Departamento de Engenharia Informática e de Sistemas

Protocolos Spanning Tree

Rafael Tavares Ribeiro - 2019131989

Guião laboratorial no âmbito da Licenciatura de Engenharia Informática, para a unidade curricular de Disponibilidade e Desempenho, lecionada pelo Docente Luís Eduardo Faria dos Santos

dezembro de 2021

Índice

Lista de figuras 3

Abreviaturas 3

1. Introdução 5

2. Estudo dos ciclos 6

3. Protocolo STP 8

3.1. Conteúdo teórico STP 8

3.2. Experiência STP - A 8

3.2.1. Capturas e injeção de falhas 10

3.3. Experiência Convergência STP – AA 14

4. Protocolo RSTP 16

4.1. Conteúdo Teórico RSTP 16

4.2. Experiência RSTP - B 17

4.2.1. Capturas e injeção de falhas 19

5. Protocolo MSTP 22

5.1. Conteúdo Teórico MSTP 22

5.2. Experiência MSTP - C 22

5.2.1. Capturas 23

5.3. Experiência MSTP – CC 27

5.3.1. Injeção de falhas 28

6. PVST+ e Rapid PVST+ 29

7. Conclusões 30

Lista de figuras

Figura 1 - Topologia de estudo de ciclos 6
Figura 2 - Pacotes STP 6
Figura 3 - Desativar STP 6
Figura 4 - Término da comunicação 7
Figura 5 - Ciclos Wireshark 7
Figura 6 - Topologia STP A 9
Figura 7 - Root Bridge A 9
Figura 8 - Captura sw1- sw4 10
Figura 9 - Último pacote antes da falha 11
Figura 10 - Primeiro pacote depois da falha 11
Figura 11 - Exemplo de TCN 12
Figura 12 - Captura de tráfego com a falha 13
Figura 13 - Passagem de estados da e0/0 até RP 13
Figura 14 - Topologia STP AA 14
Figura 15 - TCN do sw2 14
Figura 16 - Topology Change Acknowledgment 15
Figura 17 - Bridge ID sw3 15
Figura 18 - Esquema convergência no RSTP 16
Figura 19 - Topologia RSTP B 17
Figura 20 - Pacotes RSTP 18
Figura 21 - Flags RSTP 18
Figura 22 - Discarding antes da negociação 19
Figura 23 - Sychronization 19
Figura 24 - TCN enviados na falha 20
Figura 25 - TCA enviado pela Root 20
Figura 26 - Flag proposal com "yes" 21
Figura 27 - Continuação do ping após falha 21
Figura 28 - Utilização da porta backup na falha 21
Figura 29 - Topologia MSTP C 23
Figura 30 - Bound (STP) 23
Figura 31 - Exemplo da BPDU MSTP 24
Figura 32 - Exemplo MST Extension 24
Figura 33 - Alteração do Revision Number 25
Figura 34 - Alteração do nome da região 25
Figura 35 - Criação da instância 1 25
Figura 36 - Alteração do Config digest 26
Figura 37 - Análise da instância 0 e 1 26
Figura 38 - Porta e0/1 Root nas 2 instâncias 27
Figura 39 - Topologia MSTP CC 27
Figura 40 - As duas instâncias pelas duas ligações 28
Figura 41 - Injeção da falha e continuação do ping 28

Abreviaturas

BPDU: Bridge Protocol Data Unit

STP: Spanning Tree Protocol

RSTP: Rapid Spanning Tree Protocol

MSTP: Multiple Spanning Tree Protocol

ARP: Address Resolution Protocol

DP: Designated Port

RP: Root Port

TCN: Topology Change Notification

TCA: Topology Change Acknowledgement

CST: Common Spanning Tree

ICMP: Internet Control Message Protocol

1. Introdução

O presente relatório tem o propósito de servir como um guião laboratorial (provisório) do *workshop* realizado à unidade curricular de Disponibilidade e Desempenho. O tema escolhido foi um estudo comparativo entre os diversos protocolos existentes de Spanning Tree e o objetivo deste guião, ainda que provisório, é que sirva de base para que qualquer um consiga replicar as experiências realizadas ao longo do estudo. Para além disto, também serão tiradas conclusões sobre os resultados obtidos. Para que estas experiências sejam possíveis de serem concebidas a partir do acompanhamento deste guião, será necessário possuir as seguintes ferramentas: simulador GNS3 com uma imagem switch como por exemplo a i86bi_linux_l2-adventerprise-ms.high_iron_20170202.bin (ou idêntica) e imagens de terminais como VPCS ou ipterm. Aliada ao GNS3 será também necessário o uso da ferramenta Wireshark para as capturas de tráfego. Com isto, é possível replicar todo o trabalho presente neste guião.

Os protocolos Spanning Tree foram criados quando existiu a necessidade de corrigir os *loops* criados numa rede servida por switchs. Estes ciclos são criados devido à redundância que uma rede necessita de ter para o melhor funcionamento da mesma e impedem que pacotes cheguem a determinados destinos, prejudicando assim a disponibilidade dos serviços prestados pela rede. Existem diversos protocolos: Spanning Tree Protocol (STP), Rapid Spanning Tree Protocol (RSTP) (respetivas alternativas cisco Per Vlan Spanning Tree+ (PVST+) e Rapid PVST+) e ainda o Muliple Spanning Tree Protocol (MSTP). Os mesmos definem uma tabela de *forwarding* nos switches e bridges de modo que, uma rede constituída por estes equipamentos, fique livre dos ciclos.

Neste documento, vão ser relatadas todas as experiências e conclusões tiradas das mesmas. Começando por compreender como os ciclos são realmente criados, passando pelo estudo mais exaustivo de cada um dos protocolos referidos anteriormente e terminando com uma conclusão breve do que se conseguiu retirar dos aspetos estudados. Uma explicação teórica também estará presente em cada um dos conteúdos analisados.

2. Estudo dos ciclos

Não é preciso muito para termos ciclos numa topologia. Com apenas dois switchs ligados entre si a partir de duas ligações, temos um ambiente propício à criação dos *loops* (se não existir nenhum protocolo a correr que evite isso).

Para uma melhor perceção de como são realmente criadas estas menos valias numa rede, foi criada uma topologia simples que segue a ideia referida anteriormente: dois switchs com duas ligações entre si, tal como podemos ver na imagem seguinte.

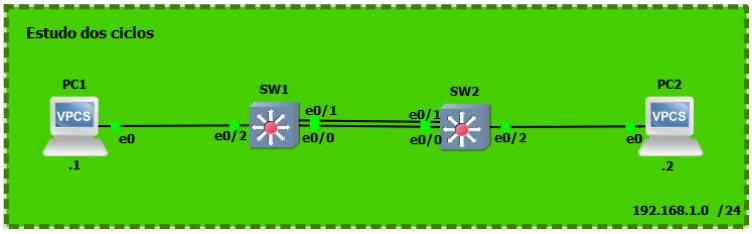


Figura 1 - Topologia de estudo de ciclos

Sabendo que o protocolo que vem administrado por defeito nos switchs é o STP, o mesmo não foi desligado inicialmente. A partir da consola do switch 1 e com a ajuda do comando `show spanning-tree`, conseguimos ver que o sw1 é a Root Bridge da rede. Com o Wireshark, capturou-se tráfego STP nas duas ligações e, como está representado na imagem seguinte, apareceram pacotes STP. Também é possível pingar do PC1 para o PC2.

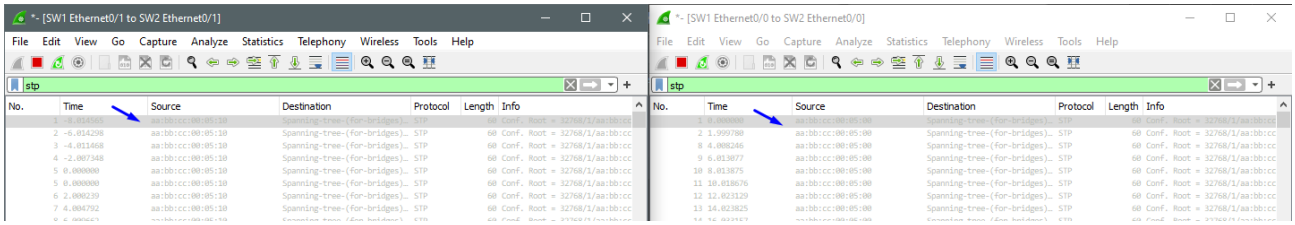


Figura 2 - Pacotes STP

De seguida, com o comando `no spanning-tree vlan 1-1014` em ambos os switchs, desativou-se o protocolo para todas as Vlans possíveis.

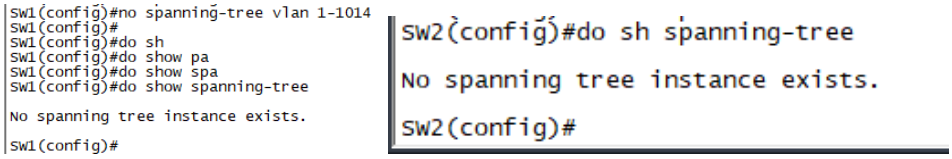


Figura 3 - Desativar STP

Posteriormente, realizou-se um ping entre os dois PCs da rede e conseguimos perceber que a comunicação nunca é estabelecida. A imagem seguinte demonstra isso.

```
PC1> ping 192.168.1.2

192.168.1.2 icmp_seq=1 timeout
192.168.1.2 icmp_seq=2 timeout
192.168.1.2 icmp_seq=3 timeout
192.168.1.2 icmp_seq=4 timeout
192.168.1.2 icmp_seq=5 timeout

PC1> ping 192.168.1.2

192.168.1.2 icmp_seq=1 timeout
192.168.1.2 icmp_seq=2 timeout
192.168.1.2 icmp_seq=3 timeout
192.168.1.2 icmp_seq=4 timeout
192.168.1.2 icmp_seq=5 timeout

PC1> ping 192.168.1.2 -t

host (192.168.1.2) not reachable

PC1> ping 192.168.1.2

host (192.168.1.2) not reachable
```

Figura 4 - Término da comunicação

Capturou-se novamente tráfego nas duas ligações (desta vez sem nenhum filtro) e conseguimos perceber que existiu uma troca contínua de pacotes ARP, nos dois caminhos, enviados pelo mesmo equipamento. Estas mensagens ocorrem com intervalos de tempo inferiores a 1 segundo. Isto acontece porque, uma vez que o STP foi desligado, o sw1 recebe o pedido ARP e envia-o para o sw2 por uma das *interfaces*. O sw2 recebe esse pedido e volta a reencaminhá-lo pela 2ª *interface*, ou seja, pela outra ligação, que não está bloqueada. Estas trocas de mensagens estão sempre e ocorrer e o sw2 nunca consegue entregar os pacotes ao PC2, visto que recebe por uma *interface* (de uma das ligações) e envia por outra que está também ligada ao sw1. Formam-se então os ciclos que congestionam uma rede e impossibilitam, por exemplo, a comunicação entre equipamentos. Na imagem seguinte, a partir da captura referida anteriormente, conseguimos ter uma amostra do que acontece quando se desliga o protocolo STP e começam-se a criar ciclos de envio e receção dos pacotes ARP. Vemos também que nas duas ligações existem pacotes enviados por ambos os switchs.

