

Frame-Relay

CISCO ACADEMY - NCE/UFRJ

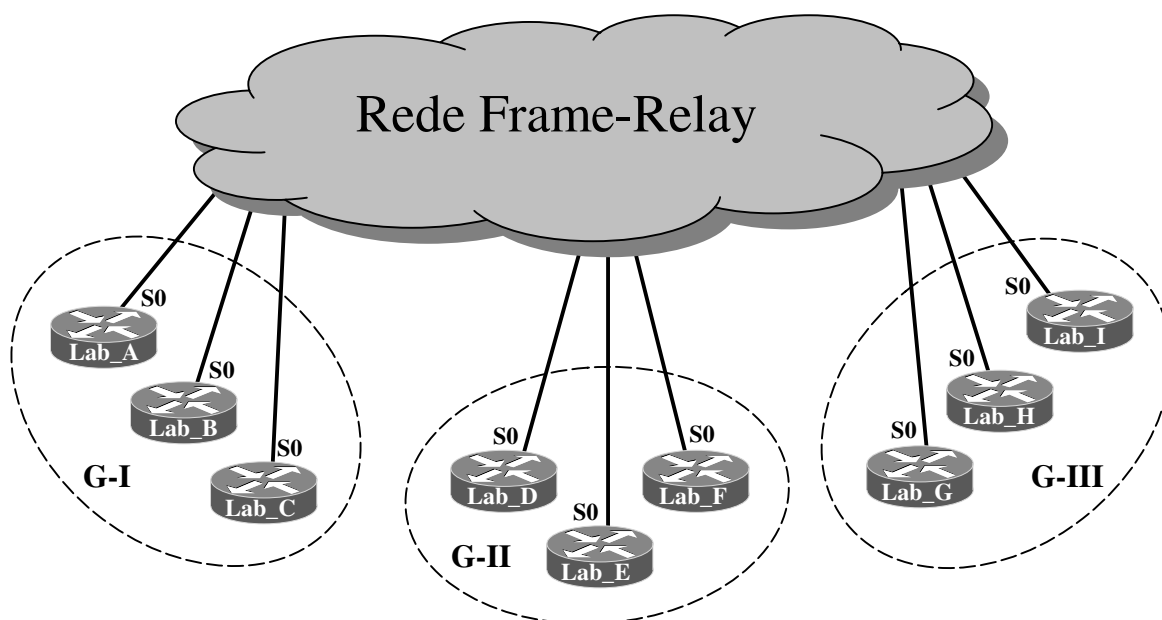


Objetivos

Familiarizar com os procedimentos para efetuar a configuração para ligar roteadores Cisco em uma rede Frame-Relay.

Montagem Inicial

Os 9 roteadores serão divididos em três grupos de três roteadores cada. Cada roteador utilizará apenas as interfaces Ethernet 0 e a Serial 0 e se ligarão à rede Frame-Relay (nuvem) do laboratório Cisco conforme o esquema abaixo.



Cada roteador se comunicará, inicialmente, apenas com os roteadores do seu grupo, e os DLCIs a serem usados são os mesmos já divulgados.

Grupo	Roteadores
I	Lab_A, Lab_B e Lab_C
II	Lab_D, Lab_E e Lab_F
III	Lab_G, Lab_H e Lab_I

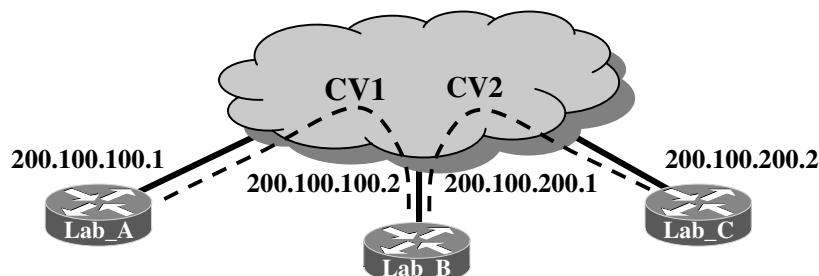
Configuração Inicial

Os endereços IP da interface Ethernet de cada roteador são os mesmos utilizados na topologia do laboratório Cisco original:

Roteador	IP da E0	Roteador	IP da E0	Roteador	IP da E0
Lab_A	220.205.75.0	Lab_D	219.17.100.0	Lab_G	210.78.212.0
Lab_B	205.7.5.0	Lab_E	195.109.140.0	Lab_H	219.54.60.0
Lab_C	192.5.5.0	Lab_F	221.157.189.0	Lab_I	199.64.3.0

Exercício 1 – Conexões Ponto-a-Ponto Hub-and-Spoke

Cada grupo será ligado conforme a figura abaixo e apenas dois circuitos virtuais foram contratados na Concessionária de Serviços: um que liga o primeiro roteador ao segundo (CV1) e outro que liga o segundo roteador ao terceiro (CV2).



Configure os roteadores utilizando os seguintes endereços IP para cada circuito virtual contratado:

Grupo	CV1	IP	CV2	IP
I	Lab_A – Lab_B	200.100.100.0	Lab_B – Lab_C	200.100.200.0
II	Lab_D – Lab_E	200.150.100.0	Lab_E – Lab_F	200.150.200.0
III	Lab_G – Lab_H	200.200.100.0	Lab_H – Lab_I	200.200.200.0

Como o roteador central tem 2 circuitos virtuais, será necessário usar subinterface para configurar a interface S0. Na configuração dos outros roteadores pode, ou não, ser usada a subinterface.

