Phụ lục 3: SPANNING TREE PROTOCOL

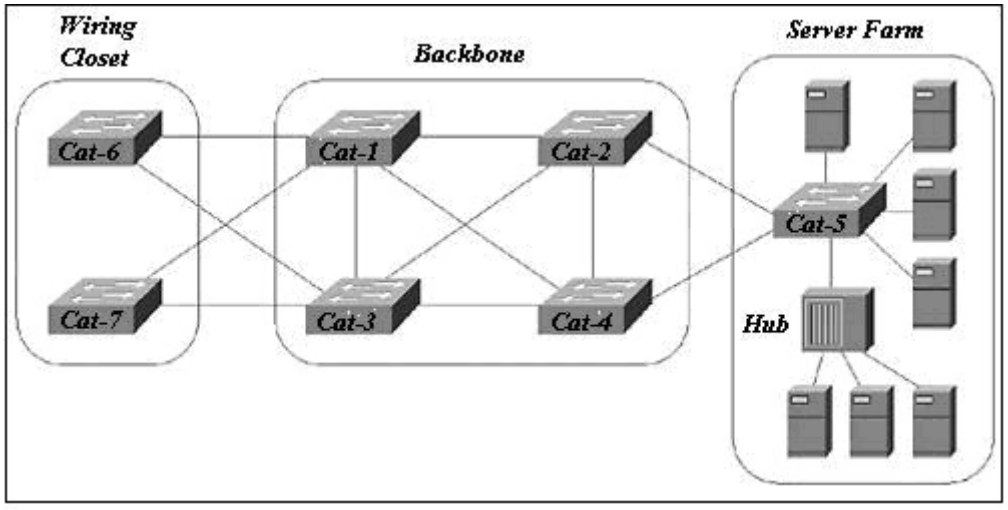
Một mạng mạnh mẽ được thiết kế không chỉ đem lại tính hiệu quả cho việc truyền các gói hoặc frame, mà còn phải xem xét làm thế nào để khôi phục hoạt động của mạng một cách nhanh chóng khi mạng xảy ra lỗi. Trong môi trường lớp 3, các giao thức định tuyến sử dụng con đường dự phòng đến mạng đích để khi con đường chính bị lỗi thì sẽ nhanh chóng tận dụng con đường thứ 2. Định tuyến lớp 3 cho phép nhiều con đường đến đích để duy trì tình trạng hoạt động của mạng và cũng cho phép cân bằng tải qua nhiều con đường.

Trong môi trường lớp 2 (switching hoặc bridging), không sử dụng giao thức định tuyến và cũng không cho phép các con đường dự phòng, thay vì bridge cung cấp việc truyền dữ liệu giữa các mạng hoặc các cổng của switch. Giao thức Spanning Tree cung cấp liên kết dự phòng để mạng chuyển mạch lớp 2 có thể khôi phục từ lỗi mà không cần có sự can thiệp kịp thời. STP được định nghĩa trong chuẩn IEEE 802.1D.

3.1 Spanning Tree là gì và tại sao phải sử dụng nó?

Spanning Tree Protocol (STP) là một giao thức ngăn chặn sự lặp vòng, cho phép các bridge truyền thông với nhau để phát hiện vòng lặp vật lý trong mạng. Sau đó giao thức này sẽ định rõ một thuật toán mà bridge có thể tạo ra một cấu trúc mạng logic chứa vòng lặp (loop-free). Nói cách khác STP sẽ tạo một cấu trúc cây của free-loop gồm các lá và các nhánh nối toàn bộ mạng lớp 2.

Vòng lặp xảy ra trong mạng với nhiều nguyên nhân. Hầu hết các nguyên nhân thông thường là kết quả của việc cố gắng tính toán để cung cấp khả năng dự phòng, trong trường hợp này, một liên kết hoặc switch bị hỏng, các liên kết hoặc switch khác vẫn tiếp tục hoạt động, tuy nhiên các vòng lặp cũng có thể xảy ra do lỗi. *Hình 3.1* biểu diễn một mạng chuyển mạch với các vòng lặp cố ý được dùng để cung cấp khả năng dự phòng như thế nào.

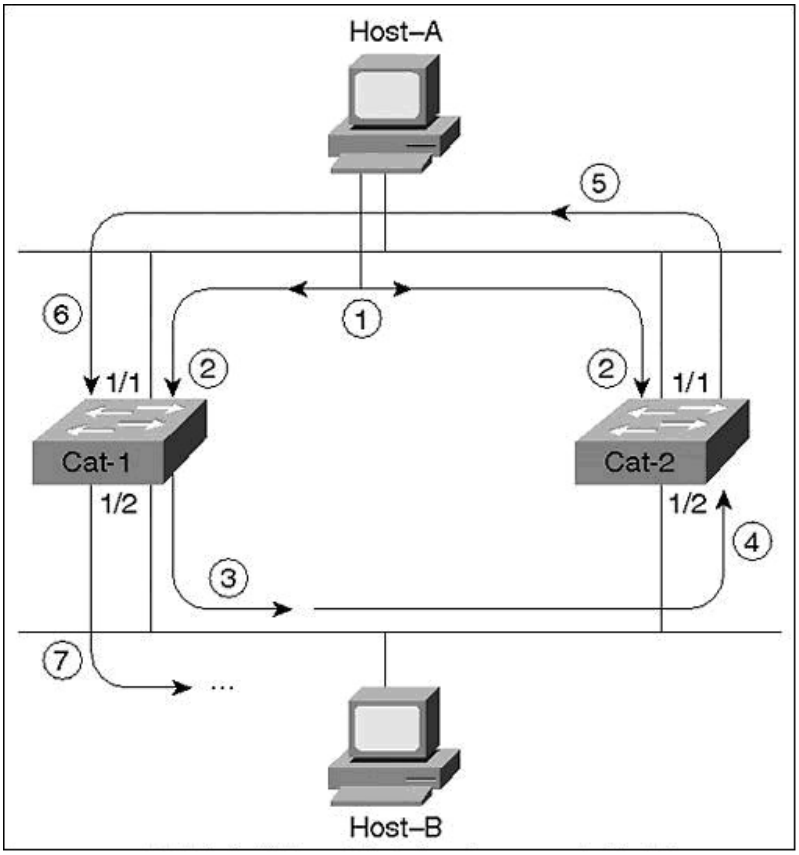


Hình 3.1: bridging loop trong mạng

Hai nguyên nhân chính gây ra sự lặp vòng tai hại trong mạng chuyển mạch là do broadcast và sự sai lệch của bảng bridge.

Vòng lặp broadcast

Vòng lặp broadcast và vòng lặp lớp 2 là một sự kết hợp nguy hiểm. *Hình 3.2* biểu diễn broadcast tạo ra vòng lặp phản hồi (feedback loop).



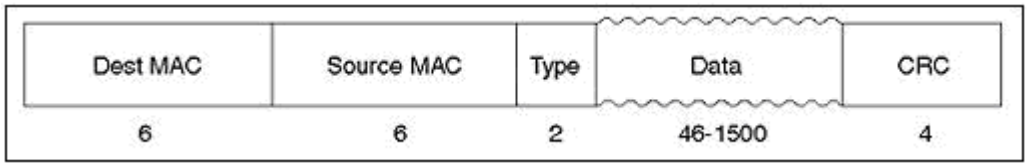
Hình 3.2: không có STP, broadcast tạo Feedback loop

Giả sử rằng, không có switch nào chạy STP:

- ***Bước 1***: host A gửi một frame bằng địa chỉ broadcast (FF-FF-FF-FF-FF-FF).
- ***Bước 2:*** frame đến cả hai Cat-1 và Cat-2 qua cổng 1/1
- ***Bước 3:*** Cat-1 sẽ đưa frame qua cổng 1/2.
- ***Bước 4***: frame được truyền đến tất cả các nút trên đoạn mạng Ethernet kể cả cổng 1/2 của Cat-2.
- ***Bước 5:*** Cat-2 đưa frame này đến cổng 1/1 của nó.
- ***Bước 6:*** một lần nữa, frame xuất hiện cổng 1/1 của Cat-1.
- ***Bước 7:*** Cat-1 sẽ gửi frame này đến cổng 1/2 lần hai. Như vậy tạo thành một vòng lặp ở đây.

Chú ý: frame này cũng tràn qua đoạn mạng Ethernet và tạo thành một vòng lặp theo hướng ngược lại, vòng lặp feedback xảy ra ở cả hai hướng. Một kết luận quan trọng nữa

trong *hình 3.2* là vòng lặp bridge nguy hiểm hơn nhiều so với vòng lặp định tuyến. *Hình 3.3* mô tả định dạng của một DIXv2 Ethernet frame.