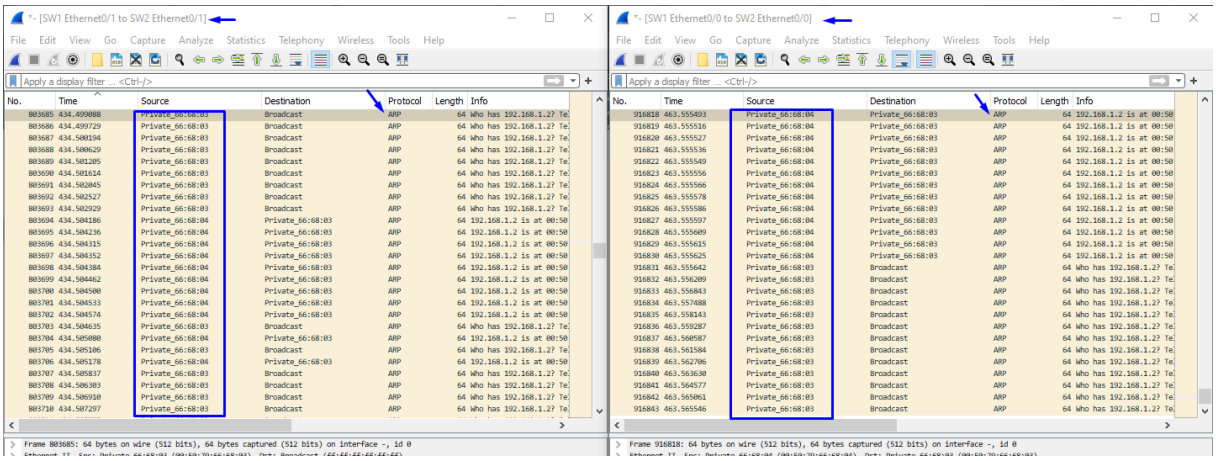


Figura 5 - Ciclos Wireshark

3. Protocolo STP

3.1. Conteúdo teórico STP

Para a redundância de caminhos existir e a criação de *loops* ser evitada em simultâneo, o STP define uma espécie de árvore que abrange todos os switches de uma rede e este aspeto é comum entre todos os protocolos. O STP força a que certos caminhos fiquem no estado de *standby* (*blocking*) e deixa outros no estado de *forwarding*. Se uma das ligações ficar indisponível, o protocolo adapta a rede de maneira que, a porta que estava bloqueada entre em ação.

Existem 3 funções às quais as portas podem estar atribuídas: *designated port, root port e alternate port*. Associado a isto, as portas também podem ficar configuradas em 4 estados*: blocking, listening, learning e forwarding.*

Todos os switchs da rede elegem uma Root Bridge que vai ser o switch mais importante. É a partir dele que todas as decisões na rede vão ser tomadas, tais como a quem e onde colocar os estados e funções referidas anteriormente. Diferentes Vlans devem ter diferentes Root Bridge. A eleição deste switch pode ser feita a partir do gestor de rede ou na escolha automática do switch com Bridge ID menor, constituída pela prioridade do switch e pelo MAC Address. Nestes protocolos, quanto menor a prioridade de um equipamento, melhor. Tudo isto é possível devido à troca de informações que os switchs fazem entre si a partir das Bridge Protocol Data Units (BPDUs). Cada switch compara as BPDUs que recebe com aquelas que envia. Quando um switch recebe uma BPDU com um Root ID (identificação da Root Bridge) inferior ao seu, atualiza o seu parâmetro do Root ID até todos os equipamentos chegarem a um acordo de qual é o switch com um identificador menor e aí consideram esse switch como a Root Bridge.

3.2. Experiência STP - A

Com a ajuda de uma topologia criada no GNS3, conseguimos perceber melhor como este protocolo funciona. Entendendo bem o STP, obtemos as bases mais importantes para que os restantes protocolos sejam mais fáceis de serem aprendidos. Na imagem seguinte conseguimos ver a topologia realizada. De notar que não foi alterada a prioridade em nenhum switch, visto que o objetivo era perceber como é que o STP vinha configurado nos switchs e como realmente funciona. A prioridade pode ser alterada, de maneira a escolhermos qual o switch que funciona como Root Bridge, mas não é relevante para realmente entender como este protocolo funciona. As questões mais relevantes postas em estudo foram por exemplo: como o protocolo escolhe a Root Bridge, como é que se atribuem as características às diferentes portas, como é feita a comunicação entre os switchs e como o mesmo reage perante falhas e alterações. Estes testes e estudos são prolongados também para os outros protocolos.

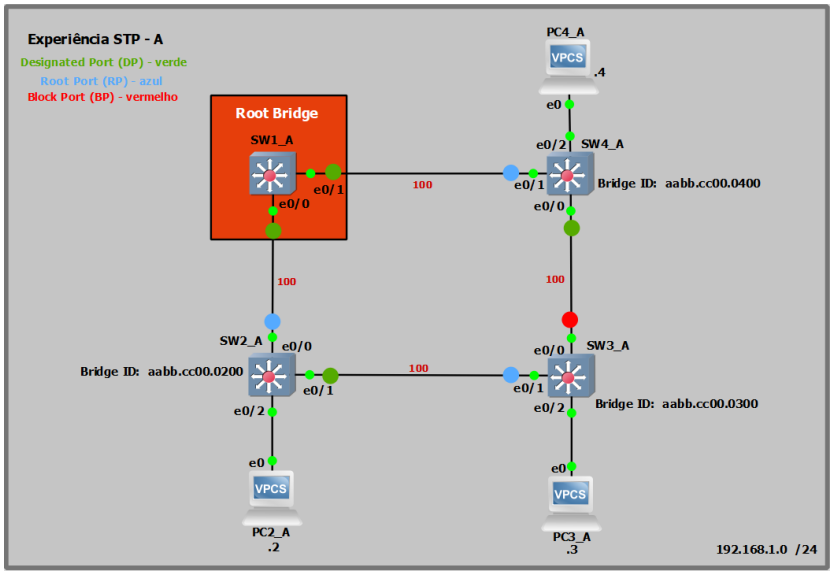


Figura 6 - Topologia STP A

Ao serem adicionados os equipamentos da imagem anterior, com o comando `show spanning-tree`, conseguimos ver que o sw1 é a Root Bridge, visto que, apesar de ter a mesma prioridade que os outros switchs, contém um MAC Address menor (aabb.cc00.0100 < .0200 < .0300 < .0400), como conseguimos ver de seguida. Assim, percebemos que se a escolha do switch principal for feita pelo protocolo, é escolhido o que tiver menor prioridade e depois o que tiver menor MAC.

```
SW1#sh spanning-tree

VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    32769
              Address     aabb.cc00.0100
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32769  (priority 32768 sys-id-ext 1)
              Address     aabb.cc00.0100
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  300 sec
```

Figura 7 - Root Bridge A

As portas foram estabelecidas segundo as designações que a imagem inicial apresenta pelos círculos coloridos (azul para *Root*, verde para *Designated* e vermelho para *Alternate*, bloqueada). Com o comando `show spanning-tree`, conseguimos ver as funções e estados das *interfaces*. Sabendo que as mesmas são escolhidas perante a posição que a Root Bridge ocupa e a partir dos custos das ligações necessárias para se chegar até ao sw1 (quanto menos custo, melhor), podemos resumir da seguinte maneira o porquê destas escolhas por parte do protocolo:

- As do sw1 são ambas *Designated* porque é a Root Bridge e as *interfaces* dela ficam com esta designação já que são a opção obviamente melhor da ligação para chegar ao sw1.
- No sw2 temos a e0/0 como Root Port porque esta é a porta do switch que acarreta menos custo (100) para chegar até ao sw1. A e0/1 ficou como designated pela mesma lógica.
- No sw4 a lógica é a mesma do sw2. A e0/1 é a *interface* que menos custo total leva para chegar à Root Bridge e a e0/0 é a melhor porta da ligação sw4-sw3.

- Por fim, no sw3, temos empate para eleger a Root Port, visto que ambas as *interfaces* levam o mesmo custo (200) para chegar ao sw1.
    - Nesta situação, é escolhida a *interface* que estabelece uma ligação ao vizinho que tem menor Bridge ID. Visto que tanto o sw2 como o sw4 têm a prioridade igual, então o aspeto a ter em conta é o switch com menor MAC da Bridge ID, que neste caso é o sw2.
    - Ficamos com a e0/1 como Root Port e a e0/0 como porta bloqueada visto que é a única porta que não pode levar outra designação que não esta. Esta porta ficará como de *backup* para uma falha.

Entendendo o porquê da Root Bridge e das portas serem escolhidas de tal maneira, conseguimos levar isto para os outros protocolos, visto que se baseiam na mesma ideia.

3.2.1. Capturas e injeção de falhas

Com a ferramenta Wireshark, para melhor se perceber a troca de pacotes BPDUs entre os switchs e a estrutura destas mensagens, realizou-se uma captura entre sw1-sw4 e sw4-sw3. A primeira é exemplificada na imagem seguinte.

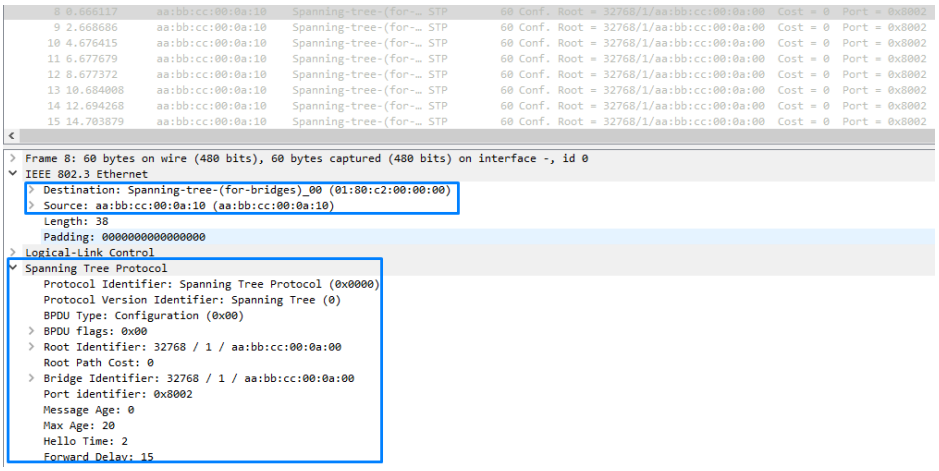


Figura 8 - Captura sw1- sw4

Estas mensagens são enviadas de 2 em 2 segundos (*Hello Time:* 2) e nelas conseguimos ver:

- **Root ID:** contém a *bridge priority* e o MAC Address da Root Bridge.
- **Root Path Cost:** apresenta o custo da ligação entre a Root Bridge e o switch que enviou o BPDU.
- **Bridge ID:** contém a *bridge priority* e o MAC Address do switch.
- **Message age**: representa a distância em switchs até à Root Bridge. Cada switch que não seja Root e que envie uma BPDU incrementa este valor em 1.
- **Max age:** representa o tempo máximo possível de existir antes que uma porta de uma bridge guarde as informações da BPDU.

- **Hello time:** intervalo de tempo entre as mensagens enviadas

Neste caso, a mensagem teve origem na *interface* e0/1 do sw1 e foi enviada para o grupo multicast de Spanning Tree. Contém o Root Id do switch 1 que é a Root Bridge da rede e também os identificadores do switch que enviou a mensagem, o sw1. Neste caso estes dois aspetos coincidem, visto que o pacote foi enviado pelo sw1 que é a Root Bridge.

De seguida, experimentou-se injetar uma falha na porta e0/1 do sw1 (fazendo o comando `shut` nessa *interface*) para entender como o protocolo se adapta a uma mudança na rede. Quando se faz isto, ao fazer uma captura de tráfego entre o sw1 e o sw4, conseguimos ver que os pacotes STP deixam de ser enviados durante, aproximadamente, 20 segundos. Sendo assim, um switch demora 20 segundos a perceber que não foi enviada nenhuma BPDU do vizinho, ou seja, ao final de 10 BPDUs não recebidas, como podemos nas duas imagens seguintes: a primeira representa o último pacote enviado antes da falha e conseguimos ver que a *flag* "topology change" encontra-se no estado "no". Na segunda imagem vemos que é o primeiro pacote enviado depois da falha e já conseguimos ver que a *flag* "topolgy change" encontra-se a "yes", pois houve uma alteração na topologia. Esta *flag* é colocada a "yes" pela Root Bridge quando recebe uma Topology Change Notification (TCN, explicado a baixo).