1. Programações Usando Interface na Serial 0 de Lab_A e Lab_C

a. Programação de Lab_A

```
Lab_A# config terminal
Lab_A(config)# interface serial 0
Lab_A(config-if)# description CV1 para Lab_B
Lab_A(config-if)# encapsulation frame-relay
Lab_A(config-if)# ip address 200.100.100.1 255.255.255.0
Lab_A(config-if)# frame-relay interface-dlci 120
Lab_A(config-fr-dlci)# end
```

b. Programação de Lab_B

```
Lab_B# config terminal
Lab_B(config)# interface serial 0
Lab_B(config-if)# encapsulation frame-relay
Lab_B(config-if)# no ip address
Lab_B(config-if)# interface serial 0.1 point-to-point
Lab_B(config-subif)# description CV1 para Lab_A
Lab_B(config-subif)# ip address 200.100.100.2 255.255.255.0
Lab_B(config-subif)# frame-relay interface-dlci 210
Lab_B(config-fr-dlci)# interface serial 0.2 point-to-point
Lab_B(config-subif)# description CV2 para Lab_C
Lab_B(config-subif)# ip address 200.100.200.1 255.255.255.0
Lab_B(config-subif)# frame-relay interface-dlci 230
Lab_B(config-fr-dlci)# end
```

c. Programação de Lab_C

```
Lab_C# config terminal
Lab_C(config)# interface serial 0
Lab_C(config-if)# description CV2 para Lab_B
Lab_C(config-if)# encapsulation frame-relay
Lab_C(config-if)# ip address 200.100.200.2 255.255.255.0
Lab_C(config-if)# frame-relay interface-dlci 320
Lab_C(config-fr-dlci)# end
```

d. Verificando os Circuitos Virtuais

Verifique o mapeamento IP x DLCI entrando com o comando **show frame-relay map**.

```
Lab_A# sh frame-relay map
Serial0 (up): ip 200.100.100.2 dlci 120(0x78,0x1C80), dynamic,
             broadcast,, status defined, active
Serial0 (up): ip 200.100.200.2 dlci 130(0x82,0x2020), dynamic,
             broadcast,, status defined, active
```

```
Lab_B# sh frame-relay map
Serial0.1 (up): point-to-point dlci, dlci 210(0xD2,0x3420), broadcast
               status defined, active
Serial0.2 (up): point-to-point dlci, dlci 230(0xE6,0x3860), broadcast
               status defined, active
```

```
Lab_C# sh frame-relay map
Serial0 (up): ip 200.100.100.1 dlci 310(0x136,0x4C60), dynamic,
             broadcast,, status defined, active
Serial0 (up): ip 200.100.200.1 dlci 320(0x140,0x5000), dynamic,
             broadcast,, status defined, active
```

Observe que o PVC entre Lab_A e Lab_C está ativo mesmo não estando definido na programação. Isso ocorre porque os DLCIs 130 (em Lab_A) e 310 (em Lab_C) estão definidos na nuvem Frame-Relay e são informados pelo switch através das LMIs. Através do *Inverse ARP* o IP da interface do outro lado é descoberto.

Da mesma forma, como os roteadores dos outros grupos também estão programados, outros DLCIs devem aparecer (isso irá depender do estado em que estão as programações dos outros roteadores).

Dessa forma a configuração dos 3 roteadores (ou eventualmente dos 9 roteadores) seria *full-mesh* (totalmente conectada) e não *hub-and-spoke*. Para que o laboratório funcione como foi definido é necessário excluir da nuvem Frame-Relay todos os DLCIs não utilizados pelos 3 grupos.

Quando usamos subinterfaces, como em Lab_B, não se atribui endereço IP à interface. Assim, os DLCIs não utilizados ficam associados à interface (sem endereço IP), de forma que o *Inverse ARP* não terá sucesso nesses DLCIs.

Através do comando **show frame-relay pvc** o estado de cada DLCI pode ser verificado. Executando esse comando em Lab_B, observe que os DLCIs 210 e 230 estão ativos e

associados às subinterfaces (Serial0.1 e Serial0.2) e os demais DLCIs estão inativos e associados à interface (Serial0).

```
Lab_B# sh frame-relay pvc
```

```
PVC Statistics for interface Serial0 (Frame Relay DTE)
```

```
DLCI=210, DLCI USAGE=LOCAL, PVC STATUS=ACTIVE, INTERFACE=Serial0.1
```

```
input pkts 18      output pkts 10      in bytes 1154
out bytes 1563     dropped pkts 0      in FECN pkts 0
in BECN pkts 0    out FECN pkts 0    out BECN pkts 0
in DE pkts 0      out DE pkts 0
out bcast pkts 10 out bcast bytes 1563
pvc create time 00:20:31, last time pvc status changed 00:20:22
```

```
DLCI=230, DLCI USAGE=LOCAL, PVC STATUS=ACTIVE, INTERFACE=Serial0.2
```

```
input pkts 97      output pkts 7       in bytes 30090
out bytes 1241     dropped pkts 0      in FECN pkts 0
in BECN pkts 0    out FECN pkts 0    out BECN pkts 0
in DE pkts 0      out DE pkts 0
out bcast pkts 7   out bcast bytes 1241
pvc create time 00:20:33, last time pvc status changed 00:11:13
```

```
DLCI=240, DLCI USAGE=UNUSED, PVC STATUS=ACTIVE, INTERFACE=Serial0
```

```
input pkts 0      output pkts 0       in bytes 0
out bytes 0       dropped pkts 0      in FECN pkts 0
in BECN pkts 0    out FECN pkts 0    out BECN pkts 0
in DE pkts 0      out DE pkts 0
out bcast pkts 0  out bcast bytes 0   Num Pkts Switched 0
pvc create time 00:20:34, last time pvc status changed 00:20:24
```

```
DLCI=250, DLCI USAGE=UNUSED, PVC STATUS=ACTIVE, INTERFACE=Serial0
```

```
input pkts 0      output pkts 0       in bytes 0
out bytes 0       dropped pkts 0      in FECN pkts 0
in BECN pkts 0    out FECN pkts 0    out BECN pkts 0
in DE pkts 0      out DE pkts 0
out bcast pkts 0  out bcast bytes 0   Num Pkts Switched 0
pvc create time 00:20:35, last time pvc status changed 00:20:25
```

```
DLCI=260, DLCI USAGE=UNUSED, PVC STATUS=INACTIVE, INTERFACE=Serial0
```

```
input pkts 0      output pkts 0       in bytes 0
out bytes 0       dropped pkts 0      in FECN pkts 0
in BECN pkts 0    out FECN pkts 0    out BECN pkts 0
in DE pkts 0      out DE pkts 0
out bcast pkts 0  out bcast bytes 0   Num Pkts Switched 0
pvc create time 00:20:36, last time pvc status changed 00:20:36
```

```
DLCI=270, DLCI USAGE=UNUSED, PVC STATUS=ACTIVE, INTERFACE=Serial0
```

```
input pkts 21      output pkts 0       in bytes 630
out bytes 0       dropped pkts 0      in FECN pkts 0
in BECN pkts 0    out FECN pkts 0    out BECN pkts 0
in DE pkts 0      out DE pkts 0
out bcast pkts 0  out bcast bytes 0   Num Pkts Switched 0
pvc create time 00:20:37, last time pvc status changed 00:20:27
```

```

DLCI=280, DLCI USAGE=UNUSED, PVC STATUS=ACTIVE, INTERFACE=Serial0
input pkts 0      output pkts 0      in bytes 0
out bytes 0      dropped pkts 0    in FECN pkts 0
in BECN pkts 0   out FECN pkts 0   out BECN pkts 0
in DE pkts 0     out DE pkts 0
out bcast pkts 0 out bcast bytes 0  Num Pkts Switched 0
pvc create time 00:20:38, last time pvc status changed 00:20:28

