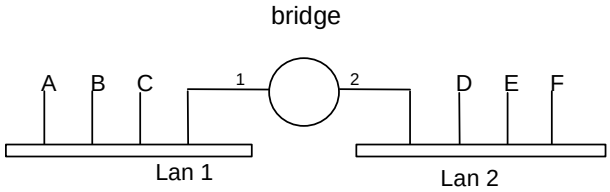
ECE 333: Introduction to Communication Networks
Fall 2002

Lecture 20: Medium Access Control VIII

- Bridges and interconnected LANs
  - § Transparent Bridges - spanning trees

Bridges

Last time we began talking about bridges. Recall, bridges are devices for interconnecting 2 or more LANs at the MAC layer. When a bridge receives a packet, it must know whether or not to forward the packet and if so on which LAN. This is a basic example of a **routing problem** - we will see other examples in lectures 22-24 at the network layer. Consider two LANs connected by a bridge as shown below. If the bridge receives a transmission on LAN 1 for node D, it should retransmit that packet on LAN 2, but it should not retransmit a packet for node A received from LAN 1.



There are two basic solution approaches to this problem. One approach is to simply have a look-up table, which specifies which nodes are on which LANs, such a table is called a ***forwarding table*** or ***forwarding database***. When a packet is received the bridge looks for the destination in the table and decides to forward the packet or not. Alternatively, each packet could have additional information in the header that identifies the destination LAN.

With either of the above approaches, a key issue is how this information (either in the forwarding table or in the packet header) comes to be known. One possibility would be to have a system administrator keep all of this information up-to-date, but for large LANs this is an undesirable approach. Instead, various algorithms are used to acquire this information.

There are two basic types of bridges (both also standardized by the IEEE 802 working group). The main conceptual difference between these approaches is in how the routing problem is handled.

Transparent Bridge (specified in IEEE 802.1)

Accepted as standard for interconnection of any 802.x LANs.
Typically used with Ethernet LANs
Use forwarding table approach.

Source Routing Bridge:

Developed separately by 802.5 committee (Token Ring)
Modified to also connect 802.3 & 802.4
Put forwarding information in packet header.

In the following we take a closer at the transparent bridge approach; for a discussion of source routing bridges see Sect. 6.7.2 in Leon-Garcia.

Transparent Bridges

Transparent bridges were designed to be a *plug-and-play* system - i.e., a transparent bridge can be simply be connected to two or more LANs without making any changes to the stations attached to the LANs or requiring any configuration on the part of the users. Everything should work automatically. Thus the bridge must learn the location of stations in the LANs and build up its forwarding database.

A transparent bridge operates in ***promiscuous mode***, this means it receives all packets transmitted on the LANs to which it is attached. Each packet contains both source and destination addresses. When a packet is received, the bridge associates the port on which the packet was received with the source address to create the forwarding table. (This is called ***backward learning***.)

When the bridge receives a packet, it forwards it if the destination is not reachable on the incoming LAN. If the destination is not in the forwarding table, then the bridge forwards the packet on every LAN, except the one it received the packet on.

Difficulties with Learning

Two things make the above problem somewhat more difficult:

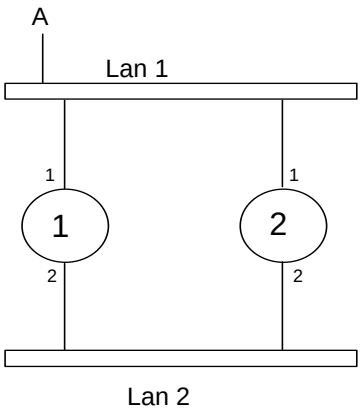
1. Stations can be moved from one LAN to another.

2. There may be loops in the topology.

The first problem is addressed by including an age field in the forwarding table for each entry. This age field is updated whenever a new frame from that station arrives. Information is discarded if it is too old.