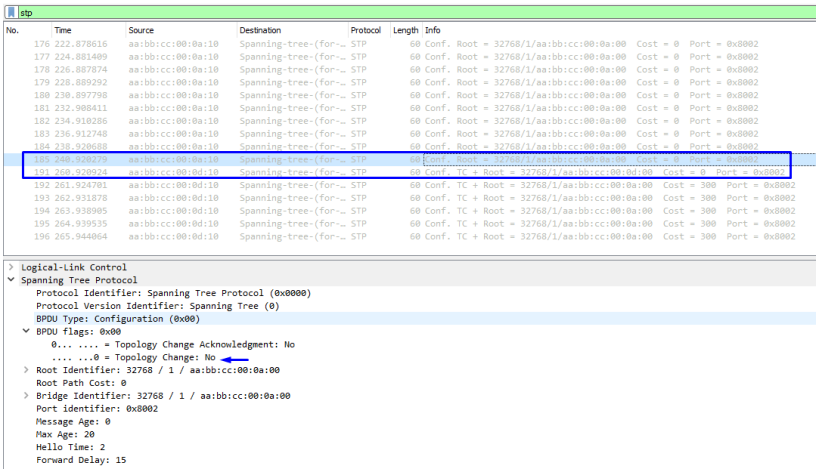


Figura 9 - Último pacote antes da falha

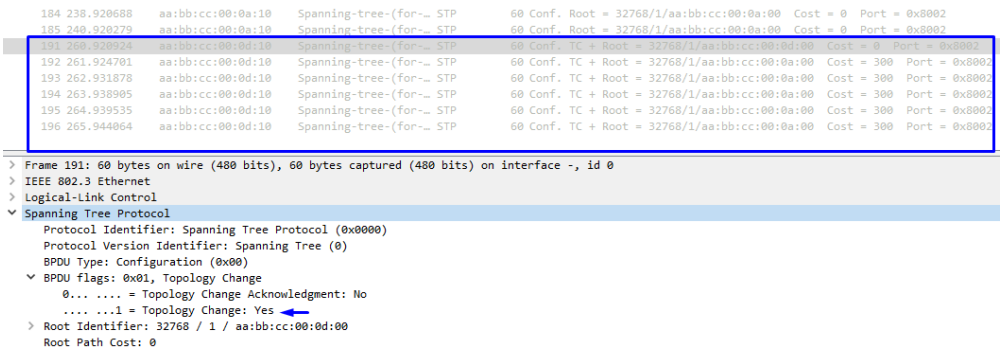


Figura 10 - Primeiro pacote depois da falha

Se fizermos capturas nas restantes 3 ligações, conseguimos ver que é enviada, em todas elas, uma mensagem que segue os mesmos aspetos de origem e destino. No entanto, apenas contém a informação de que ocorreu uma mudança na topologia, para assim os restantes switches conseguirem também ajustar as suas portas. Esta mensagem é uma TCN, uma BPDU especial que não contém qualquer tipo de informação e é enviada em direção à Root Bridge pelo switch que deteta a mudança na rede, para a avisar do que aconteceu.

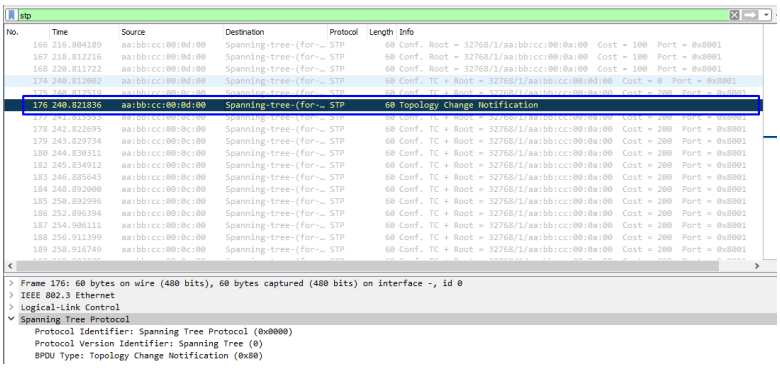


Figura 11 - Exemplo de TCN

Depois destes 20 segundos, o switch que tem a porta de *backup* (a que está no estado de bloqueada), coloca a mesma em *Listening* durante 15 segundos e em seguida no estado de *Learning* também durante 15 segundos. Só após estes dois estados é que a *interface* passa para *Forwarding* e o serviço é restabelecido. Podemos concluir que o STP demora 50 segundos a reagir a uma falha e a colocar a rede novamente disponível.

Com isto, se o comando `sh spanning-tree` for feito em todos os switchs, podemos ver que alterações as portas sofreram:

- **SW4:** a Root Port passa a ser a e0/0, visto que a anterior (e0/1) está ligada à porta da Root Bridge que se desligou. A e0/1 passa a ser *designated*, já que agora o caminho tem de passar por essa porta e dar a volta à topologia.
- **SW3:** a e0/0 do sw3 que se encontrava bloqueada, passou a estar no estado de *designated*
    - É que aqui que o STP trabalha, mantendo a redundância e assegurando a resolução de falhas na rede.
- **SW1 e SW2:** o switch 1 fica sempre com as suas portas a DP e o 2, como está ligado à Root, também mantém o que tinha anteriormente.

A próxima experiência realizada foi um ping do PC4 para o PC2 e a injeção da mesma falha anterior. Inicialmente o PC4 consegue pingar sem problemas o PC2. Os *echo request* e *reply* vão da *interface* e0/1 do sw4, visto que é esta a Root Port (conseguimos comprovar isso pois obtivemos esse mesmo tráfego entre o sw4 e o sw1, como vemos na imagem abaixo) e por fim chegam à e0/0 do sw2.

Quando é efetuada uma falha na *interface* e0/1 do sw1 e fazendo uma captura de tráfego entre o sw4-sw1 e o sw3-sw4, conseguimos ver que a primeira captura (a da esquerda) deixa de detetar tráfego ICMP e passado algum tempo, é a captura sobre a ligação sw3-sw4 que interceta tráfego, visto que o switch 4 muda a sua Root Port para a e0/0. Como referido anteriormente, o PC4 demora 50s a voltar a apresentar comunicação para com o PC2.

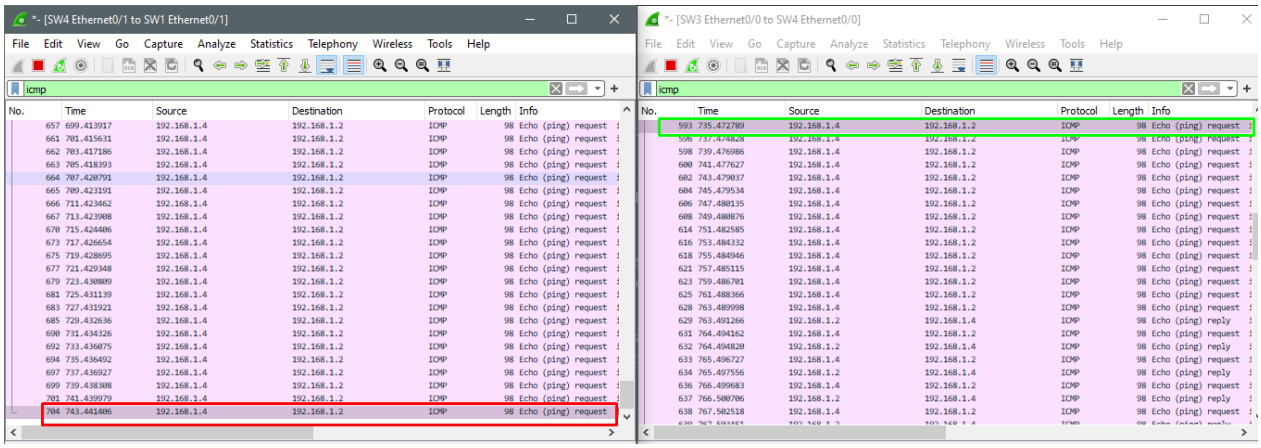


Figura 12 - Captura de tráfego com a falha

Nas imagens seguintes também conseguimos ver, com a ajuda do comando `show spanning-tree`, a passagem pelos 2 estados (*listening* e *learning*, depois dos 20 segundos sem BPDUs), que a porta que estava bloqueada no sw3 sofreu.

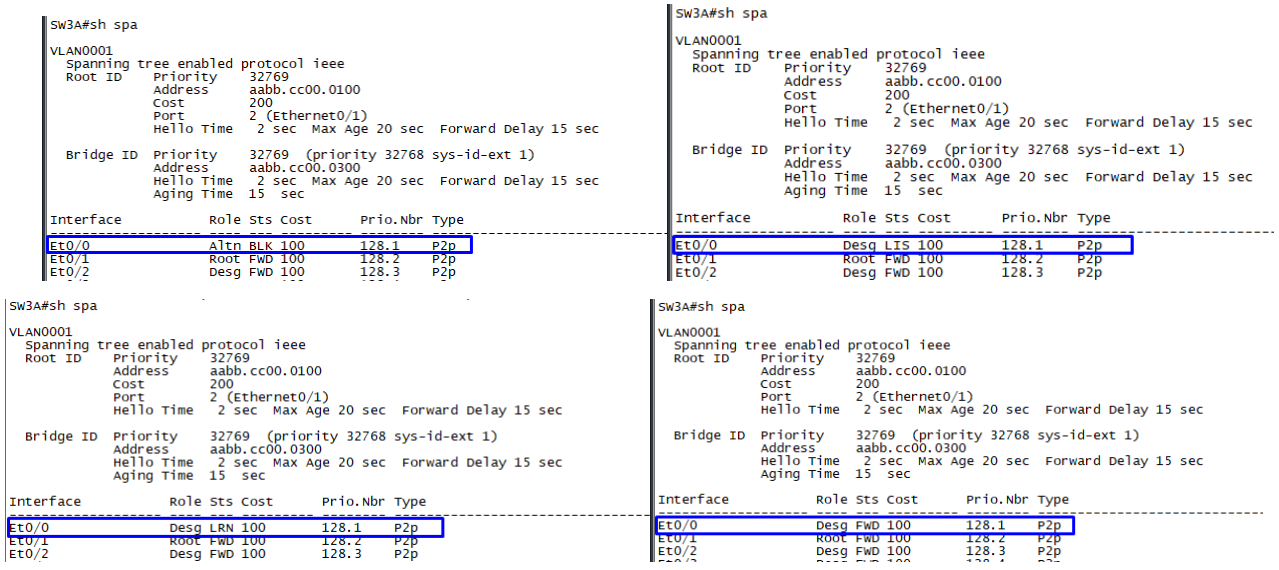


Figura 13 - Passagem de estados da e0/0 até RP

Com isto, podemos dizer que o protocolo Spanning Tree consegue adaptar-se perante falhas na rede, mudando as configurações das portas de cada router devido à falha ocorrida. As portas que se encontravam bloqueadas, entram em ação para assumirem o papel de DP e assim é possível a continuação dos serviços. No entanto há sempre uma demora relativamente grande associada a este processo. Os custos dos caminhos de cada switch até à root bridge acabam por mudar, visto que as

portas do switches mudam para outras funções e os equipamentos têm de usar outras ligações para lá chegarem. No subcapítulo seguinte, iremos perceber melhor como funciona a convergência do STP.

3.3. Experiência Convergência STP – AA

Visto que a experiência anterior serviu mais como uma base para entender os aspetos mais importantes (muitos deles gerais a todos os protocolos), realizou-se outra experiência, mais simples, para se perceber melhor como funciona a convergência no STP, ou seja, como são os processos que os equipamentos adotam e como os mesmo comunicam com a Root Bridge quando uma alteração na rede existe.