```

```

DLCI=290, DLCI USAGE=UNUSED, PVC STATUS=ACTIVE, INTERFACE=Serial0
input pkts 23     output pkts 0      in bytes 690
out bytes 0      dropped pkts 0     in FECN pkts 0
in BECN pkts 0   out FECN pkts 0   out BECN pkts 0
in DE pkts 0     out DE pkts 0
out bcast pkts 0 out bcast bytes 0  Num Pkts Switched 0
pvc create time 00:20:39, last time pvc status changed 00:20:29

```

Lab_B#

Assim, para não alterar a nuvem Frame-Relay, é preferível usar a solução através de subinterfaces (item 2, a seguir).

e. Comando Ping na Própria Interface Frame-Relay

Outro problema da solução através de interfaces é a impossibilidade de executar o comando **ping para o endereço IP da própria interface serial**, que é uma limitação da Cisco. Esse problema não ocorre quando subinterfaces são usadas.

```

Lab_A# ping 200.100.100.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)

```

← ping na própria interface

```

Lab_A# ping 200.100.100.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.2, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/68/68 ms

```

← ping na interface do vizinho

```

Lab_C# ping 200.100.200.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.200.2, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)

```

← ping na própria interface

```

Lab_C# ping 200.100.200.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.200.1, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 44/55/100 ms

```

← ping na interface do vizinho

Lab_B# ping 200.100.100.2 Type escape sequence to abort. Sending 5, 100-byte ICMP Echos to 200.100.100.2, timeout is 2 seconds: !!!!!! Success rate is 100 percent (5/5), round-trip min/avg/max = 128/131/140 ms	← ping na própria subinterface
Lab_B# ping 200.100.100.1 Type escape sequence to abort. Sending 5, 100-byte ICMP Echos to 200.100.100.1, timeout is 2 seconds: !!!!!! Success rate is 100 percent (5/5), round-trip min/avg/max = 68/73/96 ms	← ping na interface do vizinho
Lab_B# ping 200.100.200.1 Type escape sequence to abort. Sending 5, 100-byte ICMP Echos to 200.100.200.1, timeout is 2 seconds: !!!!!! Success rate is 100 percent (5/5), round-trip min/avg/max = 84/86/96 ms	← ping na própria subinterface
Lab_B# ping 200.100.200.2 Type escape sequence to abort. Sending 5, 100-byte ICMP Echos to 200.100.200.2, timeout is 2 seconds: !!!!!! Success rate is 100 percent (5/5), round-trip min/avg/max = 44/54/96 ms	← ping na interface do vizinho

f. Split Horizon

O *Split Horizon* está automaticamente desativado nas interfaces com encapsulamento Frame-Relay. Nas subinterfaces Frame-Relay ponto-a-ponto e multiponto ele está ativo. No modo EXEC privilegiado, verifique essa situação com o comando ***show ip interface serial 0:***

```
Lab_A#sh ip int s0
Serial0 is up, line protocol is up
 Internet address is 200.100.100.1/24
 Broadcast address is 255.255.255.255
 Address determined by setup command
 MTU is 1500 bytes
 Helper address is not set
 Directed broadcast forwarding is disabled
 Multicast reserved groups joined: 224.0.0.9
 Outgoing access list is not set
 Inbound access list is not set
 Proxy ARP is enabled
 Security level is default
 Split horizon is disabled
 ICMP redirects are always sent
 ICMP unreachable are always sent
 ICMP mask replies are never sent
 IP fast switching is enabled
 IP fast switching on the same interface is enabled
 -----
 (restante omitido)
```

```

Lab_B#sh ip int s0.1
Serial0.1 is up, line protocol is up
Internet address is 200.100.200.2/24
Broadcast address is 255.255.255.255
Address determined by setup command
MTU is 1500 bytes
Helper address is not set
Directed broadcast forwarding is disabled
Multicast reserved groups joined: 224.0.0.9
Outgoing access list is not set
Inbound access list is not set
Proxy ARP is enabled
Security level is default
Split horizon is enabled
ICMP redirects are always sent
ICMP unreachable are always sent
ICMP mask replies are never sent
IP fast switching is enabled
IP fast switching on the same interface is enabled
IP Flow switching is disabled
IP Fast switching turbo vector
IP multicast fast switching is enabled
IP multicast distributed fast switching is disabled
-----
(restante omitido)

```

2. Programações Usando Subinterface na Serial 0 de Lab_A e Lab_C

a. Programação de Lab_A

```

Lab_A# config terminal
Lab_A(config)# interface serial 0
Lab_A(config-if)# encapsulation frame-relay
Lab_A(config-if)# no ip address
Lab_A(config-if)# interface serial 0.1 point-to-point
Lab_A(config-subif)# description CV1 para Lab_B
Lab_A(config-subif)# ip address 200.100.100.1 255.255.255.0
Lab_A(config-subif)# frame-relay interface-dlci 120
Lab_A(config-fr-dlci)# end