Hình 3.3 : định dạng của một DIXv2 Ethernet frame

DIXv2 Ethernet Frame chỉ chứa 2 địa chỉ MAC, một trường Type và một CRC. Trong IP header chứa trường time-to-live (TTL) được thiết lập tại host gốc và nó sẽ được giảm đi 1 mỗi khi qua một router. Gói sẽ bị loại bỏ nếu TTL = 0, điều này cho phép các router ngăn chặn các datagram bị "run-away". Không giống như IP, Ethernet không có trường TTL, vì vậy sau khi một frame bắt đầu bị lặp trong mạng thì nó vẫn tiếp tục cho đến khi ai đó ngắt một trong các bridge hoặc ngắt một kiên kết.

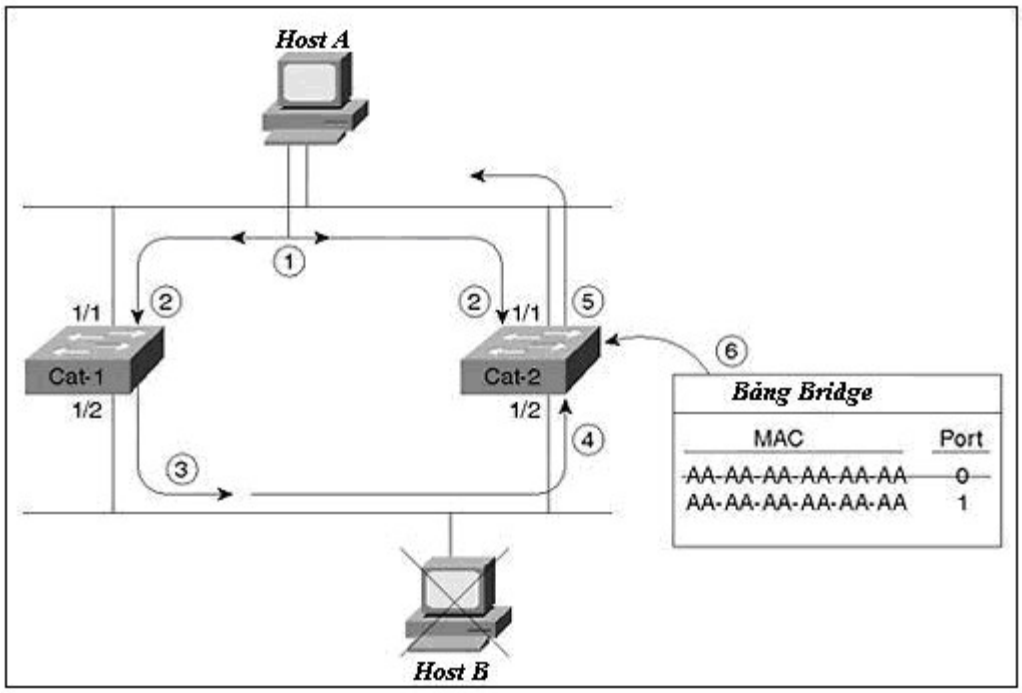
Trong một mạng phức tạp hơn mạng được mô tả trong *hình 3.1, và 3.2* thì có thể gây ra vòng lặp feedback rất nhanh theo tỉ lệ số mũ. Vì cứ mỗi frame tràn qua nhiều cổng của switch, thì tổng số frame tăng nhanh rất nhiều

Ngoài ra cần phải chú ý đến cơn bão broadcast trên người dùng của host A và B trong *hình 3.2*. Broadcast được xử lý bởi CPU ở tất cả các thiết bị trên mạng. Trong trường hợp này, các PC đều cố xử lý bão broadcast. Nếu ta ngắt một trong số các kết nối, thì nó trở lại hoạt động bình thường. Tuy nhiên, ngay khi ta kết nối nó trở lại thì broadcast sẽ sử dụng 100% CPU. Nếu ta không xử lý điều này mà vẫn tiếp tục sử dụng mạng, thì sẽ tạo ra vòng lặp vật lý trong mạng.

Việc sai lệch bảng bridge

Nhiều nhà quản trị switch/bridge đã nhận thức vấn đề cơ bản của bão broadcast, tuy nhiên ta phải biết rằng thậm chí các unicast frame cũng có thể truyền mãi trong mạng mà chứa vòng lặp. *Hình 3.4* mô tả điều này.

- ***Bước 1:*** host A muốn gửi gói unicast đến host B, tuy nhiên host B đã rời khỏi mạng, và đúng với bảng bridge của switch không có địa chỉ của host B.
- ***Bước 2:*** giả sử rằng cả hai switch đều không chạy STP, thì frame đến cổng 1/1 trên cả hai switch.
- ***Bước 3:*** vì host B bị down, nên Cat-1 không có địa chỉ MAC (BB-BB-BB-BB-BB-BB) trong bảng bridge, và nó tràn frame qua các cổng.
- ***Bước 4:*** Cat-2 nhận được frame trên cổng 1/2 . Có 2 vấn đề xảy ra:
    - ***Bước 5:*** Cat-2 tràn frame vì nó không học địa chỉ MAC BB-BB-BB-BB-BB-BB, điều này tạo ra feedback loop và làm down mạng.
    - Cat-2 chú ý rằng, nó chỉ nhận một frame trên cổng 1/2 với địa chỉ MAC là AA-AA-AA-AA-AA-AA. Nó thay đổi địa chỉ MAC của host A trong bảng bridge dẫn đến sai cổng.



Hình 3.4: frame unicast cũng có thể gây ra Bridging Loop và làm sai lệnh bảng bridge

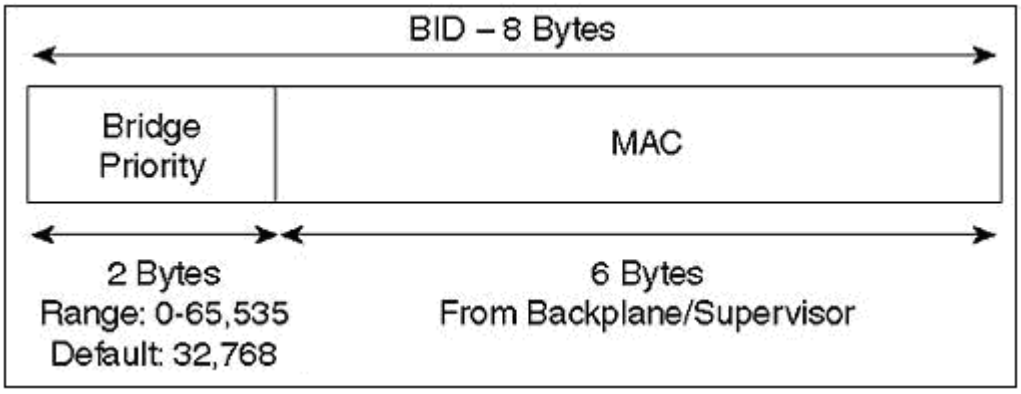
Vì frame bị lặp theo hướng ngược lại, nên ta thấy địa chỉ MAC của host A bị lẫn giữa cổng 1/1 và 1/2. Điều này không chỉ làm mạng bị tràn với các gói unicast mà còn sửa sai bảng bridge. Như vậy không chỉ có broadcast mới làm hư hại mạng.

3.2 Hai khái niệm cơ bản của STP

Việc tính toán Spanning Tree dựa trên hai khái niệm khi tạo ra vòng lặp logic trong cấu trúc mạng đó là: Bridge ID (BID) và chi phí đường đi.

Bridge ID (BID)

BID là một trường có 8 byte, nó gồm có 2 trường con được miêu tả như *hình 3.5* sau:



Hình 3.5 : hai trường của BID

Trong đó:

- ***Địa chỉ MAC:*** có 6 byte được gán cho switch. Catalyst 5000 và 6000 sử dụng một trong số các địa chỉ MAC từ vùng 1024 địa chỉ gán cho mỗi giám sát viên (supervisor) và bảng nối đa năng (backplane). Địa chỉ MAC trong BID sử dụng định dạng hexa.

Chú ý: một vài Catalyst lấy địa chỉ MAC từ module giám sát (như Catalyst 5000) và lấy địa chỉ khác từ backplane (như Catalyst 5500 và 6000)

- ***Bridge Priority:*** là độ ưu tiên bridge có 2 byte tạo thành 2^{16} giá trị từ 0 - 65.535. Độ ưu tiên bridge có giá trị mặc định là giá trị ở khoảng giữa (32.768).

Chú ý: ta chỉ tập trung vào phiên bản IEEE của giao thức Spanning Tree. Mặc dù về cơ bản là như nhau nhưng có một vài điểm khác biệt giữa IEEE STP và DEC STP như DEC STP sử dụng 8 bit Bridge priority.

Chi phí đường đi

Bridge sử dụng khái niệm chi phí để đánh giá các bridge khác. 802.1D định nghĩa chi phí là 1000 Mbps bằng cách chia băng thông của liên kết. Ví dụ như một liên kết 10BaseT có chi phí là 100 (1000/10), Fast Ethernet và FDDI sử dụng chi phí là 10 (1000/100). Tuy nhiên với việc gia tăng của Gigabit Ethernet và OC-48 ATM (2,4Gbps), thì chi phí được lưu trữ là một giá trị nguyên mà không phải là phân số. Ví dụ như kết quả OC-48 ATM trong 1000/2400 Mbps= 41667 bps, một giá trị chi phí không hợp lệ. Do đó các chi phí lớn hơn hoặc bằng 1 Gbps thì có chi phí là 1, tuy nhiên điều này ngăn cản STP lựa chọn chính xác "con đường tốt nhất" trong mạng Gigabit.