Como é possível de se ver na imagem seguinte, partiu-se de uma topologia com apenas 3 switchs onde inicialmente, não estavam todos ligados entre si, não existindo a possibilidade de acontecerem ciclos. Assim não há a necessidade de existirem portas bloqueadas. As funções iniciais das *interfaces* estão representadas também. Com o comando `spanning-tree vlan 1 root primary`, colocou-se o sw1 como Root Bridge.

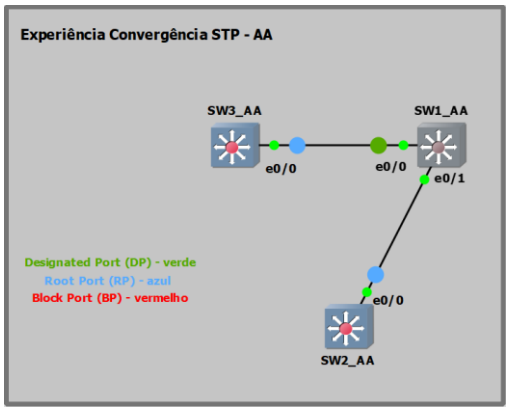


Figura 14 - Topologia STP AA

De seguida, adicionou-se uma ligação entre o switch 2 e 3, capturou-se tráfego nas 3 ligações com o Wireshark. Na captura realizada entre o sw2 e o sw1, conseguimos ver que o sw2 envia uma TCN para o sw1, visto que foi o switch 2 que detetou a mudança na rede e avisa então a Root Bridge.

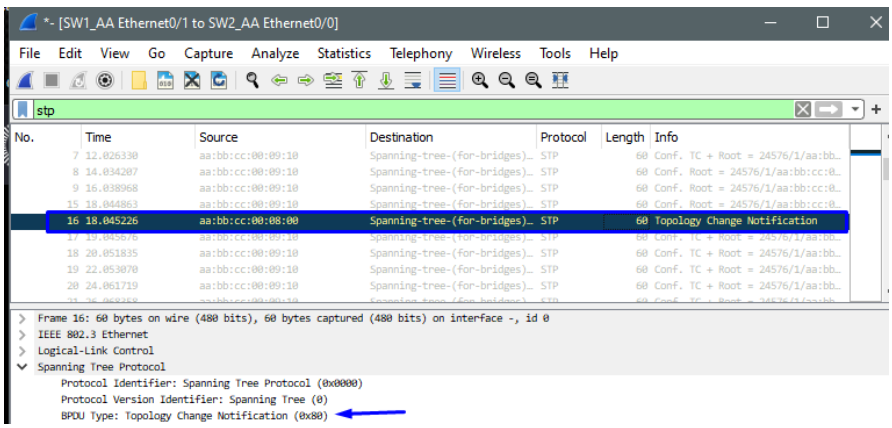


Figura 15 - TCN do sw2

A Root Bridge recebe este TCN e responde logo de seguida com uma BPDU onde a *flag* da "topology change acknowledgment" vem a "yes", para quem enviou o avisou de que a rede mudou, saber que a Root Bridge já tem conhecimento disso.

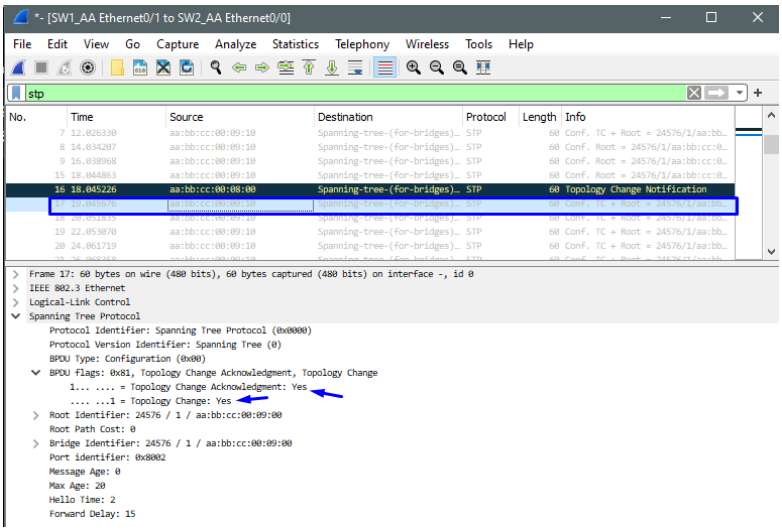


Figura 16 - Topology Change Acknowledgment

A porta e0/1 do sw3 ficou como *designated* e a e0/1 do sw2 como *alternate*, ou seja, bloqueada. Isto aconteceu porque o sw3 mandou BPDUs para o sw2 como informações melhores na Bridge ID, neste caso com um identificador mais baixo (como podemos ver na imagem seguinte). Sendo assim, o sw2 percebe que "perde" em relação ao switch 3 e bloqueia a sua porta.

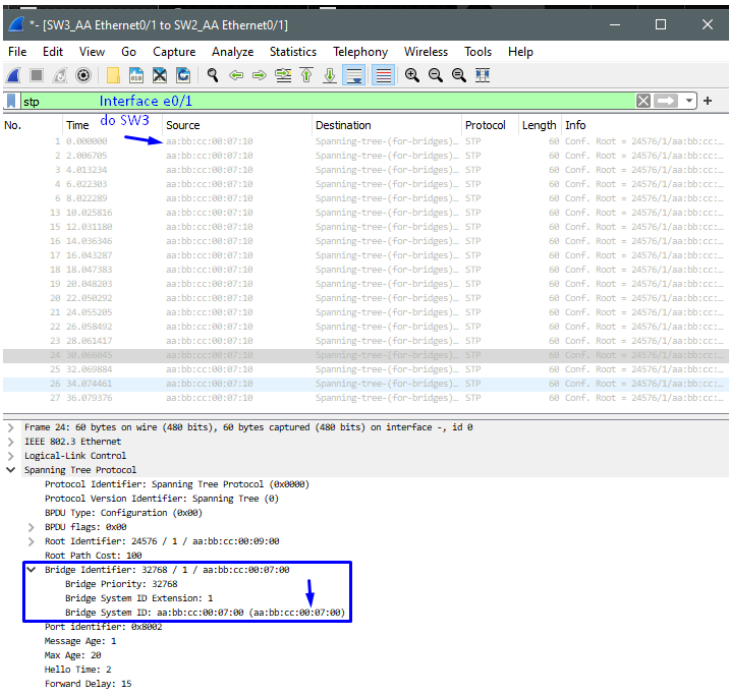


Figura 17 - Bridge ID sw3

O STP resolve perfeitamente o problema e consegue calcular, efetivamente, as novas alterações na rede. No entanto, este processo demora sempre imensos segundos. Assim, como iremos ver no capítulo seguinte, foi criado o protocolo RSTP, que é muito mais rápido.

4. Protocolo RSTP

4.1. Conteúdo Teórico RSTP

O protocolo Rapid Spanning Tree é uma evolução do STP. Melhora o tempo de convergência ao reduzir para apenas 3 Hello BPDUs perdidos para a rede perceber que tem de se adaptar perante uma alteração. O que torna este protocolo mais rápido, é que o mesmo é capaz de confirmar rapidamente que uma porta pode passar para o estado de *forwarding* sem depender de qualquer temporizador.

As características do protocolo RSTP não diferem muito do anterior. Temos novos tipos de papéis para as portas: *Alternate* e *Backup*. As mesmas podem estar nos estados de *Learning*, *Forwarding* e *Discarding*, este último substitui os estados de *Disabled*, *Blocking* e *Listening* visto que é usado para também aprender o MAC Address da *interface* e impede o encaminhamento. No entanto, temos diferenças no preenchimento do campo *flag* e no envio das BPDUs. As portas com o papel de DP e RP continuam a desempenhar a mesma função que no STP. Já a *alternate* port que tínhamos no protocolo anterior, pode ser *backup port* (*backup* à DP) ou *alternate port* (*backup* à RP). Estas funções, tal como o STP, são determinadas perante o conteúdo trocado nas BPDUs.

A convergência no RSTP é baseada no mecanismo de *proposal/agreement*. Um mecanismo de negociação entre os switchs que aliado a um processo de *synchronization*, os equipamentos entendem a melhor maneira de colocar as suas portas. Com a ajuda do esquema seguinte, onde temos a Root Bridge e as portas representadas, conseguimos perceber melhor estes processos.

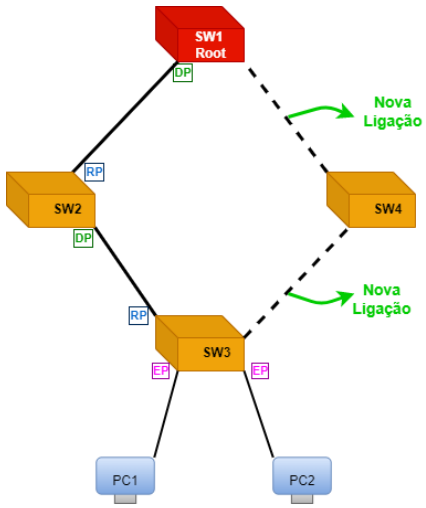


Figura 18 - Esquema convergência no RSTP

O processo de *proposal/agreement* inicia-se quando, por exemplo, é estabelecida uma nova ligação e acontece entre todos os switchs, por exemplo entre o switch 1 e 4. No entanto, antes disso, ambos colocam as portas que fazem parte dessa nova ligação no estado de *discarding*. De seguida, o sw1 envia uma proposta ao sw4 em forma de uma mensagem BPDU com as suas informações. Esta *propose* tem como intuito o sw1 mostrar que é a Root Bridge e colocar a sua porta no modo DP e indica ao sw4

que deve colocar a sua no modo Root. Quando o sw4 recebe a BPDU, percebe que é melhor do que aquela que ele poderia oferecer. No entanto, antes de aceitar a proposta, passa primeiro pelo processo de *sychronization* e tem o cuidado de colocar todas as suas portas, que não sejam *edge ports*, bloqueadas assegurando que nenhum *loop* seja criado durante a negociação entre os switchs. Depois disto, o sw4 aceita a proposta do sw1 e envia-lhe a mensagem de *agreement* e a nova ligação fica estabelecida com as seguintes decisões: DP do lado do sw1 e RP do lado do sw4. Estes processos são idênticos para as restantes ligações e no caso da sw3-sw2, o processo de negociação leva a que os switchs entendam que a porta bloqueada tem de ficar do lado do sw3, evitando assim a criação de *loops*. No subcapítulo seguinte, iremos analisar isto experiencialmente.

4.2. Experiência RSTP - B

Criou-se uma experiência idêntica à do esquema anterior (imagem seguinte) para perceber a rapidez do RSTP e como funciona a sua convergência. Foi necessário colocar todos os switchs no modo RSTP, como o comando `spanning-tree mode rapid-pvst`. Em todas as *interfaces* dos switchs que se ligavam a outros switchs, colocou-se no modo trunk com os comandos `switchport trunk encapsulation dot1q` e `switchport mode trunk` e colocou-se ligações point-to-point com o comando `spanning-tree link-type point-to-point`. Já nas portas que se ligam a terminais, colocou-se no modo access com o comando `switchport mode access` e em portfast com o `spanning-tree portfast`. Estas portas ficam do tipo *edge*. O sw1 é a Root Bridge. Como ainda não estabelecemos ligações do sw4 para o sw1 e sw3, ainda não temos a possibilidade existirem ciclos. Assim, a *interface* do sw1 encontra-se na função de *designated*, tal como a e0/1 do sw2. A e0/0 do 2º e 3º switch está em root.

