Multiple Symmetric Spanning Trees

Mick Seaman

Direct layer 2 communication between stations attached to a Virtual Bridged Local Area Network is usually supported by a single VLAN. Pairs of VLANs, using the same spanning tree and shared VLAN learning †1, are occasionally used to segregate traffic and provide high end scalability †2. This note explains how to use shared VLAN learning for a set of VLANs running over *different* trees, thus providing shortest paths between a number of bridges, with a bi-section bandwidth that is not limited to the number of links that can be trunked between two switches †3. This use of shared learning does require the use of the same FID by VLANs allocated to different spanning trees †4, but otherwise conforms to the .1Q specification for existing SVL/IVL bridges, and is likely to be supported by their frame forwarding hardware †5. Sets of bridges providing MSST shortest paths can form part of a network that supports the more conventional IVL and SVL uses of VLANs, though all bridges have to use a slightly modified version of MSTP if the shortest paths between MSST bridges lie outside their transitive closure. Further development of an MSTP like protocol may be desirable, depending on the applications for MSST.

1. Introduction

Address learning bridges, such as those specified in IEEE Standards 802.1D and 802.1Q, depend on symmetric paths between the stations they connect: a frame from station *a* to station *b* traverses the same bridges and LANs as a frame from *b* to *a*, only in the reverse direction. This is trivially true if the traffic in each direction is confined to the same spanning tree, but is also true if:

- traffic from *a* is confined to a spanning tree rooted in the bridge (*A*, say) that it is immediately attached to
- traffic from *b* is confined to a spanning tree rooted in the bridge (*B*) that *it* is immediately attached to
- the same path costs are assigned to each LAN in the calculation of the spanning trees for *A* and *B*
- there is a unique lowest cost path from *A* to *B*.

For, if the last two bullets are true, the lowest cost path from *B* to *A* will be the exact reverse of the *A* to *B* path, and then, if the first two are true, the *a*,*b* path will be symmetric. Part of this note explains how the same path can be selected in both directions if there are equal cost paths, but the basic idea and its consequences are explained first.

There is nothing in the foregoing that prevents a station *c* immediately attached to a bridge *C* from also communicating with *a* and *b* over pairwise shortest paths. Assume each of *a*, *b*, and *c* sends frames that are not VLAN tagged, and these are tagged on ingress by their bridges using PVIDs A, B, C, and untagged on egress Edge Ports (so the end stations don't have to know anything about VLANs, or which path their frames will take). If each of these VLANs is supported by a distinct tree rooted at the ingress bridge and all three VLANs share the same FID†6, then Figure 1 illustrates traffic flows, address learning, and tree configuration, for a very simple network.

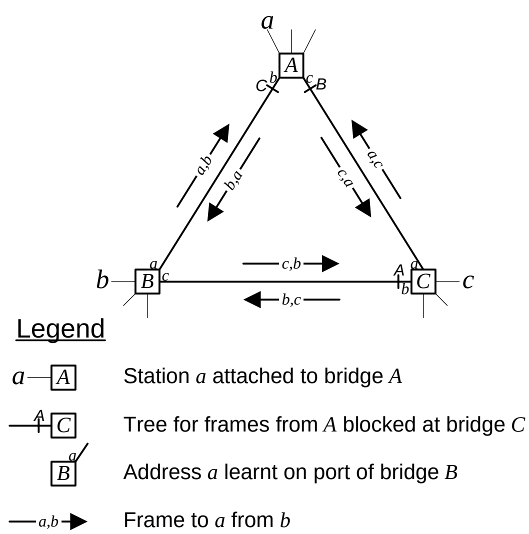

Figure 1—Shortest paths in a simple network

The rest of this note describes scalability, interoperability with existing equipment, handling of equal cost paths, network application areas, and the further development of MSTP or similar protocols to support MSST.

†6 IEEE speak for addresses learnt for a frame on any of the VLANs affects forwarding on all of them.

†1 802.1Q 8.8.3 and Annex B.