Để giải quyết tình trạng khó xử này, IEEE quyết định sửa đổi chi phí để sử dụng tính co dãn không tuyến tính. *Bảng 3.1* cho ta một danh sách giá trị chi phí mới.

Băng thông	Chi phí STP
4 Mbps	250
10 Mbps	100
16 Mbps	62
45 Mbps	39
100 Mbps	19
155 Mbps	14
622 Mbps	6
1 Gbps	4
10 Gbps	2

Bảng 3.1 Danh sách chi phí mới

Giá trị trong *bảng 3.1* được chọn cẩn thận để sơ đồ hoạt động cũ và mới có tốc độ liên kết nhanh như hiện nay. Một điểm chú ý là giá trị chi phí STP càng thấp càng tốt.

3.3 Các bước ra quyết định của STP

Khi tạo ra cấu trúc mạng logic chứa vòng lặp (loop-free) thì Spanning Tree luôn dùng trình tự bốn bước sau:

- BID gốc (Root BID) thấp nhất.
- Chi phí đường đi đến Bridge gốc thấp nhất.
- BID của người gửi thấp nhất.
- ID của cổng (PortID) thấp nhất.

Bridge trao đổi thông tin Spanning Tree với nhau, sử dụng frame xác định là đơn vị dữ liệu giao thức bridge (Bridge Protocol Data Unit - BPDU). Một bridge sử dụng trình tự bốn bước này để lưu một bản sao của BPDU tốt nhất trên mỗi cổng. Khi đánh giá, nó xem tất cả BPDU nhận được trên cổng cũng như BPDU gửi đi trên cổng đó. Mỗi BPDU đến đều được kiểm tra theo trình tự bốn bước này, nếu tốt hơn BPDU hiện tại thì nó được lưu lại cổng đó và thay thế giá trị cũ.

Chú ý: các bridge sẽ gửi BPDU cấu hình cho đến khi nhận nhiều hơn một BPDU tốt.

Thêm vào đó, quá trình lưu lại BPDU tốt nhất cũng điều khiển việc gửi các BPDU. Khi một bridge lần đầu tiên hoạt động, thì tất cả các cổng của nó được gửi BPDU 2s một lần (đây là giá trị mặc định của bộ định thời). Tuy nhiên, nếu một cổng lắng nghe một BPDU từ một bridge khác tốt hơn BPDU mà nó gửi, thì cổng sẽ ngưng gửi BPDU. Nếu BPDU này từ một lân cận ngưng đến trong một khoảng thời gian (20 s là mặc định) thì cổng tiếp tục gửi BPDU lại lần nữa.

Chú ý: có 2 loại BPDU là BPDU cấu hình và BPDU thông báo thay đổi cấu trúc mạng (TCN).

3.4 Sự hội tụ STP ban đầu (Initial STP Convergence)

Phần này ta sẽ xem xét thuật toán mà STP sử dụng để hội tụ lần đầu tiên trên cấu trúc mạng logic chứa vòng lặp (loop-free). Mặc dù có nhiều khía cạnh STP, nhưng sự hội tụ ban đầu được phân nhỏ thành ba bước sau:

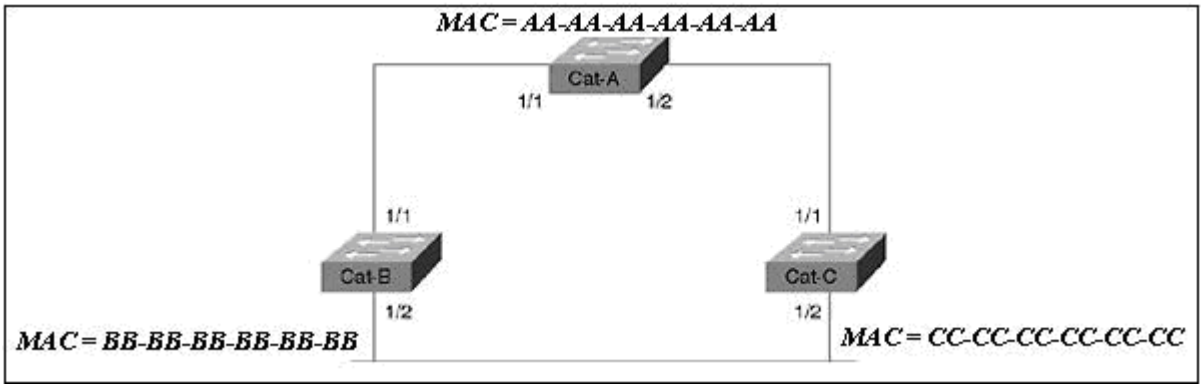
- Quyết định một bridge gốc (Root Bridge).
- Quyết định cổng gốc (Root Port).
- Quyết định cổng được chỉ định (Designated Port).

Khi một mạng khởi động lần đầu, tất cả các bridge thông báo thông tin BPDU một cách lộn xộn. Tuy nhiên, các bridge này sẽ lập tức áp dụng trình tự bốn bước (*ở phần 3.1.3*). Một bridge gốc được quyết định để hoạt động như là "trung tâm của vạn vật" đối với mạng. Tất cả các bridge còn lại tính toán việc thiết lập các cổng gốc và các cổng chỉ

định để xây dựng cấu trúc mạng chứa loop-free. Kết quả là bridge gốc giống như một hub với các đường đi loop-free ra bên ngoài. Khi mạng có trạng thái ổn định, thì bridge gốc sẽ gửi các BPDU đến mỗi đoạn mạng.

Sau khi mạng hội tụ trên cấu trúc mạng loop-free, nếu có thêm sự thay đổi thì sẽ sử dụng quá trình thay đổi cấu trúc mạng.

Hình 3.6 là mô hình của một mạng switch/bridge. Mạng này gồm có ba bridge kết nối thành một vòng lặp. Mỗi cầu nối được gán một địa chỉ MAC không có thật tương ứng với tên thiết bị (ví dụ như Cat-A sử dụng địa chỉ MAC là AA-AA-AA-AA-AA-AA).



Hình 3.6: mô hình mạng sử dụng STP

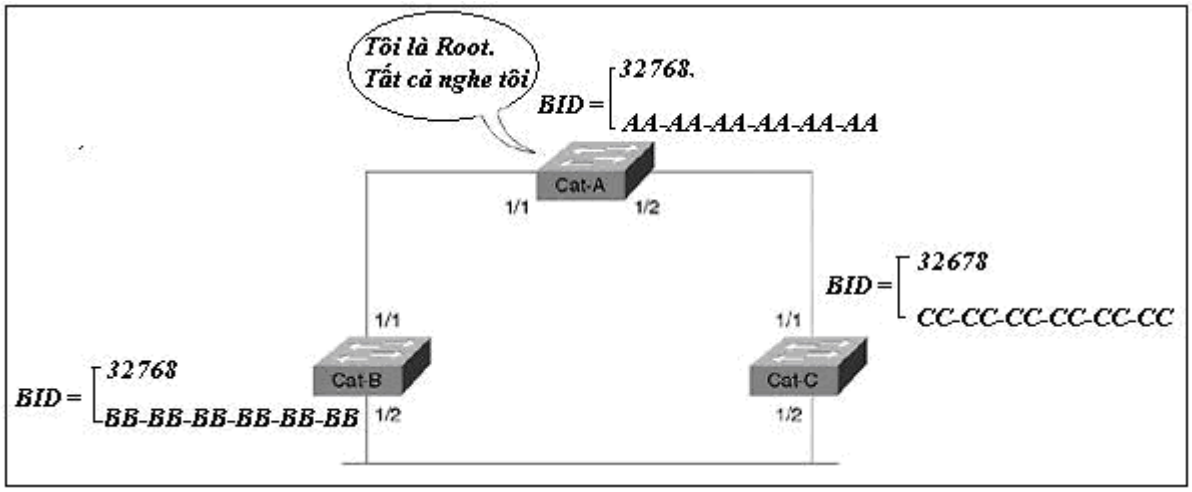
Bước 1: quyết định một bridge gốc.

Đầu tiên các switch cần chọn một bridge gốc bằng cách tìm bridge có BID thấp nhất.

Chú ý: nhiều tài liệu sử dụng tính ưu tiên cao nhất khi nói đến kết quả của quá trình chọn bridge gốc. Tuy nhiên, bridge với tính ưu tiên cao nhất thực tế có giá trị thấp nhất. Để tránh nhầm lẫn, tài liệu này luôn đề cập đến giá trị thấp nhất.