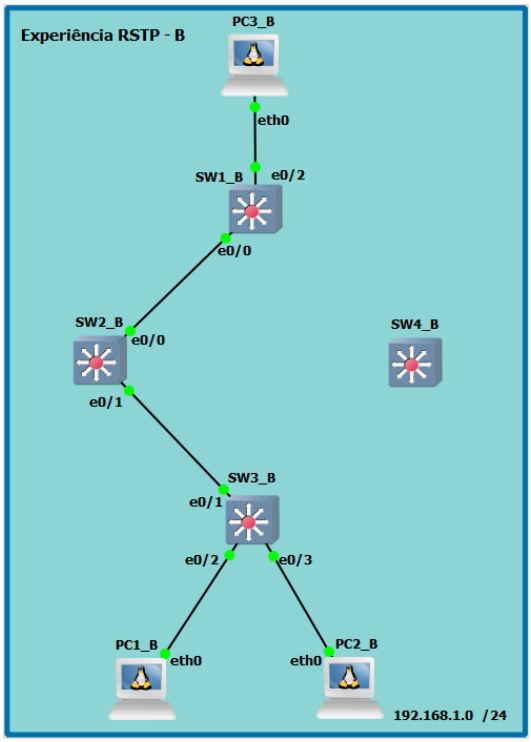


Figura 19 - Topologia RSTP B

Foi feita uma captura de tráfego entre o sw2-sw3. Segundo a próxima imagem, são enviados de cada vez, dois pacotes STP ao mesmo tempo. Um deles continua com o mesmo destino que o protocolo normal de Spanning Tree. O outro pacote é enviado também para um grupo multicast/broadcast do pvst+ (protocolo explicado mais à frente). A mensagem teve origem na e0/1 do sw2.

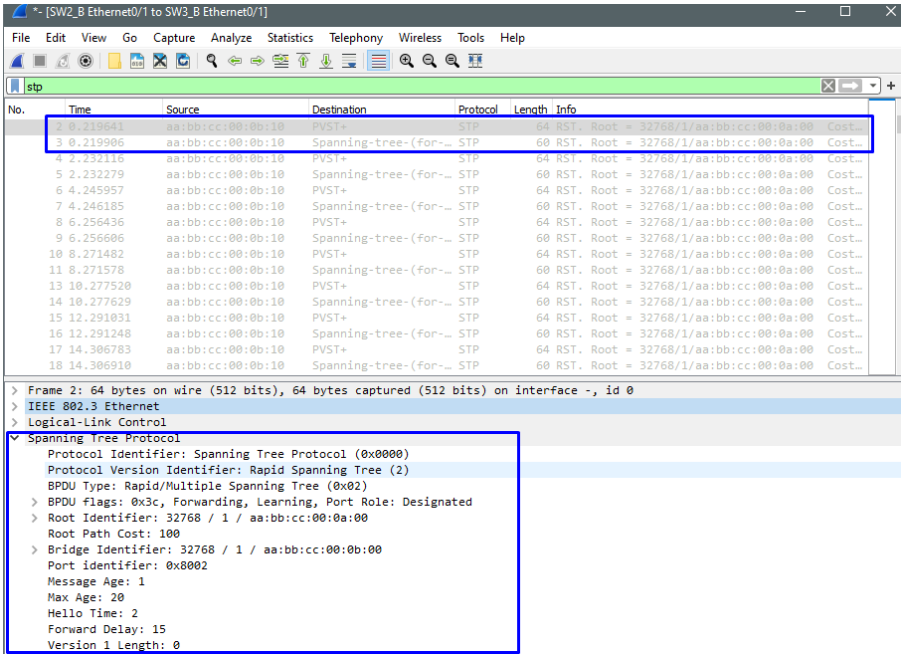


Figura 20 - Pacotes RSTP

Estes dois pacotes apresentam o mesmo conteúdo. Inicialmente conseguimos ver que a versão RSTP está identificada. Tal como referido anteriormente, este protocolo apresenta mais *flags*. Como a topologia ainda não sofreu nenhuma alteração, as *flags* de *topology change* (comuns ao STP) encontram-se a "no" e as de *Agreement* e *Proposal* também se encontram a "no". Já a *flag* que indica a função da porta por onde foi enviada a mensagem, encontra-se em *designated*, visto ser esta a função da e0/1 do switch que enviou o pacote. As *flags* de *Learning* e *Forwarding* encontram-se a "yes" e assim se mantiveram enquanto não houve uma alteração à rede. Vemos isto na próxima imagem.

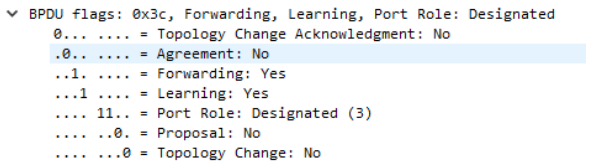


Figura 21 - Flags RSTP

Continuando a analisar o pacote, vemos a identificação da Root Bridge e da Bridge que enviou a BPDU. O custo do caminho até ao switch principal é apresentado e vemos a *message age*, o *max age* e o *Hello Time* que é igual ao protocolo estudado anteriormente.

Seguidamente, foi adicionado um novo switch, o sw4, que se ligou ao sw1 e 3. No momento da adição do equipamento, foi introduzido o comando `show spanning-tree` em todos os switchs e capturou-se tráfego STP em todas as ligações.

4.2.1. Capturas e injeção de falhas

Analisando o sw3 conseguimos ver que, inicialmente, como a *interface* e0/0 ainda não se encontrava com uma ligação estabelecida, a mesma passa para um estado de *blocking*, que representa o *discarding* inicial antes da negociação, enquanto que a Root port ainda se manteve na e0/1.

```
SW3_B(config-if-range)#do sh spa

VLAN0001
  Spanning tree enabled protocol rstp
  Root ID     Priority    4097
              Address     aabb.cc00.0a00
              Cost        200
              Port        2 (Ethernet0/1)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    12289  (priority 12288 sys-id-ext 1)
              Address     aabb.cc00.0c00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  300 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Et0/0               Desg BLK 100       128.1    P2p Peer(STP)
Et0/1               Root FWD 100       128.2    P2p
Et0/2               Desg FWD 100       128.3    P2p Edge
Et0/3               Desg FWD 100       128.4    P2p Edge
```

Figura 22 - Discarding antes da negociação

Como referido anteriormente com o esquema inicial, existiu a negociação do sw4 com o sw3 resultando em que a DP ficasse do lado deste último switch. Portanto a e0/0 do sw3 ficou com o papel de Root port. Como o switch 3 também teve de fazer o *proposal*/*agreemente* com o sw2, conseguimos ver que o sw3 fez a sincronização durante a negociação. Com isto, bloqueou todas as portas que não fossem *edge ports* (e0/2-3, imagem da esquerda). Visto que o sw2 apresentava uma BPDU melhor, ficou decidido que a DP ficaria do lado desse switch (imagem da direita) e a porta *alternate* ia para o lado do sw3, para impedir os *loops*.

```
SW3_B(config-if-range)#do sh spa

VLAN0001
  Spanning tree enabled protocol rstp
  Root ID     Priority    4097
              Address     aabb.cc00.0a00
              Cost        200
              Port        1 (Ethernet0/0)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    12289  (priority 12288 sys-id-ext 1)
              Address     aabb.cc00.0c00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  300 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Et0/0               Root FWD 100       128.1    P2p Peer(STP)
Et0/1               Altn BLK 100       128.2    P2p
Et0/2               Desg FWD 100       128.3    P2p Edge
Et0/3               Desg FWD 100       128.4    P2p Edge
Et1/0               Desg BLK 100       128.5    P2p
Et1/1               Desg BLK 100       128.6    P2p
Et1/2               Desg BLK 100       128.7    P2p

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Et1/3               Desg BLK 100       128.8    P2p
Et2/0               Desg BLK 100       128.9    P2p
```

```
SW2_B(config-if-range)#do sh spa

VLAN0001
  Spanning tree enabled protocol rstp
  Root ID     Priority    4097
              Address     aabb.cc00.0a00
              Cost        100
              Port        1 (Ethernet0/0)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    16385  (priority 16384 sys-id-ext 1)
              Address     aabb.cc00.0b00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
              Aging Time  300 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Et0/0               Root FWD 100       128.1    P2p
Et0/1               Desg FWD 100       128.2    P2p
Et0/2               Desg FWD 100       128.3    P2p
```

Figura 23 - Sychronization

Se analisarmos as ligações sw1-sw4 e sw4-sw3, aquando da introdução do novo switch, vemos a presença de pacotes STP do tipo Topology Change Notification nas duas capturas. Estes pacotes são enviados para a Root Bridge por quem deteta a alteração na rede. Neste caso, foi o sw3 primeiramente a detetar essa mudança e enviou a TCN para o sw4 que posteriormente enviou para a Root Bridge. A próxima imagem relata exatamente isto.

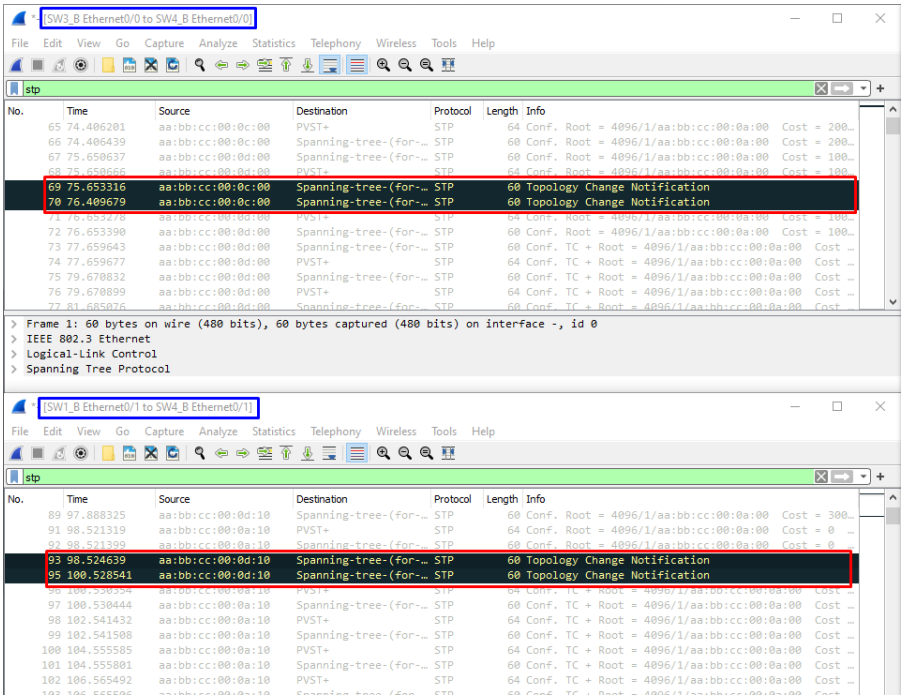


Figura 24 - TCN enviados na falha

Olhando para o pacote enviado imediatamente a seguir ao TCN no link sw1-4, o mesmo já é enviado novamente pela Root Bridge. Esta BPDU serve para o sw1 avisar quem detetou a mudança que já sabe da alteração na rede. Assim, esta mensagem vem com a *flag topology change acknowledgment* ativa, tal como a *flag topology change*.

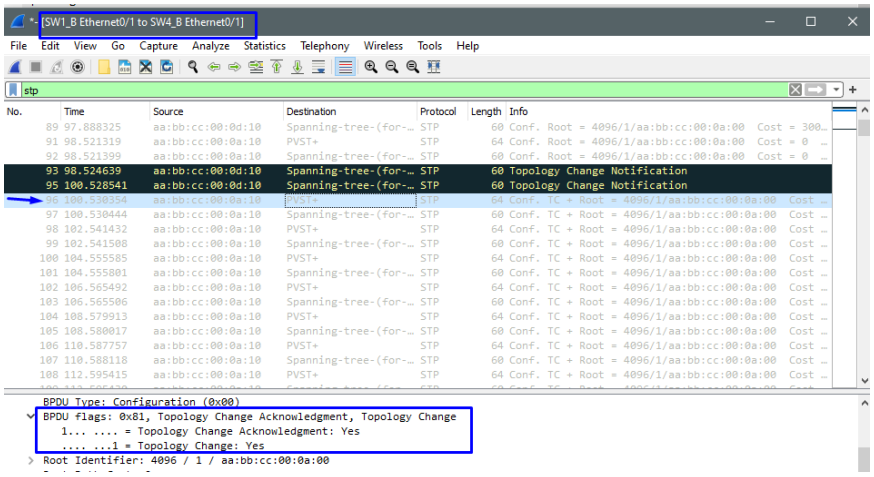


Figura 25 - TCA enviado pela Root

Entre o sw4 e o sw3 percebemos que o 4º switch, depois de receber pela Root Bridge a BPDU referida anteriormente, vai enviar também uma BPDU com as mesmas características para o sw3, visto que foi este a detetar a alteração para ele saber que a Root Bridge já foi identificada.