```

b. Programação de Lab_B

```

Lab_B# config terminal
Lab_B(config)# interface serial 0
Lab_B(config-if)# encapsulation frame-relay
Lab_B(config-if)# no ip address
Lab_B(config-if)# interface serial 0.1 point-to-point
Lab_B(config-subif)# description CV1 para Lab_A
Lab_B(config-subif)# ip address 200.100.100.2 255.255.255.0
Lab_B(config-subif)# frame-relay interface-dlci 210
Lab_B(config-fr-dlci)# interface serial 0.2 point-to-point
Lab_B(config-subif)# description CV2 para Lab_C
Lab_B(config-subif)# ip address 200.100.200.1 255.255.255.0
Lab_B(config-subif)# frame-relay interface-dlci 230
Lab_B(config-fr-dlci)# end

```

c. Programação de Lab_C

```
Lab_C# config terminal
Lab_C(config)# interface serial 0
Lab_C(config-if)# encapsulation frame-relay
Lab_C(config-if)# no ip address
Lab_C(config-if)# interface serial 0.1 point-to-point
Lab_C(config-subif)# description CV2 para Lab_B
Lab_C(config-subif)# ip address 200.100.200.2 255.255.255.0
Lab_C(config-subif)# frame-relay interface-dlci 320
Lab_C(config-fr-dlci)# end
```

d. Verificando os Circuitos Virtuais

Verifique o mapeamento IP x DLCI entrando com o comando **show frame-relay map**.

```
Lab_A# sh frame-relay map
Serial0.1 (up): point-to-point dlci, dlci 120(0x78,0x1C80), broadcast
status defined, active
```

```
Lab_B# sh frame-relay map
Serial0.1 (up): point-to-point dlci, dlci 210(0xD2,0x3420), broadcast
status defined, active
Serial0.2 (up): point-to-point dlci, dlci 230(0xE6,0x3860), broadcast
status defined, active
```

```
Lab_C# sh frame-relay map
Serial0.1 (up): point-to-point dlci, dlci 320(0x140,0x5000), broadcast
status defined, active
```

e. Tabelas de Roteamento

Libere o roteamento RIP nos roteadores e verifique se todos os 3 roteadores possuem rotas para as 3 redes ethernet (uma de cada roteador).

```
Lab_A# show ip route
R 200.100.200.0/24 [120/1] via 200.100.100.2, 00:00:10, Serial0.1
R 205.7.5.0/24 [120/1] via 200.100.100.2, 00:00:10, Serial0.1
C 200.100.100.0/24 is directly connected, Serial0.1
R 192.5.5.0/24 [120/2] via 200.100.100.2, 00:00:10, Serial0.1
C 220.205.75.0/24 is directly connected, Ethernet0
```

```
Lab_B# show ip route
C 200.100.200.0/24 is directly connected, Serial0.2
C 205.7.5.0/24 is directly connected, Ethernet0
C 200.100.100.0/24 is directly connected, Serial0.1
R 192.5.5.0/24 [120/1] via 200.100.200.2, 00:00:07, Serial0.2
R 220.205.75.0/24 [120/1] via 200.100.100.1, 00:00:14, Serial0.1
```



```
Lab_C# show ip route
C 200.100.200.0/24 is directly connected, Serial0.1
R 205.7.5.0/24 [120/1] via 200.100.200.1, 00:00:18, Serial0.1
R 200.100.100.0/24 [120/1] via 200.100.200.1, 00:00:18, Serial0.1
C 192.5.5.0/24 is directly connected, Ethernet0
R 220.205.75.0/24 [120/2] via 200.100.200.1, 00:00:18, Serial0.1
```

f. Comando Ping na Própria Interface Frame-Relay

Como estão sendo usadas subinterfaces o comando **ping para o endereço IP da própria interface serial** deverá funcionar normalmente.

```
Lab_A# ping 200.100.100.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.1, timeout is 2 seconds:
!!!!!!
Success rate is 0 percent (0/5)
```

← ping na própria subinterface

```
Lab_A# ping 200.100.100.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.2, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/68/68 ms
```

← ping na interface do vizinho

```
Lab_B# ping 200.100.100.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.2, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 128/131/140 ms
```

← ping na própria subinterface

```
Lab_B# ping 200.100.100.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.1, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 68/73/96 ms
```

← ping na interface do vizinho

```
Lab_B# ping 200.100.200.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.200.1, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 84/86/96 ms
```

← ping na própria subinterface

```
Lab_B# ping 200.100.200.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.200.2, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 44/54/96 ms
```

← ping na interface do vizinho

```

Lab_C# ping 200.100.200.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.200.2, timeout is 2 seconds:
!!!!
Success rate is 0 percent (0/5)

Lab_C# ping 200.100.200.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.200.1, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 44/55/100 ms

```

← ping na própria subinterface

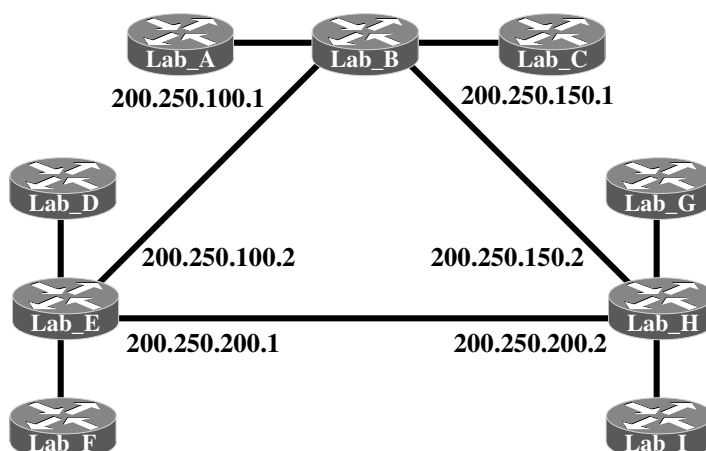
← ping na interface do vizinho

g. Interligando os 3 Grupos (opcional)

Interligue os três grupos que estavam separados através dos roteadores centrais (Lab_B, Lab_E e Lab_H) usando os circuitos virtuais cujos endereços IP estão descritos na tabela a seguir.