†2 .../docs2003/ScalableQinQLearning.pdf

†3 This is really a cheap shot, since the bandwidth between practical cuts in the set of bridges is, in a well designed network, limited by the bandwidth provided by a central bridge (spared for redundancy) which may be many times created than the maximum trunked bandwidth to another switch.

†4 In violation of a requirement of 802.1Q-REV clause 8.6.1.

†5 Whether any particular bridge implementation can support MSST, as described in this note, depends on whether VIDs are mapped to FIDS before spanning tree state is applied. My guess is that most can, since using that as an optimization is (a) not obvious (b) conflicts with other requirements for per VLAN rather than per VID state—such as, are frames with this VID in the member set for this Bridge Port? and (c) does not help with the worst case requirement for spanning tree state, which is that each VID has a separate FID.

2. Scalability, interoperability, and applicability

Figure 2 shows a slightly more complex MSST network, with four bridges (*A*, *B*, *C*, and *D*) that support a Symmetric VLAN Set (SVS). At the top of figure the entire network is shown (or at least as much of it that interest us) together with the cuts in the physical topology that ensure that each of the four trees (and therefore the corresponding VLANs) rooted at those bridges provides a fully connected (spanning) loop-free (tree) active topology. Each of the four active topologies are shown lower in the figure. Looking at the active topology for tree A and its VLAN (or VLANs) it is easy to see that there is no cut in that topology for B's tree between A and B, for C's tree between A and C, or for D's tree between A and D. This can be confirmed by checking the active topologies for each of those trees, verifying that there is full symmetric connectivity between each pair of bridges in the SVS.