Podemos ver os pacotes *proposal* (falados anteriormente) no tráfego entre o sw1- sw4, antes da mensagem TCN. É o sw1 que envia uma BPDU ao sw4 com a *flag proposal* ativa. Isto demonstra o processo de *proposal*/*agreement* entre estes dois switchs. Neste caso, é o sw1 que está a propor ao sw4 que a DP fique do lado do 1º switch.

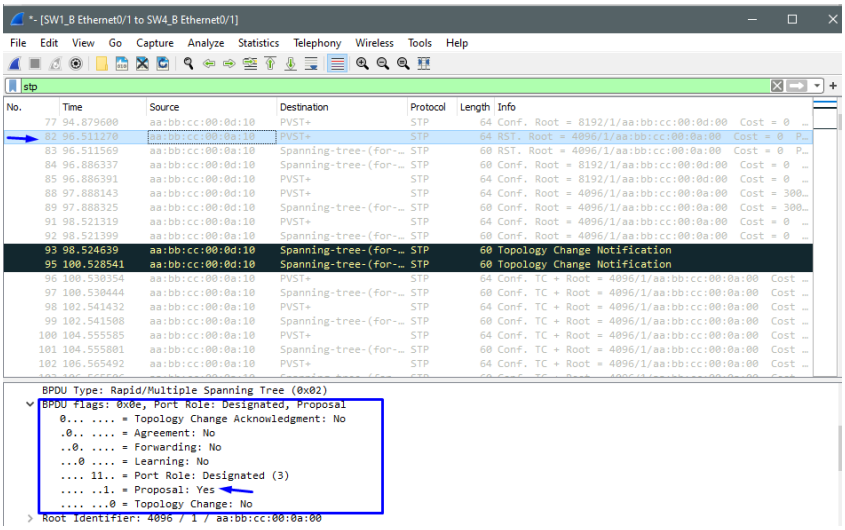


Figura 26 - Flag proposal com "yes"

Foi realizado um ping do PC1 para o 3 e, de seguida, foi injetada uma falha na *interface* e0/0 do sw3, visto ser esta a Root port do switch a que o PC1 se encontra ligado. Como confirmamos pela imagem seguinte, o protocolo RSTP é tão rápido a responder a uma falha que nem se deteta nenhuma alteração no ping entre os PCs.

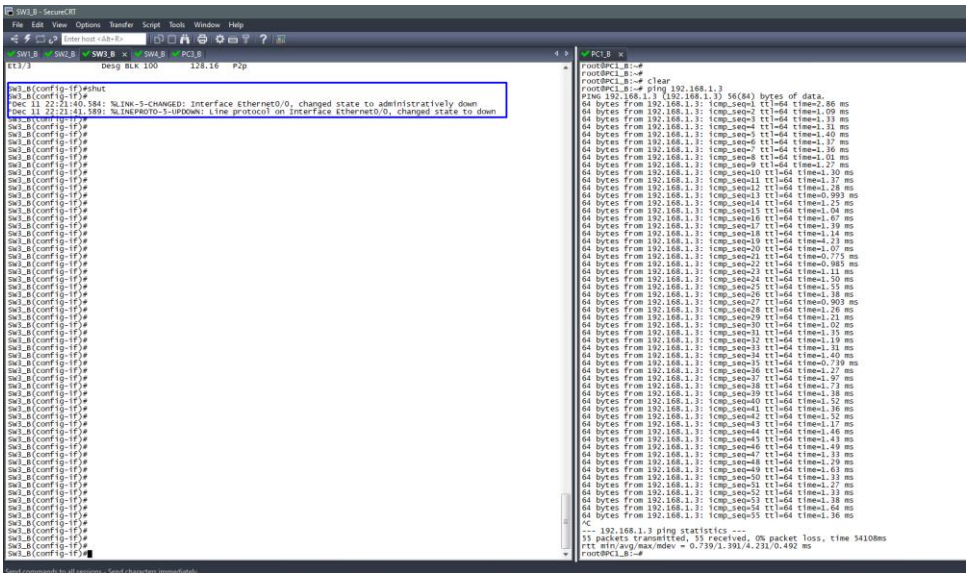


Figura 27 - Continuação do ping após falha

O sw3 rapidamente muda a sua porta Root para a porta que estava como *backup*, com o papel de *alternate*. Assim, a e0/1 passa para Root Port e as comunicações continuam.

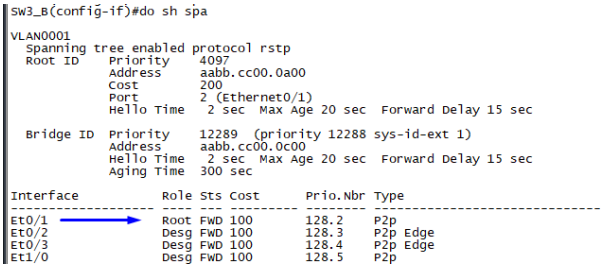
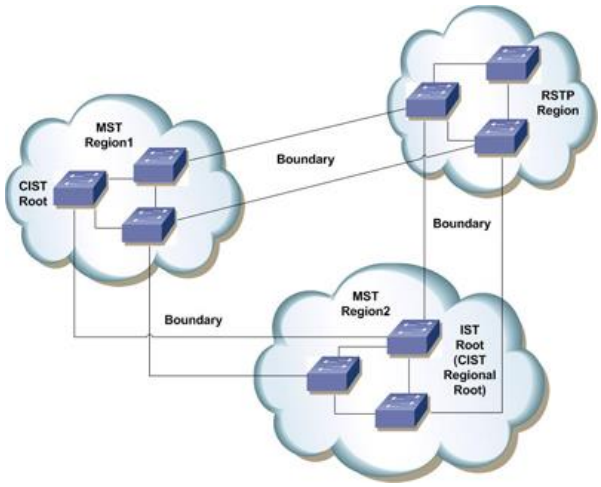


Figura 28 - Utilização da porta backup na falha

5. Protocolo MSTP

5.1. Conteúdo Teórico MSTP

O protocolo Multiple Spanning Tree foi criado segundo a norma IEEE 802.1s. O mesmo é baseado no protocolo RSTP, mas tem em conta o tratamento de várias Vlans. Enquanto que nos protocolos anteriores (STP e RSTP) cada Vlan representa uma instância, no MSTP podemos dividir as nossas Vlans num número menor de instâncias. Cada instância MSTP está associada a um conjunto de Vlans que partilham a mesma topologia lógica (spanning tree) e que pertencem a uma mesma região. Ou seja, se tivermos 10 Vlans, em vez de termos o processamento de um protocolo para cada uma das delas, agrupamos as mesmas numa instância que trata (por exemplo) da Vlan 1 até à 5 e outra instância que trabalha da Vlan 6 até à 10. Com várias instâncias, não é obrigatório que seja apenas um switch a ser a Root Bridge de cada uma delas. Diferentes instâncias podem ter diferentes switchs Root. As instâncias, associadas a diferentes regiões, são conectadas pelo Common Spanning Tree (CST), possibilitando que diferentes regiões comuniquem entre si e com isto, podemos ter o seguinte raciocínio: cada região (grupo de switchs) funciona como se fosse apenas um switch que se liga a outras regiões que funcionam da mesma maneira, o que forma o CST (ver imagem). Este protocolo causa um impacto menos prejudicial à computação dos equipamentos.



Como foi referido, o protocolo MST opera em regiões. Estas são grupos de switchs que partilham a mesma configuração que consiste num nome, num *revision number* que é um número administrativo escolhido pelo gestor e na criação das instâncias dessa região e respetiva atribuição das Vlans a essas instâncias. Por defeito, a região atribui a instância número 0 com a *revision number* igual a 0. Se um conjunto de switchs tiver estas características em comum, eles pertencem à mesma região.

5.2. Experiência MSTP - C

Para também estudarmos este protocolo de uma forma mais prática, realizou-se uma experiência com dois switchs que pertenciam à mesma região. Na imagem seguinte, está representada a topologia.

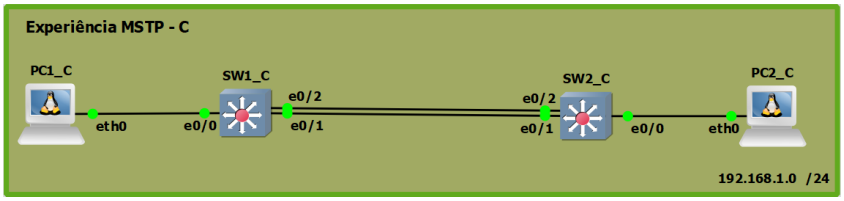


Figura 29 - Topologia MSTP C

Para ambos os switchs ficarem configurados na mesma região, foi necessário colocar-se em ambos o modo MSTP ativo, com o comando `spanning-tree mode mst`, configurar depois a região, com o seguinte conjunto de comandos: `spanning-tree mst configuration`, `name regiao1`, `revision 1`. Estas configurações só são aplicadas quando fazemos o comando `exit`. Colocou-se também no modo access nas ligações aos terminais, com o comando `switchport mode access` e `spanning-tree portfast` e as ligações entre os switchs foram colocadas no modo trunk, com os comandos `switchport trunk encapsulation dot1q` e `switchport mode trunk`.

Ao começar por se configurar apenas um switch, sendo que o mesmo estabelece ligações a outro switch, as portas começam no estado de *learning* (aprendizagem de MAC) e com a designação de Bound. Isto acontece porque as mesmas estão ligadas a equipamentos que correm protocolos de Spanning Tree que não o MST. Podemos ver, na imagem seguinte, que nas *interfaces* em questão aparece Bound(STP). Isto indica que este switch é o "limite" daquela região, visto que o outro equipamento ainda se encontrava noutro protocolo.

```
SW1_C#sh spanning-tree

MST0
  Spanning tree enabled protocol mstp
  Root ID     Priority    32768
              Address     aabb.cc00.0e00
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32768  (priority 32768 sys-id-ext 0)
              Address     aabb.cc00.0e00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Et0/0               Desg LRN 2000000   128.1    P2p Bound(STP)
Et0/1               Desg LRN 2000000   128.2    P2p Bound(STP)
Et0/2               Desg LRN 2000000   128.3    P2p
Et0/3               Desg LRN 2000000   128.4    P2p
```

Figura 30 - Bound (STP)

Se depois de configurarmos o sw2, formos ver como está a porta do sw1 (e0/0), a mesma encontra-se sem o Bound(STP), já que ambos os lados operam o protocolo MST.

5.2.1. Capturas

Analisando tráfego STP entre o switch 1 e 2, conseguimos perceber como são as BPDUs do MSTP. Visto que o *Hello Time* deste protocolo é igual ao dos restantes, as mensagens são trocadas a cada 2 segundos. Podemos ver que a estrutura do pacote BPDU é muito semelhante ao que já foi estudado até agora, principalmente em comparação ao RSTP, visto que temos também presentes as *flags* desse protocolo. Conseguimos comprovar isto, atendendo ao facto de que para além da versão aparecer identificada como "Multiple Spanning Tree", a BDPU Type vem com "Rapid/Multiple Spanning Tree", como podemos ver na próxima imagem.