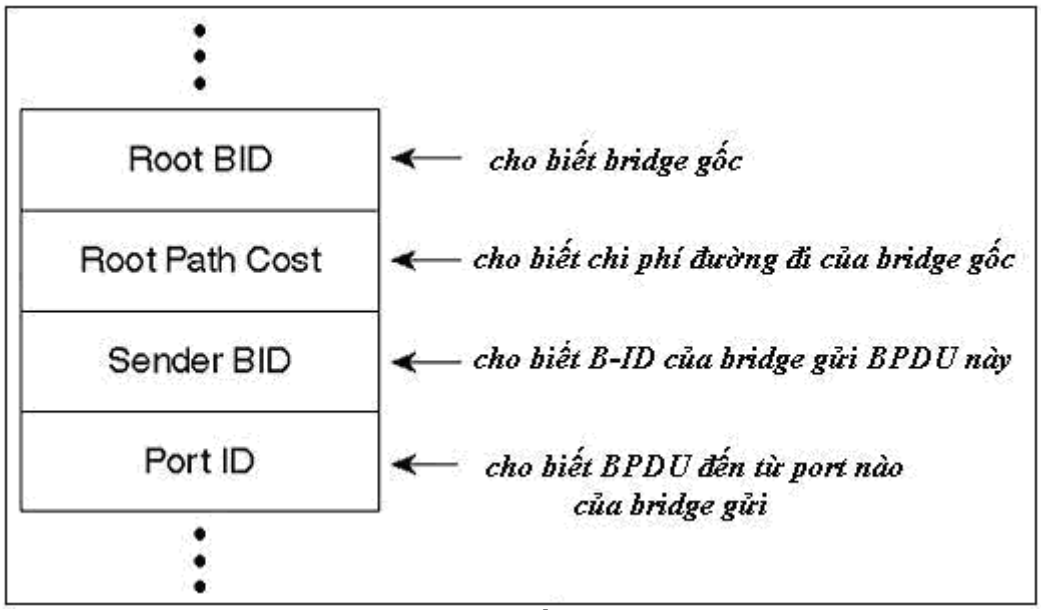
Như đã nói đến ở phần trên BID là một định danh 8 byte được chia thành 2 trường con là Bridge Priority và địa chỉ MAC từ người giám sát (supervisor) hoặc backplane. Trở lại *hình 3.6*, ta thấy Cat-A có BID mặc định là 32.768 và địa chỉ MAC là AA-AA-AA-AA-AA-AA. Cat-B là (32.768, BB-BB-BB-BB-BB-BB) và Cat-C là (32.768, CC-CC-CC-CC-CC-CC). Vì cả ba bridge đều sử dụng Bridge Priority là 32.678 nên địa chỉ MAC thấp nhất là AA-AA-AA-AA-AA-AA và Cat-A trở thành Bridge gốc. *Hình 3.7* mô tả quá trình này.

Chú ý: giá trị BID cũng là thấp nhất.



Hình 3.7: chọn Bridge Root

Nhưng làm thế nào các bridge biết được Cat-A có BID thấp nhất? Đó là do việc trao đổi các BPDU. Bridge sử dụng BPDU dành riêng để thay đổi cấu trúc mạng và thông tin Spanning Tree lẫn nhau. Các BPDU được gửi mặc định 2s một lần. Các BPDU là lưu lượng bridge-to-bridge, nó không mang lưu lượng end-to-end. *Hình 3.8* mô tả các phần cơ bản của một BPDU.



Hình 3.8: các thành phần cơ bản của BPDU

Mục đích việc chọn bridge gốc chỉ liên quan đến trường Root BID và Sender BID. Khi một bridge phát ra một BPDU 2s một lần, ngay tức khắc nó sẽ xác định bridge gốc dựa vào trường Root BID. Bridge này luôn đặt BID của chính nó trong Sender BID.

Chú ý: Root BID là ID của bridge gốc hiện tại, trong khi Sender BID là ID của bridge cục bộ hoặc switch.

Khi bridge khởi động lần đầu tiên, nó luôn luôn đặt BID trong cả hai trường Root BID và Sender BID. Giả sử rằng, Cat-B khởi động đầu tiên và bắt đầu gửi các BPDU thông báo chính nó là Bridge gốc 2s một lần. Một vài phút sau Cat-C khởi động và thông

báo chính nó là Bridge gốc. Khi BPDU của Cat-C đến Cat-B, Cat-B sẽ loại bỏ BPDU vì nó có B-ID thấp hơn được lưu trên các cổng của nó. Ngay lập tức Cat-B truyền BPDU, Cat-C biết được là giả định ban đầu của nó là sai. Tại thời điểm đó, Cat-C bắt đầu gửi BPDU với Root BID là B và Sender BID là C. Bây giờ mạng chấp nhận B là Bridge gốc.

5 phút sau đó, Cat-A khởi động, nó giả sử rằng nó là bridge gốc và bắt đầu quảng bá điều này trong BPDU. Ngay lập tức các BPDU đến Cat-B và C, các switch này sẽ nhường bridge gốc lại cho Cat-A. Bây giờ tất cả 3 switch đều gửi các BPDU thông báo Cat-A là bridge gốc và chính nó là Sender BID.

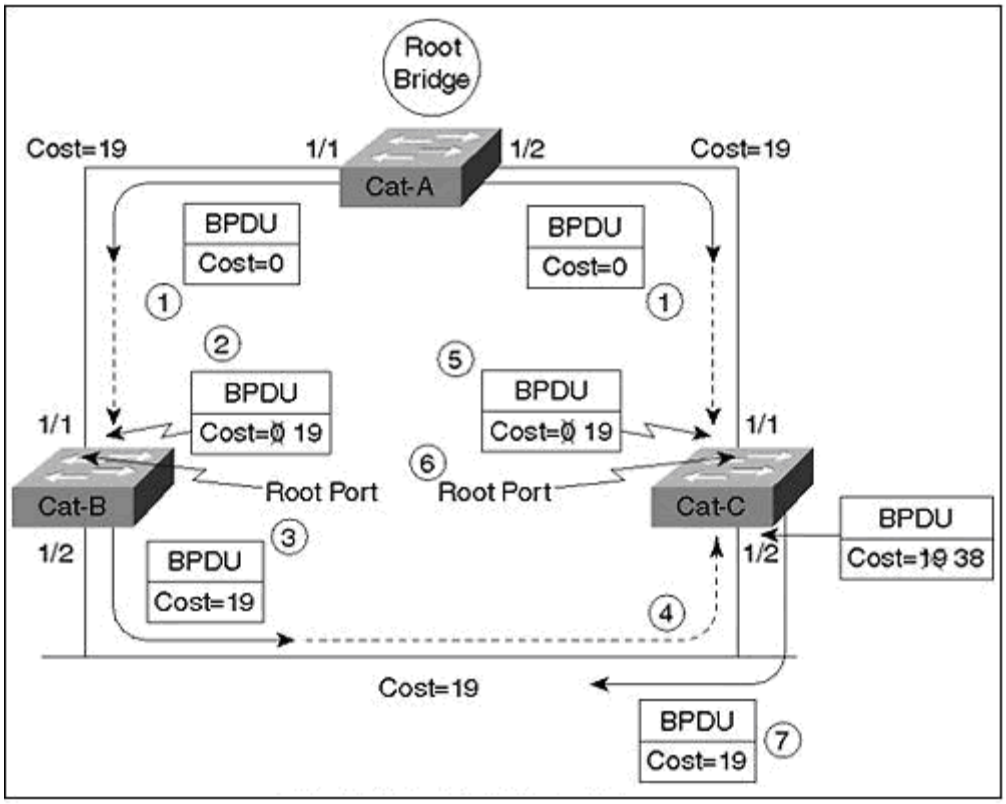
Bước 2: chọn cổng gốc.

Sau khi xác định được bridge gốc, thì các switch sẽ chuyển qua chọn cổng gốc. Cổng gốc là một cổng trên bridge cục bộ. Mỗi brigde (trừ bridge gốc) phải lựa chọn một cổng gốc.

Chú ý: Mỗi bridge (trừ bridge gốc) sẽ lựa chọn cổng gốc.

Bridge sẽ sử dụng khái niệm chi phí để xét cổng gốc. Cụ thể là các bridge theo dõi chi phí đường đi gốc, chi phí tích lũy của tất cả các liên kết đến bridge gốc. *Hình 3.9* mô tả làm thế nào tính toán qua nhiều bridge và kết quả của việc quyết định cổng gốc.

