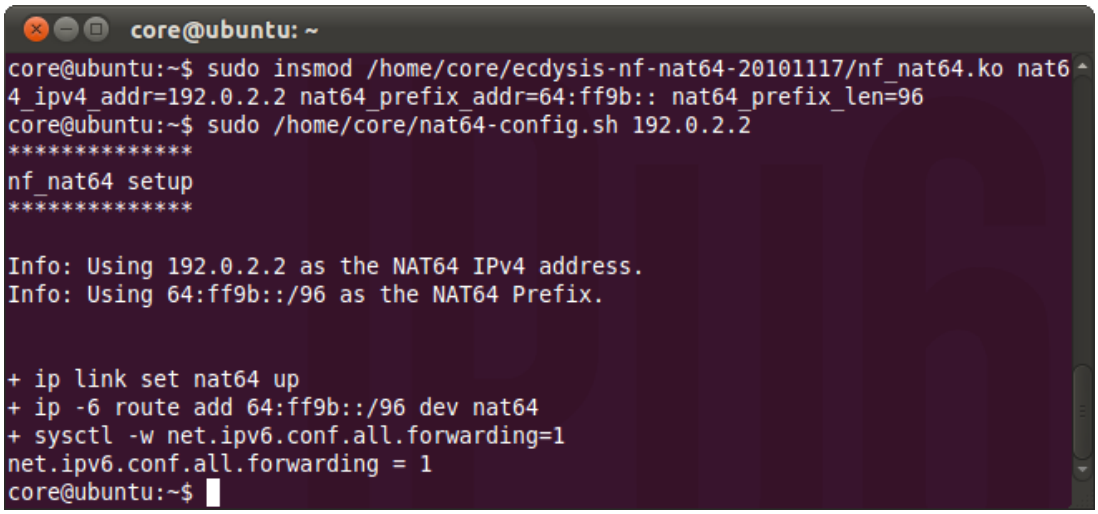
6. Configure o NAT64 na máquina virtual.

Abra o terminal da máquina virtual e utilize os seguintes comandos para a configuração:

```
# sudo insmod /home/core/ecdysis-nf-nat64-20101117/nf_nat64.ko
nat64_ipv4_addr=192.0.2.2 nat64_prefix_addr=64:ff9b::
nat64_prefix_len=96
# sudo /home/core/nat64-config.sh 192.0.2.2
```

Caso necessário, digite “core” para prosseguir com a configuração.

O resultado deve ser:



A técnica apresentada utiliza um módulo desenvolvido para Linux e o mesmo é carregado através do primeiro comando, com os seguintes parâmetros:

```
nat64_ipv4_addr=192.0.2.2 “Endereço IPv4 conectado a Internet”
nat64_prefix_addr=64:ff9b:: “Prefixo IPv6 utilizado para mapear endereços IPv4”
nat64_prefix_len=96 “Tamanho do prefixo IPv6 utilizado no mapeamento”
```

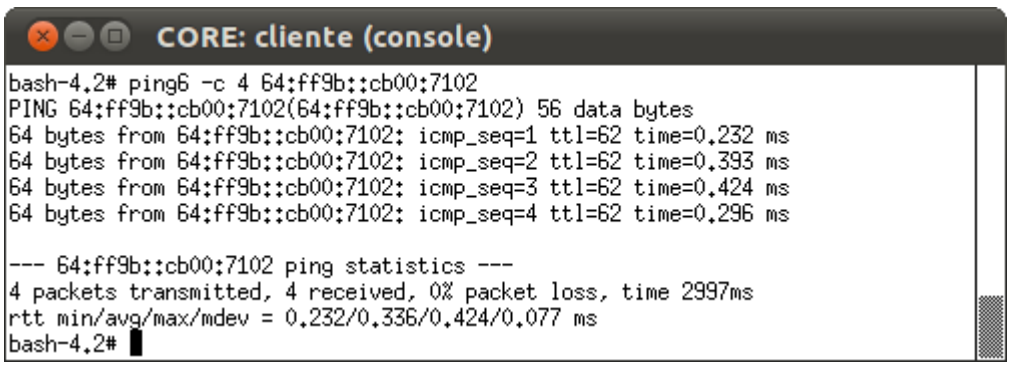
O segundo comando é responsável por habilitar a interface **nat64** e configurar a rota usada para a tradução, além de configurar o roteamento de pacotes na máquina virtual.