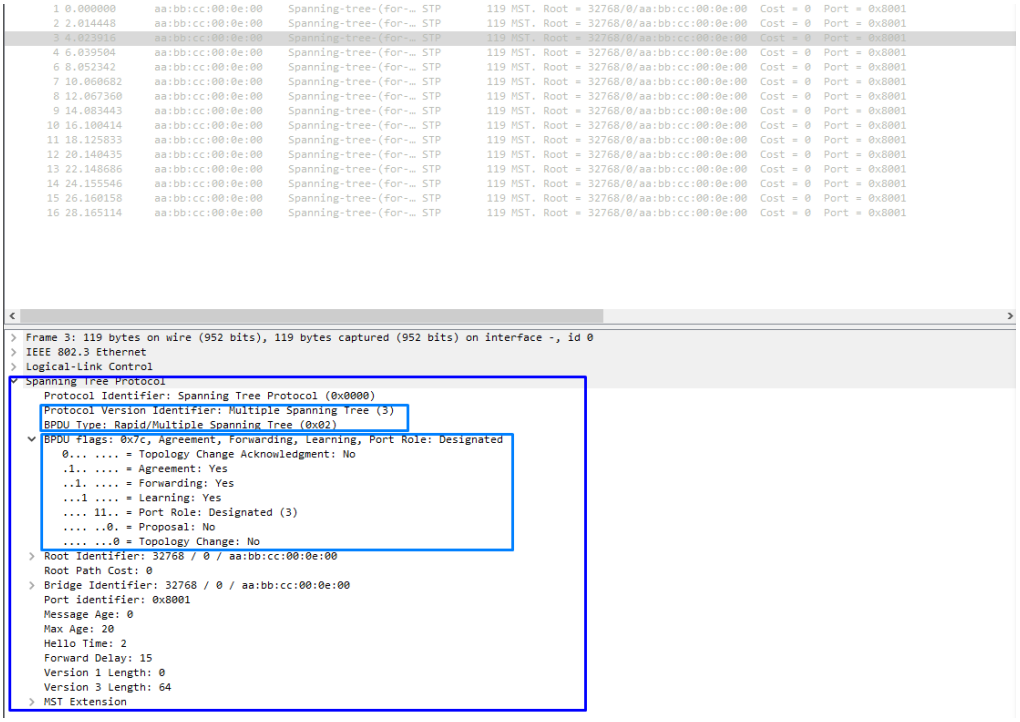


Figura 31 - Exemplo da BPDU MSTP

No entanto, estas BPDUs têm um parâmetro muito importante que leva a informação da região para a qual o switch está configurado. Esse parâmetro é o MST Extension que leva o nome da região e o número da revisão. Para além disso temos o MST Config digest, que é um código que o switch que recebe a BPDU lê para perceber se está na mesma região que a BDPU recebida. Portanto, com este MST Config digest, é definida na BPDU a região onde se enquadra o switch que a enviou.

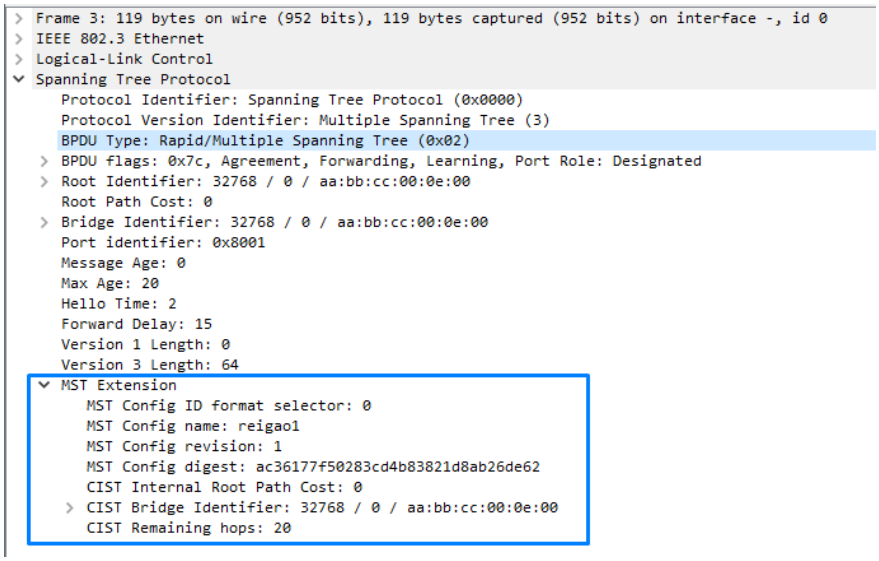


Figura 32 - Exemplo MST Extension

Se mudarmos o *revision number* ou o nome da região, o MST Digest não vai sofrer alterações, como conseguimos ver nas imagens seguintes ao alterar o *revision number* para 3 e o nome para "REGIAO1", onde os pacotes da direita representam os iniciais, sem alterações.

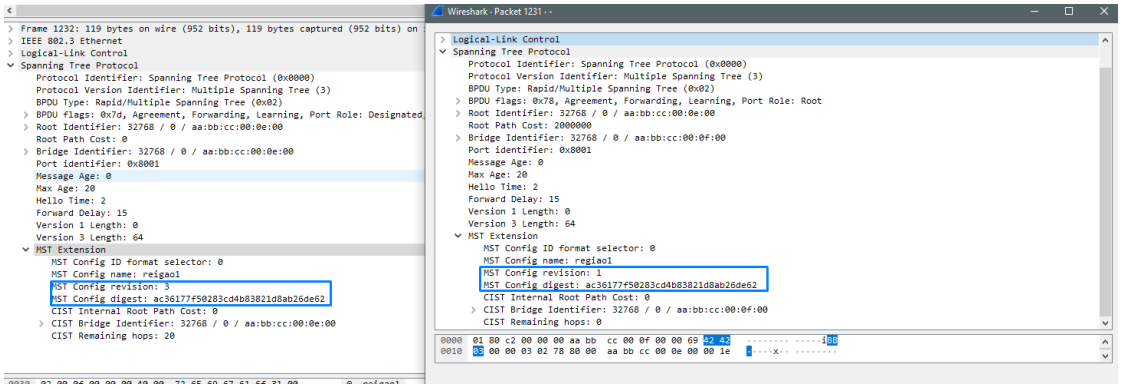


Figura 33 - Alteração do Revision Number

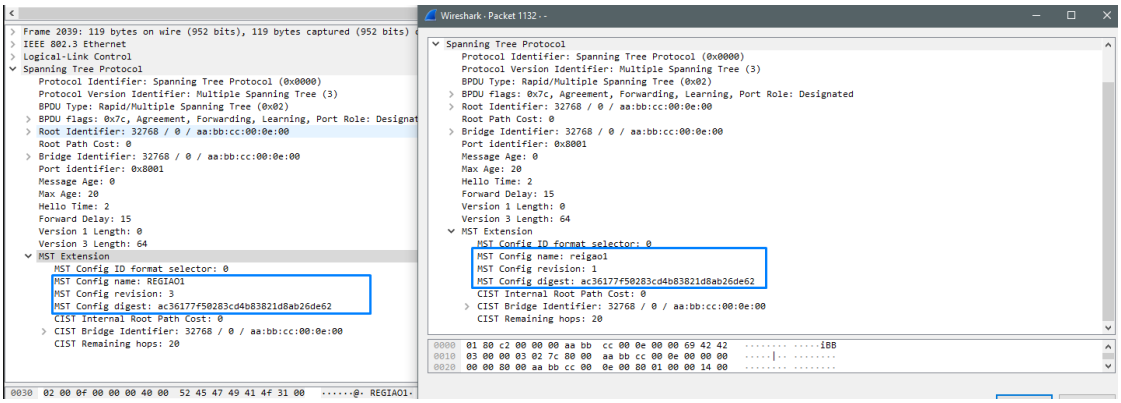


Figura 34 - Alteração do nome da região

Este valor não é alterado (nem com o nome, nem com o número) porque apenas sofre alterações quando, numa região, alteramos as Vlans a mapear para as instâncias que pretendermos. Assim, se criarmos uma nova instância no sw1 e associarmos a mesma a uma Vlan (por exemplo a 10), conseguimos ver que o MST Config digest muda de valor. Para fazer isto, apenas é necessário ir ao sw1, entrar no modo de configuração do MSTP com o comando `spanning-tree mst configuration` e com o comando `instance 1 vlan 10`, associar a Vlan 10 à instância 1. De notar que a Vlan não precisa de estar criada previamente. Apenas estamos a preparar uma instância que irá mapear essa Vlan. Com o comando `show spanning-tree configuration`, no sw1, conseguimos ver a criação da nova instância.

```
SW1_C#sh spanning-tree mst configuration
Name      [REGIAO1]
Revision  3     Instances configured 2

Instance  Vlans mapped
--------  ----------------------------------------------------------------
0         1-9,11-4094
1         10
----------------------------------------------------------------------------
SW1_C#
```

Figura 35 - Criação da instância 1

Na imagem seguinte, vemos que de facto, o valor do MST Digest foi alterado desta vez.

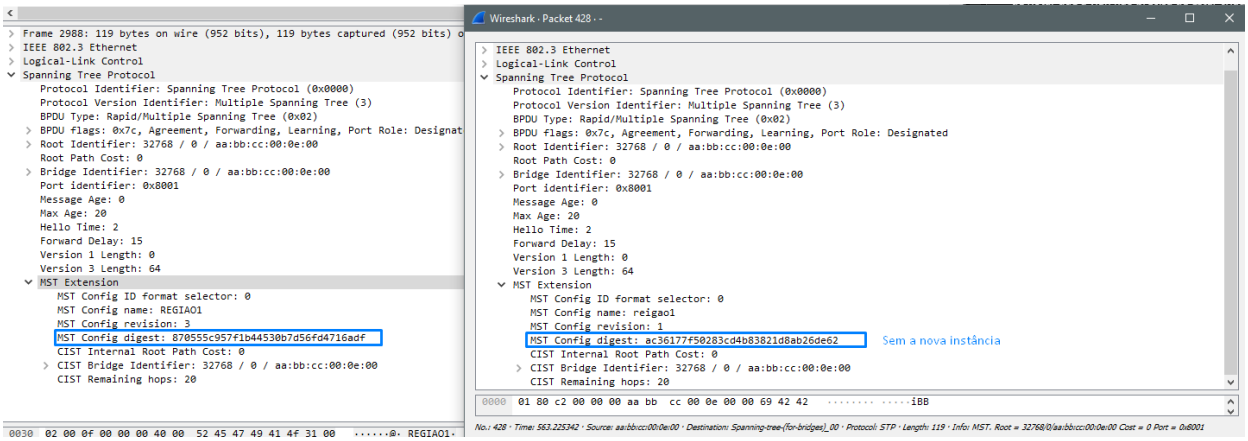


Figura 36 - Alteração do Config digest

Utilizando ainda esta topologia, como temos duas ligações entre o sw1 e o sw2, podemos experimentar usar apenas uma para encaminhar o tráfego das duas instâncias diferentes entre as duas máquinas. Para isto, apenas foram alteradas as instâncias, deixando estar os dois switchs na região "regiao1" e com o número 1. Assim, em ambos fizeram-se os comandos `instance 0 vlan 1-5, 11-4094` e `instance 1 vlan 6-10` e configuramos assim uma instância para receber da Vlan 6 até à 10 e outra que recebe as restantes Vlans. A partir da imagem seguinte, conseguimos analisar a criação das duas instâncias e que em ambas, é o sw1 que faz o papel de Root Bridge.

```
SW1_C#sh sp
MST0
  Spanning tree enabled protocol mstp
  Root ID     Priority    32768
              Address     aabb.cc00.0e00
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32768  (priority 32768 sys-id-ext 0)
              Address     aabb.cc00.0e00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Et0/0               Desg FWD 2000000   128.1    P2p Edge
Et0/1               Desg FWD 2000000   128.2    P2p
Et0/2               Desg FWD 2000000   128.3    P2p
Et0/3               Desg BLK 2000000   128.4    P2p
Et1/0               Desg BLK 2000000   128.5    P2p
Et1/1               Desg BLK 2000000   128.6    P2p
Et1/2               Desg BLK 2000000   128.7    P2p
Et1/3               Desg BLK 2000000   128.8    P2p
Et2/0               Desg BLK 2000000   128.9    P2p

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------

Et2/1               Desg BLK 2000000   128.10   P2p
Et2/2               Desg BLK 2000000   128.11   P2p
Et2/3               Desg BLK 2000000   128.12   P2p
Et3/0               Desg BLK 2000000   128.13   P2p
Et3/1               Desg BLK 2000000   128.14   P2p
Et3/2               Desg BLK 2000000   128.15   P2p
Et3/3               Desg BLK 2000000   128.16   P2p


MST1
  Spanning tree enabled protocol mstp
  Root ID     Priority    32769
              Address     aabb.cc00.0e00
              This bridge is the root
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32769  (priority 32768 sys-id-ext 1)
              Address     aabb.cc00.0e00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Et0/1               Desg FWD 2000000   128.2    P2p
Et0/2               Desg FWD 2000000   128.3    P2p
```

