Layer 2 Connectivity: Bridge Spanning Tree Protocol

Burton Rosenberg

December 30, 2002

Introduction

To do: Motivating comments.

We model a bridged network as a bipartite graph, $G = (B, L, E)$ with labeled edges. The notation: B is the collection of bridges; L the collection of links; and E are the interconnections of bridges to links via ports, $(b, l) \in E$ with $b \in B$, $l \in L$. Each edge is labeled with the port number of the interconnection. There can be multiple edges with different labeling, when a bridge connects through many ports to the same link.

Each bridge has a unique identifier which is composed of a hardware assigned unique number, e.g. a MAC address from among its interfaces, and a two byte administrator assigned prefix. The purpose of this prefix is to allow the administrator to prefer certain bridges as root bridges. Port identifiers are formed by prefixing the port number with a one byte administrator assigned prefix, for a similar purpose.

To do: It is unfinished business to describe the default values for the prefixes.

Spanning Tree Algorithm

Bridges multicast configuration messages drawn from the set

$$\mathcal{C} = B \times \Delta \times B \times p$$

with the following meaning: $(r, d, b, p) \in \mathcal{C}$ is a message sent by bridge b on port p to indicate that it believes r to be the root bridge and that b is distance d from r (counting number of bridges to cross to reach r, including b). The occurrence of a $b, r \in B$ in the configuration message is represented by the bridge identifier for bridge b, r, and p similarly represented by its the port identifier.

The bridge configuration messages are linearly ordered lexicographically, in the natural way by component. (See Figure *Configuration Message Ordering.*) Each bridge has a current assumed root r_b and a found minimum distance to the root d_b. Initially $r_b = b$ and $d_b = 0$. Each port writes its current configuration message, $c(b,p) = (r_b, d_b, b, p)$ to its connected link. The link computes and posts the best (lowest in lex order) received and non-expired configuration message $c(l_j)$.

Note: It is a mathematical fiction that the link can compute and store the configuration message. In reality this is done for the link by each port of each bridge, which selects the best incoming configuration message for the link. All ports sharing a link should agree and therefore this value is truly property of the link, rather than of the port. We adopt this fiction to ease algorithm analysis in the non-error case. Error mode operation may require that this simplification be revisited.

```
compare( (r, d, b, p), (r', d', b', p') ) {
   if ( r != r' ) return compare( r, r' )
   if ( d != d' ) return compare( d, d' )
   if ( b != b' ) return compare( b, b' )
   if ( p != p' ) return compare( p, p' )
   return EQUAL
}
```

Figure 1: *Configuration Message Ordering*

Each bridge port reads the link's current configuration message and updates it's own current configuration message. (See Figure *Current Configuration Update.*) This process is iterated while the bridge is in listening and learning modes until the configuration messages stabilize.

```
update( b, p ) {
    let (r_b,d_b,b,p) = c(b,p)
    let (r',d',b',p') = min { c(l) | (b,l) in E}
    if ( r' < r_b OR ( r' == r_b AND d'<d_b ) )
         r_b = r'
         d_b = d'+1
}
```

Figure 2: *Current Configuration Update*

Leaving learning and listening mode, each bridge now decides which ports will be on the spanning tree. Other ports will be blocked. (See Figure *Spanning Tree Decision.*) Spanning tree ports are of two types, root ports and designated ports. Traffic is accepted from the root port and any designated port and forwarded to the root port and all designated ports, except the port on which the traffic was received.

The root bridge will have only designated ports and blocked ports. If a port is blocked on a root bridge, another port connecting this bridge to the link will be designated. All other bridges will have a root port and zero or more designated ports. A bridge having no designated ports will not, in effect, use its root port, since there is no other forwarding ports.

Each link will be connected to exactly one designated port. It is said that the bridge with this port is the designated bridge for the link. The remaining ports on the link will be blocked or root. Among all ports connecting a bridge to a link, all except one will be blocked. For instance, if a

```
decide( b, p ) {
    let l be the link connected to b on port p
    if ( c(l) == c(b,p) ) then return DESIGNATED
    let l* be the link minimizing c(l') for any link l' connected to b on any port
    Assert: c(l*) < c(b,p)
    let p* be the minimum port on b connecting to link l*
    Assert: p* exists, since l is connected to p
    if ( p == p* ) then return ROOT
    else return BLOCKED
}
```

Figure 3: *Spanning Tree Decision*

bridge is designated for a link, it has no root port to the link, since all other ports to this link are blocked.

Analysis and Robustness

We assume that the spanning tree algorithm has completed successfully. Under this assumption, we replace the bipartite graph (B, L, E) representing the network with a directed bipartite graph (B, L, E^*). Each edge of the undirected graph is labeled root, designated or blocked according to whether it represents a port connecting bridge to link which is root, designated or blocked. To create the directed edge set E^*: add all root edges of E to E^* directed from bridge to link; add all designated edges of E to E^* directed from link to bridge. Omit from E^* any blocked edges of E.

Since each bridge has is at most one root port, exactly one edge leaves any bridge node, except for the root, which has no root port. Also, each link has exactly one designated bridge, so exactly one edge leaves any link node. Since every edge leaves from some node, bridge or link, $|E^*| = |B|+|L|-1$. Hence the directed graph is either cyclic or a tree. There are not cycles, since every root edge goes towards a strictly lower configuration message. Hence the result is a tree.

To do: Robustness, convergence, hello messages, and so on. Plenty to do.

References

1. Radia Perlman, INTERCONNECTIONS: BRIDGES AND ROUTERS, Chapter 3.