Circuito Virtual	Endereço IP
Lab_B – Lab_E	200.250.100.0
Lab_B – Lab_H	200.250.150.0
Lab_E – Lab_H	200.250.200.0

A rede total, com nove roteadores, ficou montada conforme a figura abaixo.



No roteador Lab_B a programação a seguir deve ser acrescentada.

```

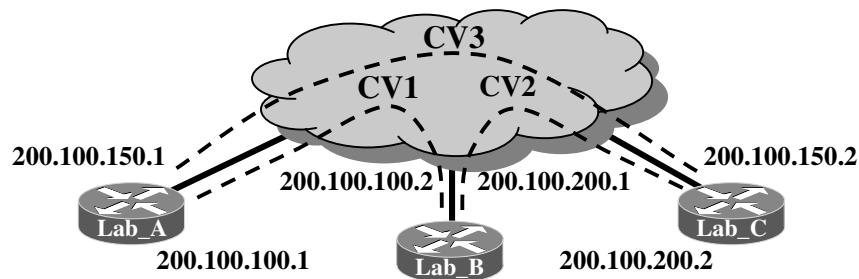
Lab_B# config terminal
Lab_B(config)# interface serial 0.3 point-to-point
Lab_B(config-subif)# description CV para Lab_E
Lab_B(config-subif)# ip address 200.250.100.1 255.255.255.0
Lab_B(config-subif)# frame-relay interface-dlci 250
Lab_B(config-fr-dlci)# interface serial 0.4 point-to-point
Lab_B(config-subif)# description CV para Lab_H
Lab_B(config-subif)# ip address 200.250.150.1 255.255.255.0
Lab_B(config-subif)# frame-relay interface-dlci 280
Lab_B(config-fr-dlci)# end

```

Verifique as tabelas de roteamento para a rede completa.

Exercício 2 – Conexões Ponto-a-Ponto Full-Mesh

Cada grupo será ligado conforme a figura abaixo e três circuitos virtuais foram contratados na Concessionária de Serviços: um que liga o primeiro roteador ao segundo (CV1), outro que liga o segundo roteador ao terceiro (CV2) e, por último um que liga o primeiro roteador ao terceiro (CV3).



Configure os roteadores utilizando os endereços IP para cada circuito virtual contratado de acordo com a tabela a seguir.

Grupo	Circuito Virtual	IP
I	CV1 = Lab_A – Lab_B	200.100.100.0
	CV2 = Lab_B – Lab_C	200.100.200.0
	CV3 = Lab_A – Lab_C	200.100.150.0
II	CV1 = Lab_D – Lab_E	200.150.100.0
	CV2 = Lab_E – Lab_F	200.150.200.0
	CV3 = Lab_D – Lab_F	200.150.150.0
III	CV1 = Lab_G – Lab_H	200.200.100.0
	CV2 = Lab_H – Lab_I	200.200.200.0
	CV3 = Lab_G – Lab_I	200.200.150.0

Como todos os roteadores possuem 2 circuitos virtuais, será necessário usar subinterface para configurar a interface S0 de todos eles.

a. Programação de Lab_A

```
Lab_A# config terminal
Lab_A(config)# interface serial 0
Lab_A(config-if)# encapsulation frame-relay
Lab_A(config-if)# no ip address
Lab_A(config-if)# interface serial 0.1 point-to-point
Lab_A(config-subif)# description CV1 para Lab_B
Lab_A(config-subif)# ip address 200.100.100.1 255.255.255.0
Lab_A(config-subif)# frame-relay interface-dlci 120
Lab_A(config-fr-dlci)# interface serial 0.2 point-to-point
Lab_A(config-subif)# description CV3 para Lab_C
Lab_A(config-subif)# ip address 200.100.150.1 255.255.255.0
Lab_A(config-subif)# frame-relay interface-dlci 130
Lab_A(config-fr-dlci)# end
```

b. Programação de Lab_B

```
Lab_B# config terminal
Lab_B(config)# interface serial 0
Lab_B(config-if)# encapsulation frame-relay
Lab_B(config-if)# no ip address
Lab_B(config-if)# interface serial 0.1 point-to-point
Lab_B(config-subif)# description CV1 para Lab_A
Lab_B(config-subif)# ip address 200.100.100.2 255.255.255.0
Lab_B(config-subif)# frame-relay interface-dlci 210
Lab_B(config-fr-dlci)# interface serial 0.2 point-to-point
Lab_B(config-subif)# description CV2 para Lab_C
Lab_B(config-subif)# ip address 200.100.200.1 255.255.255.0
Lab_B(config-subif)# frame-relay interface-dlci 230
Lab_B(config-fr-dlci)# end
```

c. Programação de Lab_C

```
Lab_C# config terminal
Lab_C(config)# interface serial 0
Lab_C(config-if)# encapsulation frame-relay
Lab_C(config-if)# no ip address
Lab_C(config-if)# interface serial 0.1 point-to-point
Lab_C(config-subif)# description CV2 para Lab_B
Lab_C(config-subif)# ip address 200.100.200.2 255.255.255.0
Lab_C(config-subif)# frame-relay interface-dlci 320
Lab_C(config-fr-dlci)# interface serial 0.2 point-to-point
Lab_C(config-subif)# description CV3 para Lab_A
Lab_C(config-subif)# ip address 200.100.150.2 255.255.255.0
Lab_C(config-subif)# frame-relay interface-dlci 310
Lab_C(config-fr-dlci)# end
```