- (1): khi Cat-A (bridge gốc) gửi các BPDU, thì nó chứa chi phí đường đi gốc là 0.
- (2): khi B nhận các BPDU này, nó thêm vào chi phí đường đi của cổng 1/1vào chi phí đường đi gốc chứa trong BPDU nhận. Giả sử rằng mạng đang chạy switch Catalyst 5000 có mã lớn hơn phiên bản 2.4 và ba liên kết trong *hình 3.9* đều là Fast Ethernet. Cat-B nhận chi phí đường đi gốc là 0 và thêm vào chi phí của cổng 1/1 là 19.
- (3): sau đó Cat-B sử dụng giá trị 19 và gửi BPDU với chi phí đường đi gốc là 10 ra cổng 1/2.
- (4): khi Cat-C nhận BPDU này từ B, thì nó tăng chi phí đường đi gốc thành 38 (19+19).
- (5): tuy nhiên Cat-C cũng nhận BPDU từ bridge gốc trên cổng 1/1. Cat-C sẽ thêm vào cổng 1/1 với chi phí là 0, và ngay lập tức nó tăng chi phí lên 19.
- (6): Cat-C thấy chi phí đường đi gốc là 19 trên cổng 1/1 và 38 trên cổng 1/2, nó quyết định cổng 1/1 là cổng gốc (chọn giá trị nhỏ nhất).
- (7): sau đó Cat-C bắt đầu quảng bá chi phí đường đi gốc với giá trị 19 đến các switch xuôi dòng.



Hình 3.9 : chọn Root Port

Hình 3.9 biểu diễn Cat-B tính toán và chọn ra cổng 1/1 là cổng gốc với chi phí là 19, và chú ý là khi một cổng nhận BPDU thì chi phí sẽ tăng dần.

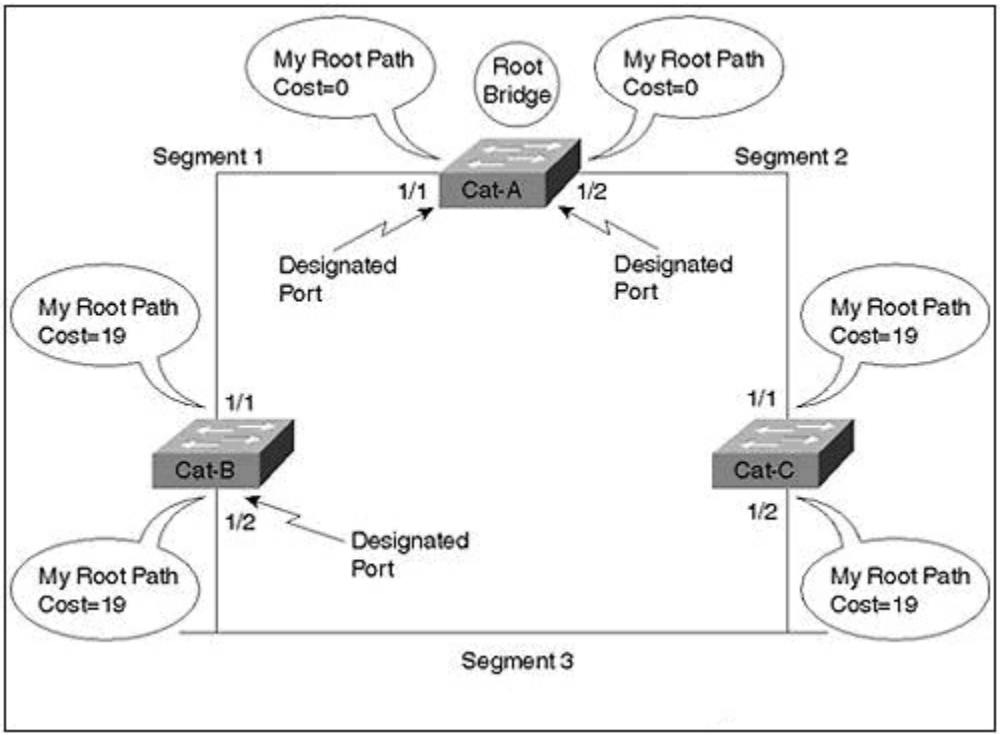
Chú ý:

- Chi phí STP được tăng khi một cổng nhận BPDU, chứ không phải vì nó được gửi ra khỏi cổng. Ví dụ như, các BPDU đến trên cổng 1/1 của Cat-B với chi phí là 0 và tăng lên 19 bên trong Cat-B.
- Sự khác nhau giữa chi phí đường đi và chi phí đường đi gốc.
- Chi phí đường đi là giá trị được gán cho mỗi cổng, nó được thêm vào các BPDU được nhận trên cổng đó để tính toán chi phía đường đi gốc.
- Chi phí đường đi gốc là chi phí tích lũy đến bridge gốc. Trong BPDU, đây là giá trị của trường chi phí. Đối với một bridge, giá trị này được tính bằng cách cộng các chi phí đường đi của các cổng nhận với giá trị chứa trong BPDU.

Bước 3: quyết định cổng được chỉ định.

Mỗi đoạn mạng trên một bridge có một cổng được chỉ định, cổng này có chức năng nhận và gửi lưu lượng đến đoạn mạng kia và bridge gốc. Nếu chỉ có một cổng nắm giữ lưu lượng trên mỗi liên kết, thì tất cả vòng lặp bị phá bỏ. Bridge chứa cổng được chỉ định được gọi là designated bridge cho đoạn mạng đó.

Việc lựa chọn cổng được chỉ định cũng dựa trên chi phí tích lũy của đường đi gốc đến bridge gốc (*hình 3.10*).



Hình 3.10: chọn Designated Port

Để xác định cổng được chỉ định, ta hãy nhìn vào mỗi đoạn mạng. Đầu tiên là đoạn 1, liên kết giữa Cat-A và B có 2 cổng là Cat-A: cổng 1/1, và Cat-B: cổng 1/1. Cổng 1/1 của Cat-A có chi phí đường đi gốc là 0, và cổng 1/1 của B là 19 (giá trị 0 được nhận trong BPDU từ A cộng với chi phí đường đi được gán cho cổng 1/1 của B). Vì cổng 1/1 của A có chi phí đường đi thấp hơn nên nó trở thánh cổng được chỉ định đối với liên kết này.

Đối với đoạn mạng 2 (kiên kết giữa Cat-A và C), tương tự cổng 1/2 của A trở thành cổng được chỉ định. Chú ý là mỗi cổng hoạt động trên bridge gốc đều trở thành cổng được chỉ định.

Bây giờ hãy xem đoạn 3 (liên kết giữa Cat-B và C), cả hai cổng 1/2 của B và 1/2 của C đều có chi phí đường đi gốc là 19. Đây là một sự hạn chế, và STP thường sử dụng trình tự bốn bước để quyết định:

- B-ID gốc thấp nhất.
- Chi phi đường đi đến bridge gốc thấp nhất.
- Sender BID thấp nhất.
- ID của cổng thấp nhất.

Trong ví dụ ở *hình 3.10*, tất cả các bridge đều tán thành Cat-A là Bridge gốc, cả B và C đều có chi phí là 19, nên ta sẽ lấy yếu tố BID để quyết định. BID của B là (32.768.BB-BB-BB-BB-BB-BB) và của C là (32.768.CC-CC-CC-CC-CC-CC), do đó cổng 1/2 của B là Cổng được chỉ định cho đoạn 3.

Ví dụ trong một mạng chứa 15 switch và có 146 đoạn mạng (mỗi cổng là một đoạn mạng duy nhất), số thành phần STP hiện có là

Các thành phần STP	Số
Bridge gốc	1
Cổng gốc	14
Cổng được chỉ định	146

Bảng 3.2: Các thành phần STP trong mạng có 15 switch và 146 đoạn mạng

Tất cả các quyết định STP đều dựa trên một trình tự như đã đề cập:

- BID gốc thấp nhất.
- Chi phí đường đi đến bridge gốc thấp nhất.
- Sender BID thấp nhất.
- ID của cổng thấp nhất.

Khi một cổng nhận BPDU nó sẽ so sánh với các BPDU nhận được trên các cổng khác (cũng như BPDU được gửi trên cổng đó). Chỉ BPDU tốt nhất mới được lưu lại. Tốt nhất ở đây có nghĩa là giá trị thấp nhất (ví dụ như BID thấp nhất trở thành Bridge gốc, giá trị thấp nhất cũng được sử dụng để chọn cổng gốc và cổng được chỉ định). Một cổng sẽ ngưng truyền BPDU nếu nó nghe được một BPDU tốt hơn BPDU của nó.

3.5 Các trạng thái của STP

Sau khi bridge phân chia được các cổng như cổng gốc, cổng được chỉ định và cổng không được chỉ định, thì việc tạo ra cấu trúc mạng chứa loop-free không phức tạp lắm, cổng gốc và cCổng được chỉ định chuyển tiếp lưu lượng, trong khi cổng không được chỉ định thì khóa lưu lượng. Việc chuyển tiếp và khóa chỉ là 2 trạng thái thông thường trong mạng, *bảng 3.3* mô tả 5 trạng thái của STP.