7. Verifique a conectividade via IPv6 entre cliente e v4.

Abra o terminal de cliente através do duplo-clique e utilize o seguinte comando:

```
# ping6 -c 4 64:ff9b::cb00:7102
```

O resultado deve ser:



Conforme apresentado no item anterior, os endereços IPv4 são acessíveis via NAT64 através da composição do prefixo IPv6 (64:ff9b::/96) seguido do endereço IPv4 escrito em hexadecimal (IPv4 de v4: 203.0.113.2 → cb 00 71 02).

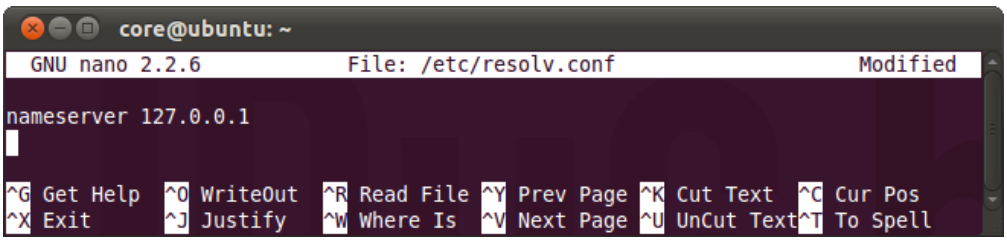
8. Neste passo, verificaremos o funcionamento do serviço DNS sem a opção de DNS64.
   a. Abra o terminal da máquina virtual e utilize o seguinte comando para editar a configuração DNS:

```
# sudo nano /etc/resolv.conf
```

   b. Edite o arquivo para conter somente a linha:

```
nameserver 127.0.0.1
```

   O resultado deve ser:

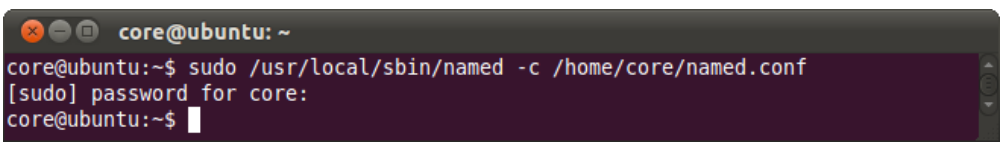


   Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

   c. Abra o terminal da máquina virtual e utilize o seguinte comando para inicializar o serviço DNS:

```
# sudo /usr/local/sbin/named -c /home/core/named.conf
```

   O resultado deve ser:



   d. Abra o terminal de cliente através do duplo-clique e utilize os seguintes comandos:

```
# dig -t ANY servidor.pd
# dig -t ANY servidor.v4
```

O resultado deve ser:

CORE: cliente (console)

```
bash-4.2# dig -t any servidor.pd

; <<>> DiG 9.8.1-P1 <<>> -t any servidor.pd
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 19941
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 1, ADDITIONAL: 2

;; QUESTION SECTION:
;servidor.pd.                   IN      ANY

;; ANSWER SECTION:
servidor.pd.            86400   IN      A       198.51.100.1
servidor.pd.            86400   IN      AAAA    2012:2::6:12

;; AUTHORITY SECTION:
pd.                     86400   IN      NS      ns.pd.

;; ADDITIONAL SECTION:
ns.pd.                  86400   IN      A       192.0.2.2
ns.pd.                  86400   IN      AAAA    2001:db8::2

;; Query time: 2 msec
;; SERVER: 2001:db8:b1c0::1#53(2001:db8:b1c0::1)
;; WHEN: Sat Mar 31 02:30:06 2012
;; MSG SIZE  rcvd: 134

bash-4.2#
```