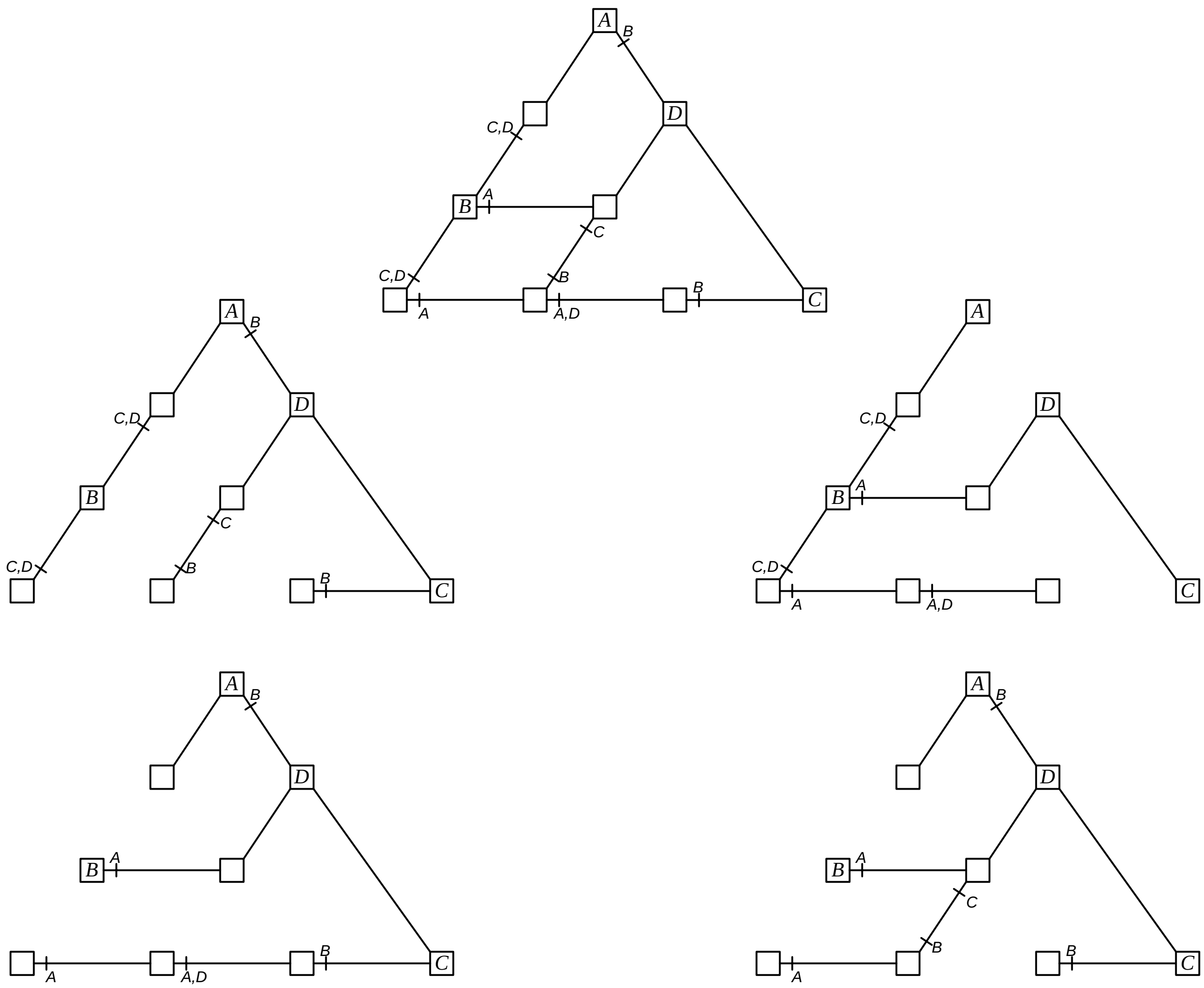

Figure 2—Multiple symmetric trees in a network

A number of other observations come to mind:

1) *A*, *B*, *C*, and *D* do not have to be connected directly to each other.
2) The same network can continue to support the independent and shared learning VLANs that are already in use today, without change.
3) Other symmetric VLAN sets can be supported, independently, over the same network at the same time. Each could, for example, support a distinct IP subnet without a requirement for distinct physical separation.
4) Each SVS makes no assumptions about the behavior of higher layer protocols not already made by bridges.
5) The end stations that benefit from the shortest path connectivity across the MSST region do not have to be directly connected to a bridge at the boundary of that region. Their traffic may simply follow a spanning tree (the CST or an MSTI) to and from an MSST Bridge Port.
6) In particular, an MSST Region can function exactly as an MST Region does today, autoconfiguring over the CST.
7) Unicast and multicast traffic between members of an SVS flow over exactly the same path, so control traffic for higher layer protocols is not divorced from the data it controls, and diagnostic tools that use a multicast to trace a unicast path from bridge to bridge still work.
8) While the broadcast domain for an SVS naturally includes all the MSST bridges in the SVS, dynamic GVRP/MVRP pruning can still be carried out to reduce the set of links that see the broadcast/multicasts to those currently interconnecting SVS members. Each member bridge registers for the VLANs from each of the others over its own multicast distribution tree.

9) The address learning optimizations described elsewhere and now incorporated in P802.1ad still apply. Where address learning for frames on one VLAN in the SVS would not affect forwarding for the others through a given bridge, those addresses do not have to be learnt by that bridge.

10) Although the costs of each link need to be the same for all trees supporting a given SVS, costs for each SVS can be different, thus providing load balancing capability between SVSs.

11) Spanning tree configuration is independent of the number of end stations, so the dynamics of bridging remain unchanged.

12) While an SVS has been described as being fully connected, its inter-connectivity can be subsetted in the same way as for pairwise shared learning VLANs, if that is desired. A number of useful and interesting configurations are possible.

13) The number of VLANs and trees used scales linearly with the number of MSST bridges, unlike solutions that use VLANs to mimic point-to-point circuits. So with 32 or 64 trees we can put together a quite impressive network core.

The total number of end stations that a given MSST region can support with shortest path networking is therefore somewhere between the maximum supportable by a single bridge, and that number multiplied by the number of SVS regions. For high end bridges, and without expending much additional effort on MSTP enhancements, that number is probably somewhere between one hundred thousand and three million.