Trạng thái	Mục đích
Chuyển tiếp (forwading)	Gửi và nhận dữ liệu người dùng
Học hỏi (learning)	Xây dựng bảng bridge
Lắng nghe (listening)	Xây dựng cấu trúc mạng "active"
Khóa (blocking)	Chỉ nhận các BPDU
Vô hiệu hóa (disable)	Các cổng bị down

Bảng 3.3: Các trạng thái của STP

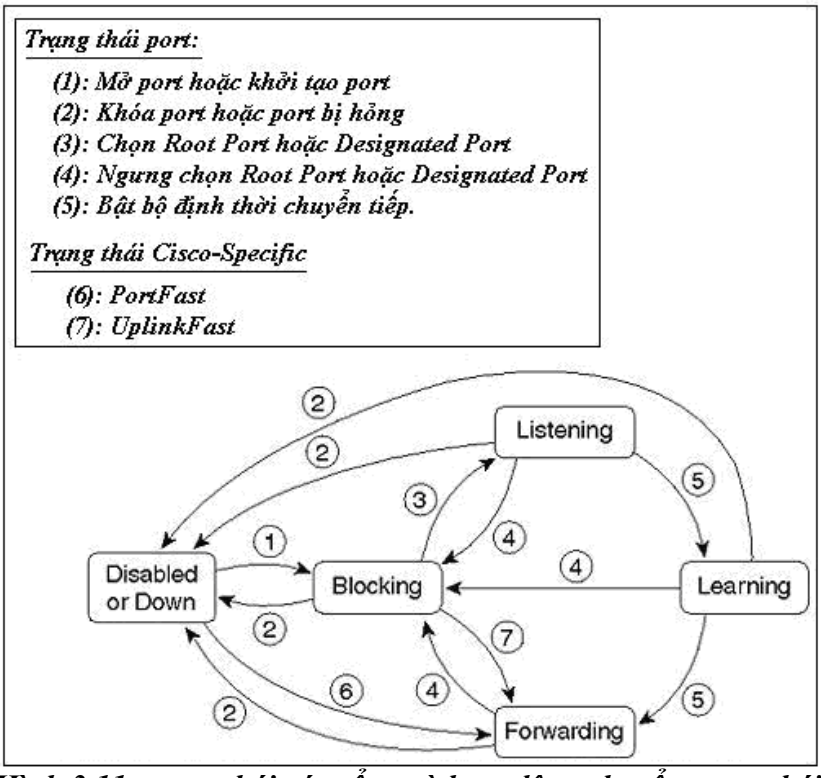
Trạng thái "disable" cho phép người quản trị mạng quản lý việc ngừng hoạt động của một cổng. Sau khi khởi tạo, các cổng bắt đầu trong trạng thái "blocking" để lắng nghe các BPDU.

Do sự đa dạng của các sự kiện mà bridge truyền trong trạng thái "listening" (ví dụ như một bridge nghĩ nó là bridge gốc ngay sau khi khởi động). Ở trạng thái này, không có dữ liệu người dùng được truyền qua, tức là cổng đang gửi và nhận các BPDU để cố gắng tạo cấu trúc mạng hoạt động. Trong trạng thái "listening" sẽ sử dụng ba bước hội tụ đã nói ở trên, các cổng bị mất quyền cổng được chỉ định sẽ trở thành cổng không được chỉ định và trở lại trạng thái "blocking".

Các cổng được chỉ định và cổng gốc sau 15s (giá trị mặc định của bộ định thời) sẽ chuyển qua trạng thái "learning". Trong khoảng 15s khác, bridge vẫn không chuyển các frame của người dùng qua, mà xây dựng bảng bridge của nó. Khi bridge nhận frame, nó đưa địa chỉ MAC và cổng vào bảng bridge. Trạng thái "learning" sẽ giảm bớt số lượng tràn ngập khi việc chuyển tiếp dữ liệu bắt đầu.

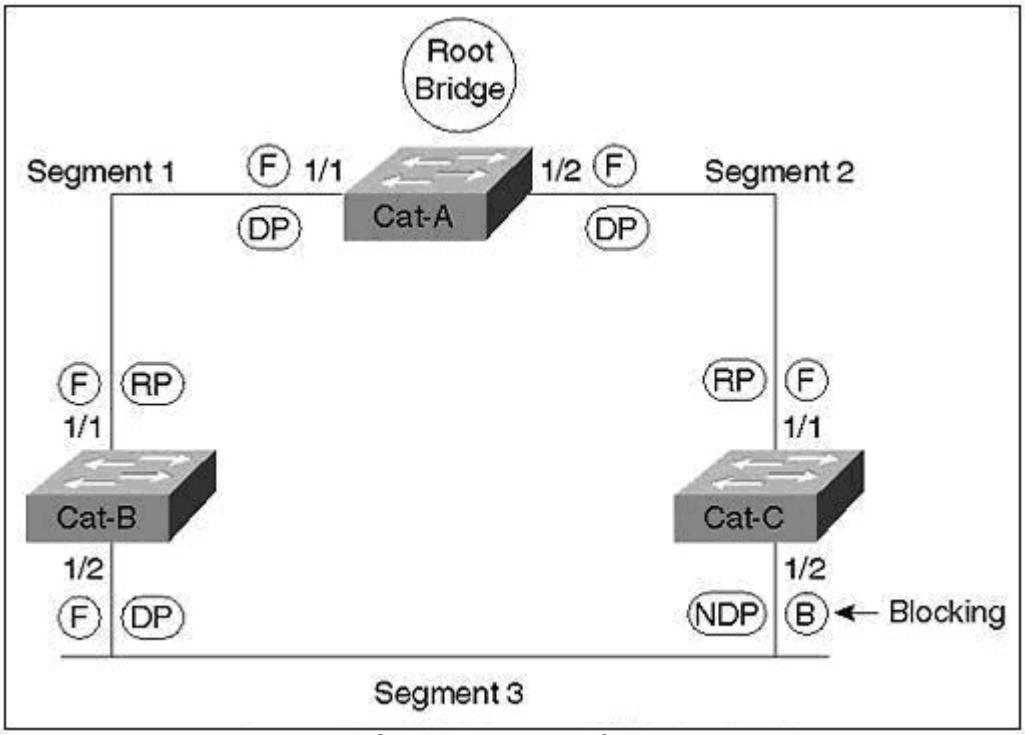
Chú ý: Trong việc lưu trữ địa chỉ MAC và thông tin cổng, các Catalyst học các thông tin như VLAN nguồn.

Nếu một cổng vẫn là cổng được chỉ định hay cổng gốc ở khoảng thời gian cuối của trạng thái "learning", thì cổng chuyển qua trạng thái "forwading". Ở trạng thái này, nó bắt đầu gửi và nhận các frame của người dùng. *Hình 3.11* mô tả trạng thái các cổng và việc chuyển trạng thái.



Hình 3.11: trạng thái các cổng và hoạt động chuyển trạng thái

Hình 3.12 biểu diễn mạng với sự phân chia cổng và danh sách các trạng thái. Chú ý là tất cả các cổng đều chuyển tiếp trừ cổng 1/2 của Cat-C.



Hình 3.12: sơ đồ mạng với các cổng được định danh

Trạng thái/cổng	Ký hiệu
Blocking	B
Forwading	F
Cổng được chỉ định	DP
Cổng gốc	RP
Cổng không được chỉ định	NDP

Bảng 3.4 : Các trạng thái STP và các ký hiệu cổng

3.6 Bộ định thời gian STP

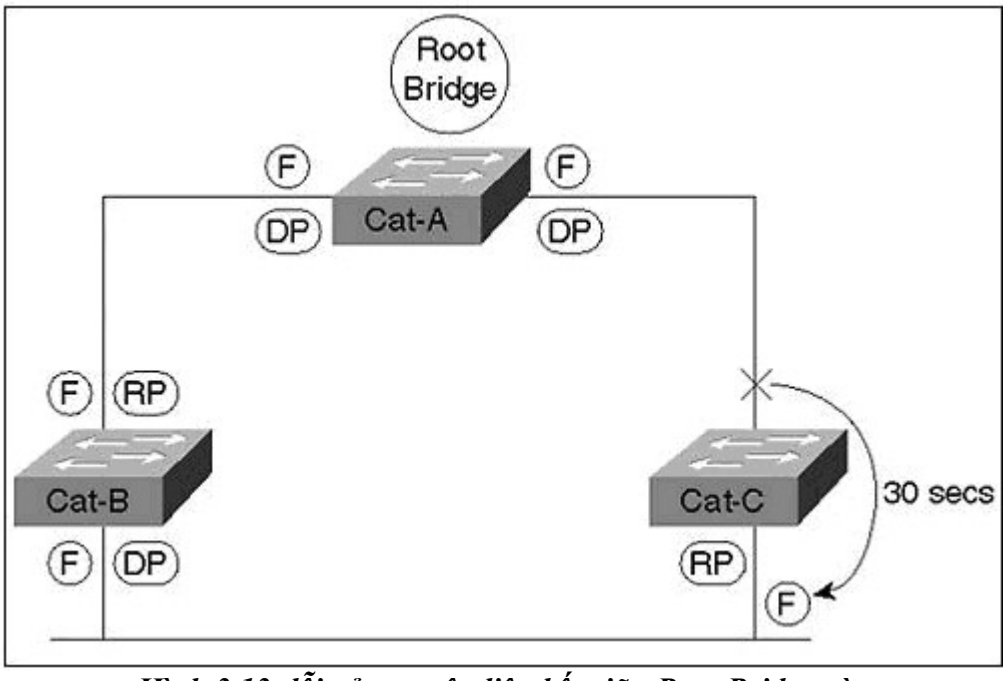
Một bridge trải qua 15s ở mỗi trạng thái "listening" và "learning". STP được điều khiển bởi ba bộ đếm thời gian (timer) như trong *bảng 3.5*.