CORE: cliente (console)

```
bash-4.2# dig -t any servidor.v4

; <<>> DiG 9.8.1-P1 <<>> -t any servidor.v4
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 20879
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; QUESTION SECTION:
;servidor.v4.                   IN      ANY

;; ANSWER SECTION:
servidor.v4.            86400   IN      A       203.0.113.2

;; AUTHORITY SECTION:
v4.                     86400   IN      NS      ns.v4.

;; ADDITIONAL SECTION:
ns.v4.                  86400   IN      A       192.0.2.2
ns.v4.                  86400   IN      AAAA    2001:db8::2

;; Query time: 1 msec
;; SERVER: 2001:db8:b1c0::1#53(2001:db8:b1c0::1)
;; WHEN: Sat Mar 31 02:28:58 2012
;; MSG SIZE  rcvd: 106

bash-4.2#
```

Nestas consultas, podemos observar que apenas o domínio servidor.pd possui endereçamento IPv4 e IPv6 (pilha dupla) e que o servidor.v4 possui apenas endereço IPv4.

9. Neste passo, modificaremos a configuração de DNS para habilitar o DNS64.
   a. Abra o terminal da máquina virtual e utilize o seguinte comando para encerrar o serviço:

```
# sudo killall named
```

O resultado deve ser:

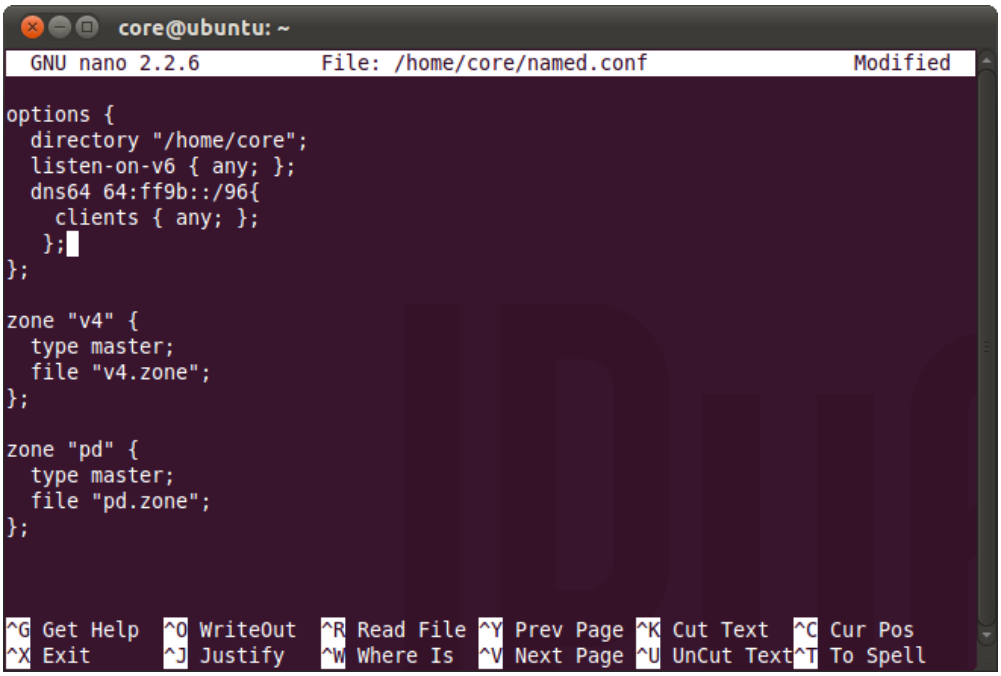
   b. Modifique o arquivo de configuração do DNS através do seguinte comando:

```
# nano /home/core/named.conf
```

Insira o texto destacado conforme abaixo:

```
options {
  directory "/home/core/";
  listen-on-v6 { any; };
  dns64 64:ff9b::/96 {
    clients { any; };
  };
};

zone "v4" {
  type master;
  file "v4.zone";
};

zone "pd" {
  type master;
  file "pd.zone";
};
```

O resultado deve ser:

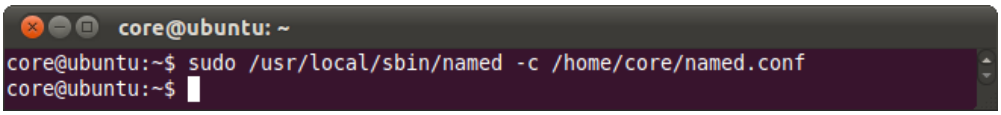


Aperte Ctrl+X para sair do nano e 'Y' para confirmar a modificação do arquivo.