d. Tabelas de Roteamento

Libere o roteamento RIP nos roteadores e verifique se os 3 roteadores possuem rotas para as redes ethernet dos demais. Como a rede permite caminhos alternativos, na tabela de roteamento de cada roteador existe uma rede com 2 rotas de mesma métrica.

```
Lab_A# show ip route
R 200.100.200.0/24 [120/1] via 200.100.100.2, 00:00:02, Serial0.1
   [120/1] via 200.100.150.2, 00:00:25, Serial0.2
R 205.7.5.0/24 [120/1] via 200.100.100.2, 00:00:02, Serial0.1
C 200.100.100.0/24 is directly connected, Serial0.1
R 192.5.5.0/24 [120/1] via 200.100.150.2, 00:00:25, Serial0.2
C 200.100.150.0/24 is directly connected, Serial0.2
C 220.205.75.0/24 is directly connected, Ethernet0
```

```
Lab_B# show ip route
C 200.100.200.0/24 is directly connected, Serial0.2
C 205.7.5.0/24 is directly connected, Ethernet0
C 200.100.100.0/24 is directly connected, Serial0.1
R 192.5.5.0/24 [120/1] via 200.100.200.2, 00:00:28, Serial0.2
R 200.100.150.0/24 [120/1] via 200.100.100.1, 00:00:03, Serial0.1
   [120/1] via 200.100.200.2, 00:00:28, Serial0.2
R 220.205.75.0/24 [120/1] via 200.100.100.1, 00:00:03, Serial0.1
```

```
Lab_C# show ip route
C 200.100.200.0/24 is directly connected, Serial0.1
R 205.7.5.0/24 [120/1] via 200.100.200.1, 00:00:10, Serial0.1
R 200.100.100.0/24 [120/1] via 200.100.200.1, 00:00:10, Serial0.1
    [120/1] via 200.100.150.1, 00:00:14, Serial0.2
C 192.5.5.0/24 is directly connected, Ethernet0
C 200.100.150.0/24 is directly connected, Serial0.2
R 220.205.75.0/24 [120/1] via 200.100.150.1, 00:00:14, Serial0.2
```

e. Verificando os Circuitos Virtuais

Verifique o mapeamento IP x DLCI entrando com o comando ***show frame-relay map***.

```
Lab_A# sh frame-relay map
Serial0.1 (up): point-to-point dlci, dlci 120(0x78,0x1C80), broadcast
    status defined, active
Serial0.2 (up): point-to-point dlci, dlci 130(0x82,0x2020), broadcast
    status defined, active
```

```
Lab_B# sh frame-relay map
Serial0.1 (up): point-to-point dlci, dlci 210(0xD2,0x3420), broadcast
    status defined, active
Serial0.2 (up): point-to-point dlci, dlci 230(0xE6,0x3860), broadcast
    status defined, active
```

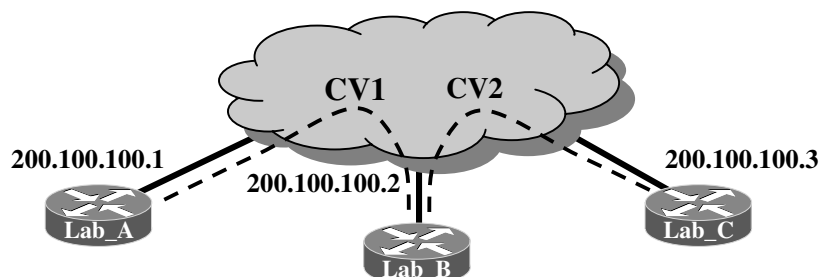
```
Lab_C# sh frame-relay map
Serial0.1 (up): point-to-point dlci, dlci 320(0x140,0x5000), broadcast
    status defined, active
Serial0.2 (up): point-to-point dlci, dlci 310(0x136,0x4C60), broadcast
    status defined, active
```

f. Comando Ping na Própria Interface Frame-Relay

Como estão sendo usadas subinterfaces o comando **ping para o endereço IP da própria interface serial** deverá funcionar normalmente.

Exercício 3 – Conexões Multiponto

Cada grupo será ligado conforme a figura abaixo e apenas dois circuitos virtuais foram contratados na Concessionária de Serviços: um que liga o primeiro roteador ao segundo (CV1) e outro que liga o segundo roteador ao terceiro (CV2). Os 3 roteadores devem estar na mesma sub-rede, implementando, assim, uma conexão multiponto.



Configure os roteadores utilizando os seguintes endereços IP para cada sub-rede:

Grupo	IP
I	200.100.100.0
II	200.150.100.0
III	200.200.100.0

Como o roteador central tem um circuito virtual multiponto o uso de subinterface para configurar a interface S0 é obrigatório. Na configuração dos outros roteadores pode ser usada interface ou subinterface.

1. Programações Usando Interface na Serial 0 de Lab_A e Lab_C

Mesmo que todas as interfaces dos 3 roteadores estejam na mesma rede IP (no caso 200.100.100.0), como a interface serial é ponto-a-ponto, Lab_A não sabe qual DLCI deve ser usado para chegar em Lab_C (ele só sabe qual usar para chegar em Lab_B). O mesmo se aplica para Lab_C chegar em Lab_A. Isso não ocorre quando se usa subinterfaces pois elas permitem broadcast por default.

Esse problema é solucionado usando-se o comando **frame-relay map** de forma a se criar uma entrada estática na tabela IP x DLCI. Esse comando deve ser dado em Lab_A e Lab_C. A programação de Lab_B não se altera.

Para evitar os problemas encontrados quando são usadas interfaces (aparecimento de outros DLCIs existentes na nuvem que forma o laboratório e o não funcionamento do comando ping na própria interface) deve-se usar apenas subinterfaces.

a. Programação de Lab_A

```
Lab_A# config terminal
Lab_A(config)# interface serial 0
Lab_A(config-if)# shutdown
Lab_A(config-if)# description CV1 para Lab_B
Lab_A(config-if)# encapsulation frame-relay
Lab_A(config-if)# ip address 200.100.100.1 255.255.255.0
Lab_A(config-if)# frame-relay interface-dlci 120
Lab_A(config-fr-dlci)# frame-relay map ip 200.100.100.3 120 broadcast
Lab_A(config-if)# no shutdown
Lab_A(config-if)# end
```