In practice the real scalability of MSST is likely to be limited by the scale desired in network applications where the restriction to equal link costs for all VLANs in an SVS is not burdensome. Clearly it is more difficult to reconfigure the path taken from one MSST bridge to another, while leaving everything else in the network unchanged, than it is to manage MPLS routes †1. Network technology cannot be usefully compared without reference to its real use, so the desire to enhance the scalability of MSST beyond that easily achieved (probably in the region of 32 to 64 SVS members in a general mesh, although 256 on a ring should be fairly easy) should be driven by application requirements.

High bandwidth connection of a few hundred or a few thousand compute or file servers in a data center is one potential application, although the bandwidth requirements probably need to be in excess of 200 Gb/s, and not readily localized, for anything other than a simple redundant star using a pair of very large switches to be needed †2. One solution to such a high but simple bandwidth problem might be a larger version †3 of the network illustrated by Figure 3.

†1 Although some serious thinking about this subject would pay dividends. And of course it is possible to use MPLS to provide virtual connectivity between MSST bridges.

†2 I would love to have more hard application requirements in this area.

†3 I am sure you can think of a much better example.

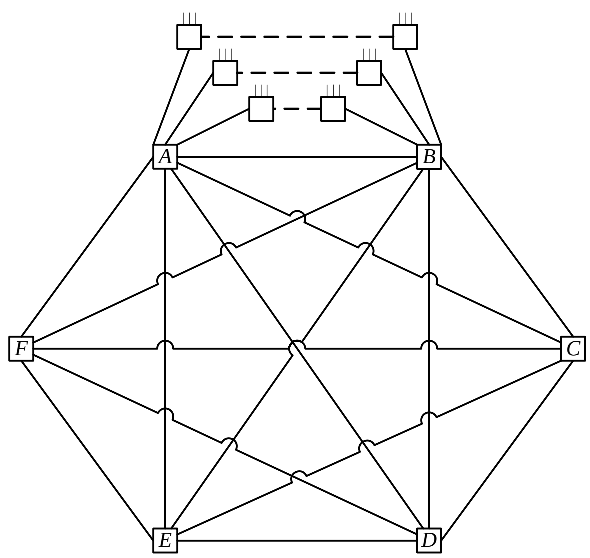

Figure 3—A simple high performance network core (part)

MSST Bridges *A* through *F* provide a fully connected core for the low cost bridges (just a few shown at the top) that actually attach to the end stations. Each of these bridges runs ordinary RSTP and has a backup connection to a similar switch that connects to another MSST core bridge. The network is thus protected against the failure of any single core bridge, while the loss of an end station attachment bridge just affects the directly connected end stations. The effective bandwidth and design of the network depends on the locality of communication. Assuming that each MSST bridge connects to each of its peers with 8 link aggregated 1Gb/s links and with a single 1Gb/s to each of 40 simple bridges, each of which attaches to 24 end stations, then slightly less than 6,000 end stations are connected with a core bandwidth of over 480 Gb/s. The total available bandwidth may be higher if there is significant communication between end stations attached to the same bridge†4.

†4 But how would they know?

3. Equal cost paths

The introductory section stated a requirement for a unique lowest cost path between any two of the MSST bridges. This section shows how this can pose a problem, and how the problem can be overcome with a modest enhancement to the existing Multiple Spanning Tree Protocol (MSTP).

All the existing spanning tree protocols use a local tie-breaker (bridge identifier, port identifier) to select between equal cost paths. This means that it is possible for the path from A to B (say) (using a tree rooted at A) to differ from that from B to A (using a tree rooted at B), even though the port path costs are identical for every link†1. See Figure 4.