c. Abra o terminal da máquina virtual e utilize o seguinte comando para inicializar o serviço DNS:

```
# sudo /usr/local/sbin/named -c /home/core/named.conf
```

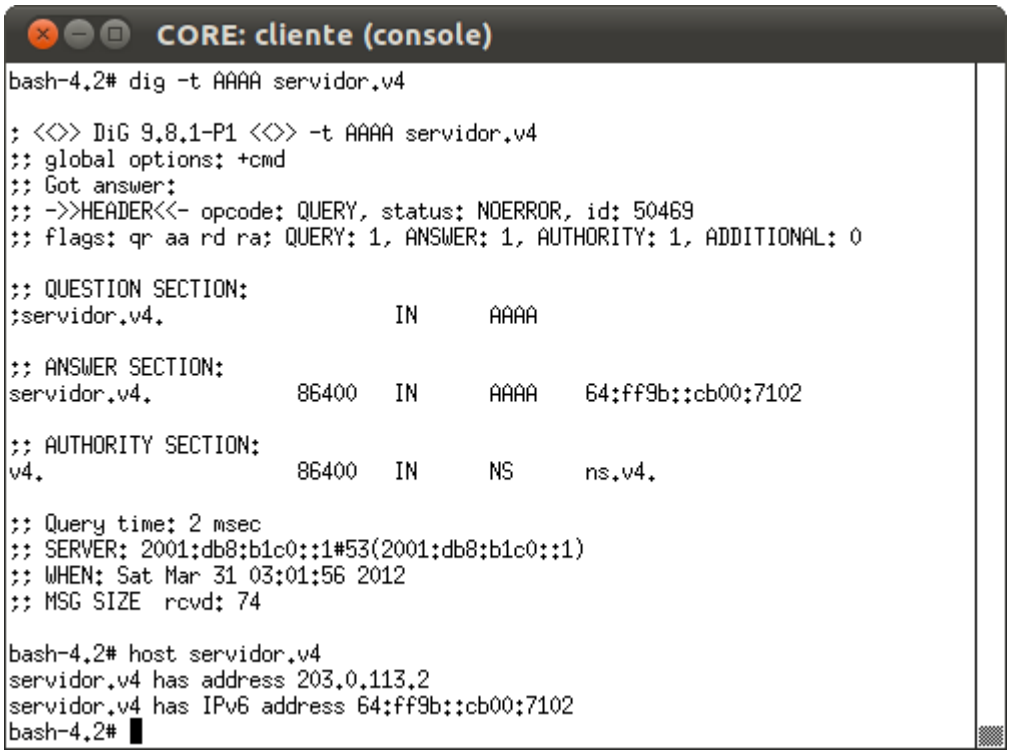
O resultado deve ser:



d. Abra o terminal de cliente através do duplo-clique e utilize os seguintes comandos:

```
# dig -t AAAA servidor.v4
# host servidor.v4
```

O resultado deve ser:

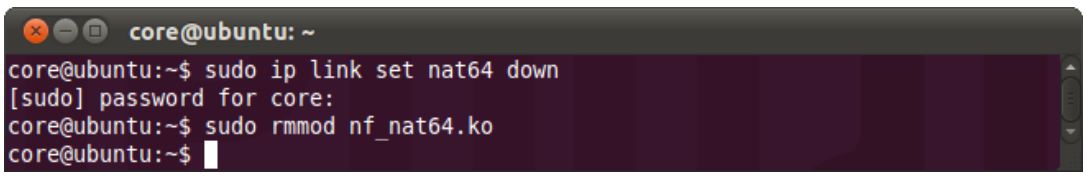


Nestas consultas, podemos observar que a consulta ao domínio servidor.v4 agora possui como resposta um endereço IPv4 mapeado.

10. Antes de encerrar a simulação, abra o terminal da máquina virtual e utilize os seguintes comandos:

```
# sudo ip link set nat64 down
# sudo rmmod nf_nat64.ko
```

Caso necessário, digite “core” para prosseguir.



Esses comandos desabilitam a interface do NAT64 na máquina virtual e o módulo necessário para sua execução.

11. Encerre a simulação:

a. aperte o botão  ; ou
b. utilize o menu Experiment > Stop.