Cuidado: Os DLCIs entre Lab_A e Lab_C (130 e 310) estão definidos na nuvem Frame-Relay e não podem ser eliminados. Assim, ao se programar o segundo roteador (por exemplo Lab_C programado depois de Lab_A) o *Inverse ARP* em Lab_C irá automaticamente, após programado o encapsulamento Frame-Relay, obter o IP de Lab_A (200.100.100.1) pelo DLCI 310 e colocá-lo na tabela IP x DLCI. Assim, o comando **frame-relay map ip 200.100.100.1 320 broadcast**, a ser dado em Lab_C, dará erro pois esse endereço IP já estará na tabela como dinâmico. Para resolver esse problema coloque a interface serial 0 em shutdown antes de programá-la (isso pode ser feito em Lab_A e Lab_C).

b. Programação de Lab_B

Nos exercícios anteriores as subinterfaces S0.1 e S0.2 foram usadas como tipo ponto-a-ponto. Para usar uma dessas subinterfaces como tipo multiponto seria necessário excluir essas subinterfaces (comando **no interface s0.1** e **no interface s0.2**) e dar um **reload** no roteador Lab_B. Isso ocorre porque o IOS não permite trocar o tipo de uma subinterface. Para evitar isso será usada uma subinterface nova.

```
Lab_B# config terminal
Lab_B(config)# interface serial 0
Lab_B(config-if)# encapsulation frame-relay
Lab_B(config-if)# no ip address
Lab_B(config-if)# interface serial 0.3 multipoint
Lab_B(config-subif)# description CV multiponto para Lab_A e Lab_C
Lab_B(config-subif)# ip address 200.100.100.2 255.255.255.0
Lab_B(config-subif)# frame-relay interface-dlci 210
Lab_B(config-fr-dlci)# frame-relay interface-dlci 230
Lab_B(config-fr-dlci)# end
```

c. Programação de Lab_C

```
Lab_C# config terminal
Lab_C(config)# interface serial 0
Lab_C(config-if)# shutdown
Lab_C(config-if)# description CV2 para Lab_B
Lab_C(config-if)# encapsulation frame-relay
Lab_C(config-if)# ip address 200.100.100.3 255.255.255.0
Lab_C(config-if)# frame-relay interface-dlci 320
Lab_C(config-fr-dlci)# frame-relay map ip 200.100.100.1 320 broadcast
Lab_C(config-if)# no shutdown
Lab_C(config-if)# end
```

d. Verificando os Circuitos Virtuais

Verifique o mapeamento IP x DLCI entrando com o comando **show frame-relay map**. Para esse laboratório funcionar é necessário excluir da nuvem Frame-Relay todos os circuitos virtuais não utilizados pelos 3 grupos.

```
Lab_A# sh frame-relay map
Serial0 (up): ip 200.100.100.2 dlci 120(0x78,0x1C80), dynamic,
              broadcast,, status defined, active
Serial0 (up): ip 200.100.100.3 dlci 120(0x78,0x1C80), static,
              broadcast,
              CISCO, status defined, active
```

```
Lab_B# sh frame-relay map
Serial0.3 (up): ip 200.100.100.1 dlci 210(0xD2,0x3420), dynamic,
                broadcast,, status defined, active
Serial0.3 (up): ip 200.100.100.3 dlci 230(0xE6,0x3860), dynamic,
                broadcast,, status defined, active
```

```
Lab_C# sh frame-relay map
Serial0 (up): ip 200.100.100.1 dlci 320(0x140,0x5000), static,
                broadcast,
                CISCO, status defined, active
Serial0 (up): ip 200.100.100.2 dlci 320(0x140,0x5000), dynamic,
                broadcast,, status defined, active
```

e. Comando Ping na Própria Interface Frame-Relay

O comando ping na própria interface Frame-Relay não funciona em Lab_A e Lab_C porque está sendo usada interface (e não subinterface). No caso de Lab_B, o ping na própria subinterface Frame-Relay tipo multiponto não funciona, conforme definido no IOS da Cisco. Isso pode ser visto no artigo técnico da Cisco “*Frame Relay Characteristics*” que está disponível link abaixo:

http://www.cisco.com/en/US/tech/tk713/tk237/technologies_tech_note09186a00800942ab.shtml

```
Lab_A# ping 200.100.100.1                                ← ping na própria interface
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)

Lab_A# ping 200.100.100.2                                ← ping na interface de Lab_B
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.2, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 64/65/68 ms

Lab_A# ping 200.100.100.3                                ← ping na interface de Lab_C
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.3, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 108/108/108 ms
```

```
Lab_B# ping 200.100.100.1                                ← ping na interface de Lab_A
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.1, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 64/64/64 ms

Lab_B# ping 200.100.100.2                                ← ping na própria subinterface
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.2, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```



```
Lab_B# ping 200.100.100.3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.3, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 44/44/44 ms
```

← ping na interface de Lab_C

```
Lab_C# ping 200.100.100.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.1, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 108/110/120 ms
```

← ping na interface de Lab_A

```
Lab_C# ping 200.100.100.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.2, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 44/44/44 ms
```

← ping na interface de Lab_B

```
Lab_C# ping 200.100.100.3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.3, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```

← ping na própria interface

2. Programações Usando Subinterface na Serial 0 de Lab_A e Lab_C

a. Programação de Lab_A

Como está sendo usada subinterface, que é broadcast, o comando **frame-relay map** e dar shutdown na interface não são necessários.

```
Lab_A# config terminal
Lab_A(config)# interface serial 0
Lab_A(config-if)# encapsulation frame-relay
Lab_A(config-if)# no ip address
Lab_A(config-if)# interface serial 0.1 point-to-point
Lab_A(config-subif)# description CV1 para Lab_B
Lab_A(config-subif)# ip address 200.100.100.1 255.255.255.0
Lab_A(config-subif)# frame-relay interface-dlci 120
Lab_A(config-fr-dlci)# end
```

b. Programação de Lab_B

```
Lab_B# config terminal
Lab_B(config)# interface serial 0
Lab_B(config-if)# encapsulation frame-relay
Lab_B(config-if)# no ip address
Lab_B(config-if)# interface serial 0.3 multipoint
Lab_B(config-subif)# description CV multiponto para Lab_A e Lab_C
Lab_B(config-subif)# ip address 200.100.100.2 255.255.255.0
Lab_B(config-subif)# frame-relay interface-dlci 210
Lab_B(config-fr-dlci)# frame-relay interface-dlci 230
Lab_B(config-fr-dlci)# end
```