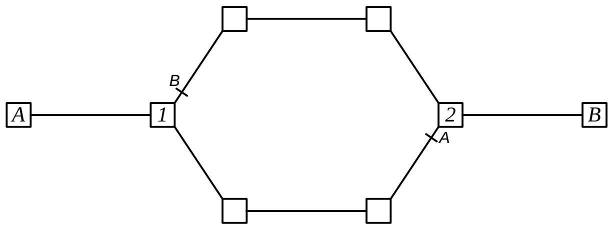

Figure 4—Non-symmetric equal cost paths

Independent application of a tie-breaker for tree A at bridge 2, and for tree B at bridge 1 has resulted in the non-symmetric path. One solution is to use a link state rather than distance vector algorithm as a basis of the multicast tree computation, thus allowing each bridge to see the whole network picture and choose coordinated tie breakers. The actual port state transitions can still be coordinated from bridge to bridge using the RSTP Proposal/Agreement mechanism so that a loop is never created. A simpler solution is to add a ‘cut vector’, with one bit per tree in the SVS, to the information propagated for each tree and to order the trees for the purposes of making tie-break decisions. The remainder of this section describes this approach.

As information for tree B passes through bridge 2 towards A, the cut bit for A is set in the information propagated on the lower path. The cut bit is ignored by any bridge that can make decision purely on path cost, but when bridge 1 has to choose its Root Port it prefers the upper path (without the cut bit) to the lower. If the information from the Root Port chosen by a bridge does not have the cut bit set for any given tree, the cut bit is clear (for that tree) in information propagated through its Designated Ports. Note that the cut bit for a given tree (C, say) only matters for tie-breaks on a tree (F, say) if F is proceeding toward the root of C. Once F has passed C, i.e. if a bridge port that is Designated for F is also Designated for C, there is no need to propagate the cut bit for C on that port as it will never form part of a symmetric path between C and F. The requirement for coordinated tie-breaking is thus limited to choices between alternate Root Ports, and does not affect Designated Port selection.

Once a port has been chosen in a tie-breaker, the same choice should be made for any lower trees (in the same SVS) for which the same tie-breaking choice has to be made. This short circuits convergence, as choices for tree A affect those for tree B, and those for B affect C, and so on. See Figure 5 for an example.

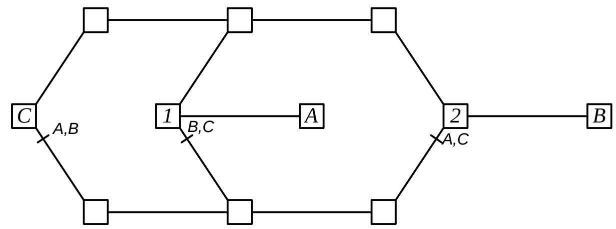

Figure 5—Coordinated equal cost path cutting

Fortunately a tie-break change in a tree, preferring the Root Port to an equal root path cost Alternate Port, has a purely local effect on the tree. The spanning tree priority vectors propagated through Designated Ports do not change, no ports have to change Port State etc. Coordinating equal path cost cutting use a cut bit vector required one more propagation time across the network, but does not cause reflecting ripples in the way that attempting to dynamically tweak link costs to avoid equal cost paths would.

MSTP is intentionally limited to 64 trees, and all the MSTP information that needs to be transmitted through a given Bridge Port at any instant will fit in a legal sized Ethernet frame. Unfortunately the addition of 64 cut bit vectors for each of the trees would exceed the limit. I propose that we allow a maximum of 32 bridges in any SVS, with up to 32 SVSs†2, which fits. It is possible to extend MSTP to use multiple PDUs, as previously proposed, but our previous requirements discussions would seem to indicate that there would be little demand for an SVS of more than 32 bridges. I am considering a slightly different extension that could help with providing shortest paths over ring media, with a maximum of 256 hops around the ring.

†1 A less significant problem is that of two bridges connected to the same LAN assigning different costs to that LAN. This can be fixed for point-to-point to point links by adding in half the cost for the Root Port and half the cost for the Designated Port, instead of only adding Root Port Path Costs to the Root Path Cost.

†2 The maximum possible for 64 trees.