Figura 37 - Análise da instância 0 e 1

Já na próxima imagem, vemos que a porta e0/1 é a Root port do sw2 para ambas as instâncias. Ou seja, está a ser usada apenas uma ligação para a instância 0 e 1. No entanto isto pode ser alterado.

```
Sw2_C#sh spanning-tree

MST0
  Spanning tree enabled protocol mstp
  Root ID     Priority    32768
              Address     aabb.cc00.0e00
              Cost        0
              Port        2 (Ethernet0/1)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32768  (priority 32768 sys-id-ext 0)
              Address     aabb.cc00.0f00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Et0/0               Desg FWD 2000000   128.1    P2p Edge
Et0/1               Root FWD 2000000   128.2    P2p
Et0/2               Altn BLK 2000000   128.3    P2p
Et0/3               Desg FWD 2000000   128.4    P2p
Et1/0               Desg FWD 2000000   128.5    P2p
Et1/1               Desg FWD 2000000   128.6    P2p
Et1/2               Desg FWD 2000000   128.7    P2p
Et1/3               Desg FWD 2000000   128.8    P2p

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------

Et2/0               Desg FWD 2000000   128.9    P2p
Et2/1               Desg FWD 2000000   128.10   P2p
Et2/2               Desg FWD 2000000   128.11   P2p
Et2/3               Desg FWD 2000000   128.12   P2p
Et3/0               Desg FWD 2000000   128.13   P2p
Et3/1               Desg FWD 2000000   128.14   P2p
Et3/2               Desg FWD 2000000   128.15   P2p
Et3/3               Desg FWD 2000000   128.16   P2p


MST1
  Spanning tree enabled protocol mstp
  Root ID     Priority    32769
              Address     aabb.cc00.0e00
              Cost        2000000
              Port        2 (Ethernet0/1)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32769  (priority 32768 sys-id-ext 1)
              Address     aabb.cc00.0f00
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Et0/1               Root FWD 2000000   128.2    P2p
Et0/2               Altn BLK 2000000   128.3    P2p
```

Figura 38 - Porta e0/1 Root nas 2 instâncias

5.3. Experiência MSTP – CC

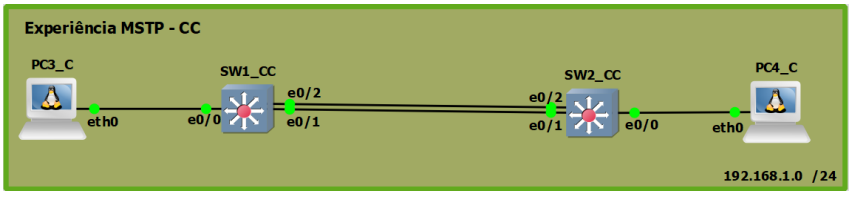


Figura 39 - Topologia MSTP CC

Podemos aproveitar quando temos ligações redundantes para dividir as instâncias pelas diferentes ligações, não colocando tanto tráfego numa ligação. Para isso, podemos por exemplo aumentar o custo da e0/1 do sw2 (porta Root) numa das instâncias (por exemplo MTS1) e assim, passará a ser a e0/2 que encaminha o tráfego para a Root Bridge, já que a porta preferida vai ser a que tem menos custo. Isto foi feito numa topologia igual à anterior entrando na *interface* e0/1 do 2º switch, com o comando `spanning-tree mst 1 cost 4500000`, fazemos com que a porta que tenha uma ligação que implica menos custo para a instância 1 seja a e0/1. Assim, como podemos ver na imagem seguinte, o sw2 passa a encaminhar o tráfego das Vlans da instância 0 pela porta e0/1 e as Vlans da instância 1 pela e0/2. Com isto, estamos a tirar proveito das duas ligações que existem entre os dois switchs.

```
Sw2_CC#sh spanning-tree

MST0
  Spanning tree enabled protocol mstp
  Root ID     Priority    32768
              Address     aabb.cc00.1000
              Cost        0
              Port        2 (Ethernet0/1)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32768  (priority 32768 sys-id-ext 0)
              Address     aabb.cc00.1100
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Et0/0               Desg FWD 2000000   128.1    P2p Edge
Et0/1               Root FWD 2000000   128.2    P2p
Et0/2               Altn BLK 2000000   128.3    P2p
Et0/3               Desg FWD 2000000   128.4    P2p
Et1/0               Desg FWD 2000000   128.5    P2p
Et1/1               Desg FWD 2000000   128.6    P2p
Et1/2               Desg FWD 2000000   128.7    P2p
Et1/3               Desg FWD 2000000   128.8    P2p
Et2/0               Desg FWD 2000000   128.9    P2p
Et2/1               Desg FWD 2000000   128.10   P2p
Et2/2               Desg FWD 2000000   128.11   P2p
Et2/3               Desg FWD 2000000   128.12   P2p
Et3/0               Desg FWD 2000000   128.13   P2p
Et3/1               Desg FWD 2000000   128.14   P2p
Et3/2               Desg FWD 2000000   128.15   P2p
Et3/3               Desg FWD 2000000   128.16   P2p


MST1
  Spanning tree enabled protocol mstp
  Root ID     Priority    32769
              Address     aabb.cc00.1000
              Cost        2000000
              Port        3 (Ethernet0/2)
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID   Priority    32769  (priority 32768 sys-id-ext 1)
              Address     aabb.cc00.1100
              Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec

Interface           Role Sts Cost      Prio.Nbr Type
------------------- ---- --- --------- -------- --------------------------------
Et0/1               Altn BLK 4500000   128.2    P2p
Et0/2               Root FWD 2000000   128.3    P2p
```

Figura 40 - As duas instâncias pelas duas ligações

5.3.1. Injeção de falhas

Para perceber se a rapidez deste protocolo é efetivamente igual à do RSTP, foi iniciado um ping entre o PC1 e o PC2. Durante o mesmo, foi injetada uma falha na porta Root do sw2 da primeira experiência, a e0/1.

Tal como o RSTP, com pings de 1 em 1 segundo, não se nota a falha que existiu na rede, visto que o MSTP usa o protocolo de proposal/agreement do RSTP.

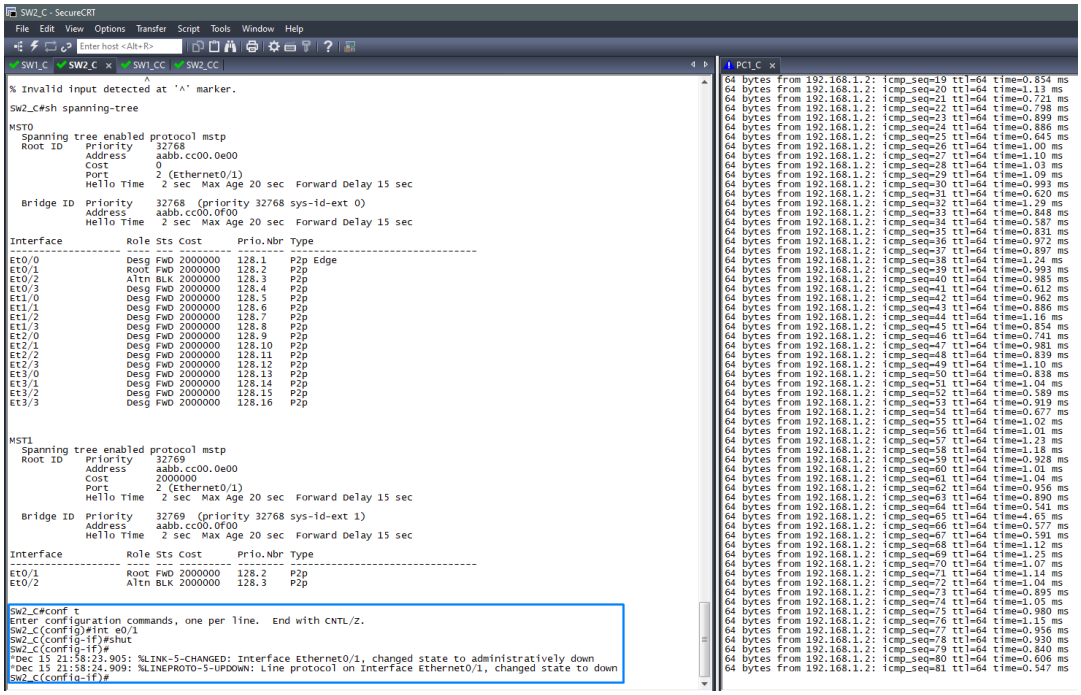


Figura 41 - Injeção da falha e continuação do ping

6. PVST+ e Rapid PVST+

Até agora, este relatório tem relatado as duas primeiras experiências, STP e RSTP, no ambiente GNS3. Na verdade, como estamos em ambientes Cisco, os mesmos não operam os protocolos anteriores no seu estado “puro”. São os protocolos PVST e Rapid PVST que correm nos equipamentos Cisco. Assim, a Cisco implementa nos seus switchs, o protocolo STP e RSTP mas para cada Vlan, tal como o nome do protocolo o diz: “Per Vlan”.

O PVST é um *upgrade* do STP que providencia uma Spannin Tree separada (uma instância) para cada Vlan, configurada na rede (vimos no protocolo do capítulo anterior (o MSTP) que o mesmo altera este aspeto).

Já o Rapid PVST melhora o RSTP com uma utilização do PVST. Fornece também uma instância 802.1w separada por cada Vlan, tal como o PVST+.

7. Conclusões

Todas os protocolos estudados conseguem produzir uma topologia redundante e livre de *loops*. Estas soluções oferecem métodos válidos para gerar redes fiáveis e com um bom grau de disponibilidade.

A melhor solução a ser implementada depende do porte da rede que é preciso gerir. Se a mesma for relativamente pequena, isto é, que contenha poucos switchs e sirva para um propósito de situações não urgentes e ambientes pouco sérios, o STP pode ser adotado e será o suficiente.

Contudo, o RSTP é um protocolo que pode ser utilizado nas mesmas situações que o STP e em circunstâncias em que a rede seja de porte médio, ou seja, um ambiente onde mais equipamentos estejam envolvidos ou numa situação mais relevante e onde a rapidez é indispensável, visto ser um protocolo com um menor tempo de convergência do que o seu antecessor. Em comparação direta com o STP, o protocolo Rapid acaba por ser preferível.

Caso a rede seja de um porte muito maior, abrangendo várias Vlans, o MSTP é a solução mais adequada já que foi criado para esse fim e também traz a velocidade do RSTP.