Timer	Mục đích	Giá trị mặc định
Hello Timer	Khoảng thời gian gửi các BPDU cấu hình gửi bởi Bridge gốc	2s
Forward Delay	Thời hạn ở trạng thái Listening và Learning	15s
Max Age	Thời gian lưu trữ BPDU	20s

Bảng 3.5: STP Timer

Ví dụ: giả sử rằng liên kết đoạn 3 trong *hình 3.12* sử dụng một hub và cổng 1/2 của Cat-B truyền ra ngoài. Cat-C không thông báo lỗi liền vì nó vẫn đang nhận liên kết Ethernet từ hub. Cat-C chỉ thông báo là các BPDU ngừng đến. Sau 20s (Max Age), thì cổng 1/2 của Cat-C lấy thông tin BPDU cũ với cổng 1/2 của Cat-B là cổng được chỉ định cho đoạn mạng 3. Điều này làm cho cổng 1/2 của Cat-C truyền trong trạng thái "listening" để cố gắng trở thành cổng được chỉ định. Vì vậy cổng 1/2 của Cat-C cung cấp truy cập tốt nhất từ bridge gốc đến liên kết này, nên nó chuyển sang trạng thái "forwarding". Như vậy, Cat-C mất 50s (20s Max Age + 15s Listenning + 15s Forwarding) để vượt qua sau khi cổng 1/2 trên Cat-B bị lỗi.

Trong trường hợp này, các bridge có thể phát hiện sự thay đổi cấu trúc mạng trên các liên kết kết nối trực tiếp và ngay lập tức chuyển sang trạng thái "listening" mà không cần chờ thời gian Max Age. Xem ví dụ trong *hình 3.13*.



Hình 3.13: lỗi xảy ra trên liên kết giữa Root Bridge và Root Port của Cat-C

Trong trường hợp này, cổng 1/1 của Cat-C bị lỗi, vì liên kết trên cổng gốc cũng bị lỗi nên ngay lập tức cổng 1/2 của Cat-C chuyển sang trạng thái "learning" để trở thành cổng gốc mới thay vì chờ 20s rồi mới lấy thông tin cũ. Điều này làm cho thời gian hội tụ STP giảm từ 50s xuống 30s (15s listening + 14s learning).

Chú ý: thời gian hội tụ STP là từ 30s đến 50s.

Hai điểm quan trọng cần nhớ khi sử dụng bộ định thời STP là:

- Thứ nhất: không thay đổi giá trị thời gian mặc định khi không có sự cân nhắc cẩn thận.
- Thứ hai: ta chỉ được sửa thời gian từ bridge gốc.

3.7 Hai loại BPDU

Có hai loại BPDU là :

- BPDU cấu hình.
- BPDU thông báo thay đổi cấu trúc mạng – TCN BPDU (Topology Change Notification BPDU).

BPDU cấu hình được bắt đầu bởi bridge gốc và phát ra trên các con đường hoạt động từ bridge gốc, còn TCN BPDU hướng về bridge gốc để cảnh báo với bridge gốc là cấu trúc mạng mạng có sự thay đổi.

BPDU cấu hình : các trường trong BPDU cấu hình được tóm tắt trong *bảng 3.6*

Trường	Chiều dài (octet)	Ý nghĩa
Protocol ID	2	Luôn bằng 0
Version	1	Luôn bằng 0
Type	1	Cho biết kiểu BPDU BPDU cấu hình = 0
Flag	1	LSB = Cờ thay đổi cấu trúc mạng MSB = Cờ xác nhận thay đổi cấu trúc mạng
Root ID	8	BID của bridge gốc hiện tại
Root Path Cost (chi phí đường đi gốc)	4	Chi phí tích lũy đến bridge gốc
Sender BID	8	BID của bridge hiện tại
Cổng ID	2	ID của cổng gửi BPDU này
Message Age	2	Khoảng thời gian tử khi bridge gốc tạo BPDU đến khi phát BPDU đi.
Max Age	2	Khoảng thời gian lưu thông tin BPDU
Hello Time	2	Khoảng thời gian giữa các BPDU
Forward Delay	2	Thời gian trong trạng thái listening và learning

Bảng 3.6 : Các trường trong BPDU cấu hình

TCN BPDU (Topology Change Notification BPDU) :

TCN BPDU đơn giản hơn BPDU cấu hình và chỉ gồm có ba trường, giống như ba trường đầu tiên của BPDU cấu hình nhưng trường Type thì thay đổi với giá trị như sau :

- 0x00 (0000 0000): BPDU cấu hình.
- 0x80 (1000 0000): TCN BPDU.

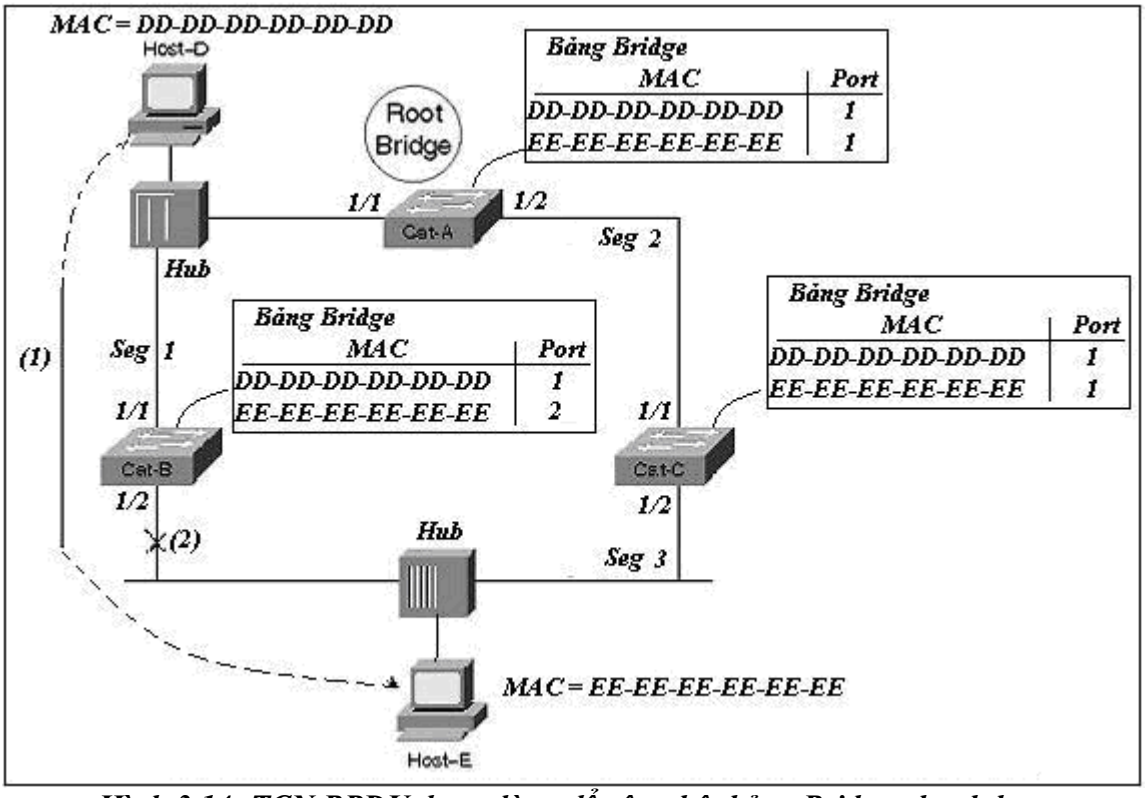
Chú ý : TCN BPDU không mang bất cứ thông tin bổ sung nào.

3.8 Quá trình thay đổi cấu trúc mạng

Nếu TCP BPDU đơn giản thì làm thế nào nó thể hiện được vai trò quan trọng của nó? Ta hãy xem xét sự thay đổi cấu trúc mạng trong *hình 3.14*.

Host D đang liên lạc với host E qua hai bước:

- (1): lưu lượng từ host D qua Cat-B để liên lạc với host E.
- (2): giả sử bộ thu phát trên cổng 1/2 của Cat-B bị hỏng.



Hình 3.14: TCN BPDU được dùng để cập nhật bảng Bridge nhanh hơn

Như đã thảo luận, cổng 1/2 của Cat-C mất 50s để trở thành cổng được chỉ định. Tuy nhiên nếu không có TCN BPDU thì nó tiếp tục bị ngắt khoảng 250s. Trong khoảng thời gian lỗi, cả ba switch đều chứa địa chỉ MAC của host E trong bảng Bridge như bảng 3.7.