c. Programação de Lab_C

Como está sendo usada subinterface, o comando **frame-relay map** não é necessário.

```
Lab_C# config terminal
Lab_C(config)# interface serial 0
Lab_C(config-if)# encapsulation frame-relay
Lab_C(config-if)# no ip address
Lab_C(config-if)# interface serial 0.1 point-to-point
Lab_C(config-subif)# description CV2 para Lab_B
Lab_C(config-subif)# ip address 200.100.100.3 255.255.255.0
Lab_C(config-subif)# frame-relay interface-dlci 320
Lab_C(config-fr-dlci)# end
```

d. Tabelas de Roteamento

Ative o roteamento RIP nos roteadores e verifique as tabelas de roteamento.

```
Lab_A# show ip route
R 205.7.5.0/24 [120/1] via 200.100.100.2, 00:00:02, Serial0.1
C 200.100.100.0/24 is directly connected, Serial0.1
C 220.205.75.0/24 is directly connected, Ethernet0
```

```
Lab_B# show ip route
C 205.7.5.0/24 is directly connected, Ethernet0
C 200.100.100.0/24 is directly connected, Serial0.3
R 192.5.5.0/24 [120/1] via 200.100.100.3, 00:00:28, Serial0.3
R 220.205.75.0/24 [120/1] via 200.100.100.1, 00:00:05, Serial0.3
```

```
Lab_C# show ip route
R 205.7.5.0/24 [120/1] via 200.100.100.2, 00:00:08, Serial0.1
C 200.100.100.0/24 is directly connected, Serial0.1
C 192.5.5.0/24 is directly connected, Ethernet0
```

Observe que o roteador Lab_A está sem rota para a ethernet de Lab_C e vice-versa, embora Lab_B possua todas as rotas. Isso ocorre porque na subinterface Frame-Relay multiponto existente em Lab_B o *split horizon* está ativo por default. Assim Lab_B não divulga pela interface serial 0 as rotas aprendidas por essa interface.

Desative o *split horizon* (comando **no ip split-horizon**) na subinterface de Lab_B (esse comando não irá funcionar se dado na interface) e verifique as novas tabelas de roteamento.

```
Lab_B# config terminal
Lab_B(config-if)# interface serial 0.3
Lab_B(config-subif)# no ip split-horizon
Lab_B(config-subif)# end
```

Como a subinterface serial 0.3 já está criada não é necessário colocar o tipo (multipoint) ao fazer a programação.

```
Lab_A# show ip route
R 205.7.5.0/24 [120/1] via 200.100.100.2, 00:00:15, Serial0.1
C 200.100.100.0/24 is directly connected, Serial0.1
R 192.5.5.0/24 [120/2] via 200.100.100.2, 00:00:15, Serial0.1
C 220.205.75.0/24 is directly connected, Ethernet0
```

```
Lab_B# show ip route
C 205.7.5.0/24 is directly connected, Ethernet0
C 200.100.100.0/24 is directly connected, Serial0.3
R 192.5.5.0/24 [120/1] via 200.100.100.3, 00:00:28, Serial0.3
R 220.205.75.0/24 [120/1] via 200.100.100.1, 00:00:05, Serial0.3
```

```
Lab_C# show ip route
R 205.7.5.0/24 [120/1] via 200.100.100.2, 00:00:07, Serial0.1
C 200.100.100.0/24 is directly connected, Serial0.1
C 192.5.5.0/24 is directly connected, Ethernet0
R 220.205.75.0/24 [120/2] via 200.100.100.2, 00:00:08, Serial0.1
```

e. Verificando os Circuitos Virtuais

Verifique o mapeamento IP x DLCI entrando com o comando ***show frame-relay map***.

```
Lab_A# sh frame-relay map
Serial0.1 (up): point-to-point dlci, dlci 120(0x78,0x1C80), broadcast
status defined, active
```

```
Lab_B# sh frame-relay map
Serial0.3 (up): ip 200.100.100.1 dlci 210(0xD2,0x3420), dynamic,
broadcast,, status defined, active
Serial0.3 (up): ip 200.100.100.3 dlci 230(0xE6,0x3860), dynamic,
broadcast,, status defined, active
```

```
Lab_C# sh frame-relay map
Serial0.1 (up): point-to-point dlci, dlci 320(0x140,0x5000), broadcast
status defined, active
```

Como o tipo da interface em Lab_B é multiponto, não é possível saber em Lab_A e em Lab_C qual o endereço IP que está associado à interface de Lab_B do outro lado do circuito virtual.

f. Comando Ping na Própria Interface Frame-Relay

No caso de Lab_B, o ping na própria subinterface Frame-Relay tipo multiponto não funciona conforme definido no IOS da Cisco. Isso pode ser visto no artigo técnico da Cisco “*Frame Relay Characteristics*”.

Lab_A# ping 200.100.100.1 ← ping na própria subinterface
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.1, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 128/132/140 ms

Lab_A# ping 200.100.100.2 ← ping na interface de Lab_B
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.2, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 64/64/64 ms

Lab_A# ping 200.100.100.3 ← ping na interface de Lab_C
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.3, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 104/109/120 ms

Lab_B# ping 200.100.100.1 ← ping na interface de Lab_A
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.1, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 64/64/64 ms

Lab_B# ping 200.100.100.2 ← ping na própria subinterface
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.2, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)

Lab_B# ping 200.100.100.3 ← ping na interface de Lab_C
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.3, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 44/44/44 ms

Lab_C# ping 200.100.100.1 ← ping na interface de Lab_A
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.1, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 108/111/124 ms

Lab_C# ping 200.100.100.2 ← ping na interface de Lab_B
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.2, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 44/44/44 ms

Lab_C# ping 200.100.100.3 ← ping na própria subinterface
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.100.100.3, timeout is 2 seconds:
!!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 84/84/84 ms