Bảng Bridge	Cổng liên quan đến địa chỉ MAC của host E
Cat-A	Cổng 1/1
Cat-B	Cổng 1/2
Cat-C	Cổng 1/1

Bảng 3.7: Giá trị bảng Bridge trước khi có sự thay đổi cấu trúc mạng

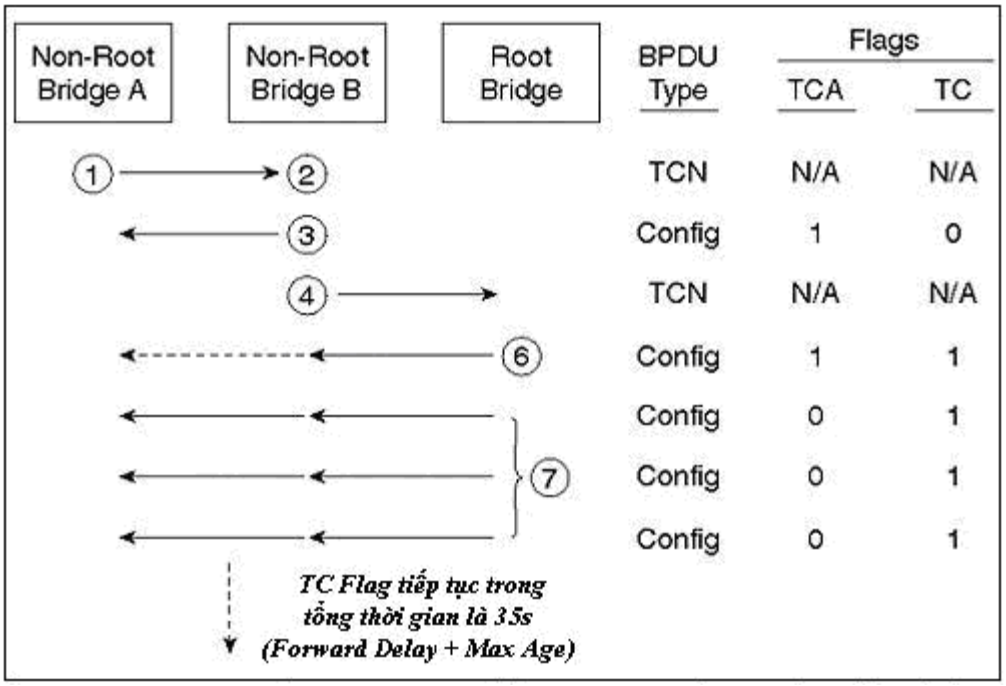
TCN BPDU là một phương pháp đơn giản để cải tiến thời gian hội tụ, và nó làm việc chặt chẽ với BPDU cấu hình như sau:

1. Một bridge bắt đầu một TCN BPDU khi:
    - Nó chuyển một cổng sang trạng thái "forwarding" và nó có ít nhất một cổng được chỉ định.
    - Nó chuyển một cổng từ trạng thái "forwarding" hoặc "learning" sang blocking.

Sự thay đổi cấu trúc mạng đòi hỏi phải gửi thông báo đến bridge gốc, giả sử rằng bridge hiện tại không phải là bridge gốc, thì nó bắt đầu quá trình thông báo bằng cách gửi TCN BPDU ra cổng gốc của nó. Nó tiếp tục gửi TCN BPDU cho đến khi thông điệp TCN được xác nhận.

2. Bridge upstream sẽ nhận TCN BPDU. Mặc dù, một vài bridge nghe được TCN BPDU (vì nó kết nối trực tiếp vào cổng gốc của đoạn mạng) nhưng chỉ có cổng được chỉ định chấp nhận và xử lý TCN BPDU.
3. Bridge upstream sẽ thiết lập cờ xác nhận thay đổi cấu trúc mạng TCA (Topology Change Acknowledgement) trong BPDU cấu hình kế tiếp được gửi ngược lại(ra cổng được chỉ định). Cờ này dùng để xác nhận với bridge khởi đầu để nó ngưng phát TCN BPDU.
4. Bridge upstream sẽ truyền TCN BPDU ra cổng gốc của nó.
5. Tiếp tục bước 2 đến bước 4 cho đến khi bridge gốc nhận TCN BPDU.
6. Sau đó bridge gốc sẽ thiết lập cờ xác nhận thay đổi cấu trúc mạng – TCA (để xác nhận với bridge trước đó), và cờ thay đổi cấu trúc mạng – TC (Topology Change) trong BPDU cấu hình mà nó sẽ gửi đi.
7. Bridge gốc tiếp tục thiết lập cờ thay đổi cấu trúc mạng – TC trong tất cả các BPDU cấu hình mà nó gửi ra ngoài với tổng thời gian là 35s (Forward Delay + Max Age). Cờ này sẽ thu ngắn giá trị 300s xuống 15s (tức độ trễ chuyển tiếp - Forward Delay).

Hình 1.15 tóm tắt các bước trong quá trình thay đổi cấu trúc mạng.



Hình 3.15: trình tự các bước trong quá trình thay đổi cấu trúc mạng

Dựa vào *hình 3.15* ta có thể biết được quá trình thay đổi cấu trúc mạng cho *hình 3.14* như sau: bước 1 Cat-B và C gửi TCN BPDU ra cổng 1/1. Vì bridge upstream cũng là bridge gốc nên bỏ qua bước 3 và 4. Sau đó bước 2 và 5 xảy ra đồng thời. Trong BPDU cấu hình kế tiếp mà bridge gốc gửi đi, cờ TCN ACK sẽ được thiết lập để xác nhận là đã nhận TCN của hai brigde downstream. Tiếp theo là bước 6 và 7, Cat-A cũng thiết lập cờ TA trong 35s (Forward Delay + Max Age) để cập nhật bảng bridge nhanh hơn. Như vậy cả ba switch đều nhận được cờ TA và khoảng thời gian cho bảng bridge là 15s.

Chú ý là khoảng thời gian ngắn 15s này không bắt buộc cho toàn bộ bảng, nó chỉ làm quá trình này nhanh hơn thôi. Các thiết bị tiếp tục nói suốt 15s này mà không cho bảng bridge nghĩ. Tuy nhiên, nếu host D cố gắng gửi một frame đến host E trong 20s (giả sử host E không nói gì hết), thì frame sẽ được tràn đến tất cả các đoạn mạng vì địa chỉ EE-EE-EE-EE-EE-EE không còn có trong bảng bridge nữa. Ngay khi frame đến host E và host E trả lời, thì switch học được giá trị bảng bridge mới tương ứng với cấu trúc mạng mới.

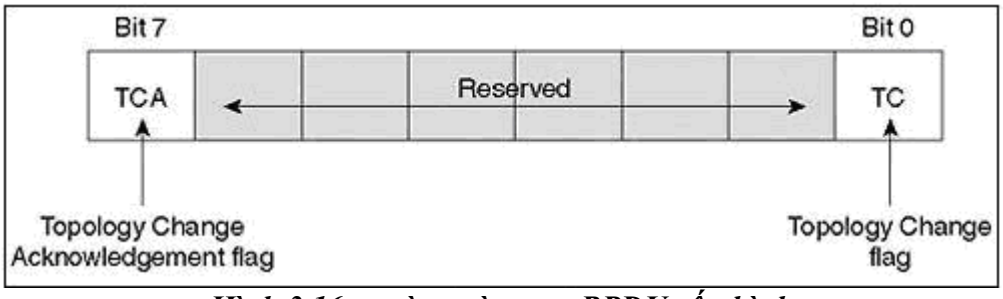
Bảng 3.8 biểu diễn toàn bộ bảng bridge cho địa chỉ MAC của E trên cả ba switch sau khi cấu trúc mạng mới hội tụ và lưu lượng lại tiếp tục.

Bảng Bridge	Cổng liên quan đến địa chỉ MAC của host E
Cat-A	Cổng 1/2
Cat-B	Cổng 1/1
Cat-C	Cổng 1/2

Bảng 3.8: Giá trị bảng Bridge sau khi thay đổi cấu trúc mạng

Tại thời điểm này, kết nối giữa host D và E đã được thiết lập lại và lưu lượng lại tiếp tục. Chú ý là TCN BPDU giảm thời gian lỗi từ 300s (5ph) xuống 50s.

Hình 3.16 mô tả trường cờ trong BPDU cấu hình, cả hai cờ TCA và TA đều được lưu trữ trong cùng một octet của BPDU cấu hình.



Hình 3.16: trường cờ trong BPDU cấu hình

Như đã thảo luận, cờ TCN được thiết lập bởi bridge upstream để nói cho các bridge dowstream ngưng gửi TCN BPDU. Còn cờ TC được thiết lập bởi bridge gốc để giảm khoảng thời gian lỗi từ 300s xuống 15s (Forward Delay).