Loops

Often extra bridges are used to increase reliability. However, this can create loops in the topology:

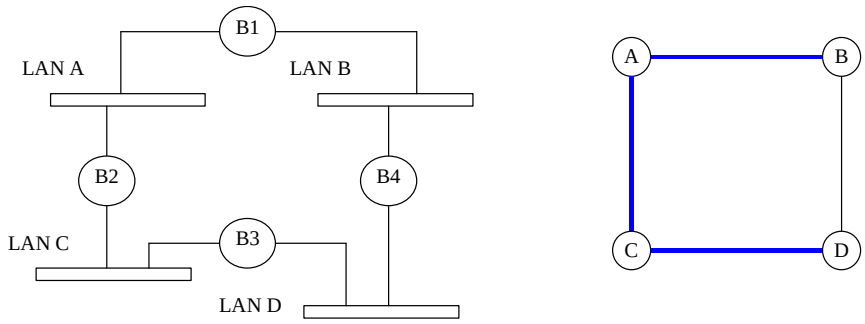


Suppose **node A** sends a packet with a destination address that is not in either bridges forwarding table. In this case, both bridges would forward to LAN 2. Each packet arrives at the other bridge on LAN 2 and is forwarded to LAN 1, etc.

Spanning Trees

To avoid packets being broadcast indefinitely as above, transparent bridges use an algorithm to construct a ***spanning tree***. A spanning tree is a loop-free connection of all LANs in the network. We can represent the network as a graph[1], where each node in the graph corresponds to a LAN and arcs connect any two LANs that are connected by a bridge. A spanning tree is a loop-free sub-graph that connects all the nodes.

Example: A network of bridges and LANs is shown on the left below, the corresponding graph is shown on the right. One spanning tree is to include only the arcs that are highlighted. Notice there may be several possible spanning trees for a graph.



[1] Actually this representation may be a multi-graph, because there may be multiple edges between a pair of nodes.

Spanning tree algorithm

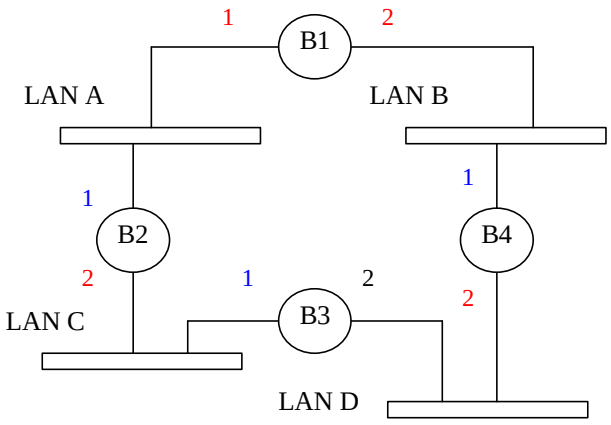
To decide on a specific spanning tree, the bridges exchange control information and execute a distributed algorithm called appropriately the spanning tree algorithm. For the spanning tree algorithm, each bridge is assigned a unique ID number. Each port on a bridge is also given an ID that is unique for that bridge. The spanning tree algorithm executed by the bridges, attempts to do the following:

1. Elect a ***root bridge,*** this will be the bridge with the smallest ID.
2. Determine a ***root port*** for each bridge (except the root), this is the port that is on the ***shortest path*** to the root bridge. Distance is measured as the number of LANs a packet would have to cross to reach the root bridge. Ties are broken by choosing the port with the smallest ID.
3. Select a ***designated bridge*** for each LAN. This is the bridge which provides the shortest path to root bridge from the LAN. Ties are broken by choosing the bridge with the smallest ID. The port that connects a LAN to its designated bridge is called the ***designated port***.

Each bridge then blocks any port that is not a root or designated port, *i.e.* it does not forward any packets to a LAN attached to such a port. This removes in any loops in the topology and forms a spanning tree.

Example:

The figure below shows a collection of LANs and bridges, the bridges are numbered 1,2,3,4, the port numbers for each bridge are also indicated. In this case B1 would be the root bridge, since it has the smallest ID. The designated ports are shown in red and the root port are shown in blue. For example, on LAN C, bridge B2 is the designated bridge, since it is only one LAN away from the root, while B3 is 2 LANs away. Notice that port 2 on bridge B3 is neither a root bridge nor a designated bridge. Thus bridge B3 will not forward any packets, creating a tree.



Spanning Tree Algorithm

To construct and maintain the above spanning tree, the bridges exchange a sequence of control messages called *Bridge Protocol Data Units (BPDU's)*.

At any time, each bridge stores the following two values in memory:

- **Root_ID** - the bridge's estimate of the root.
- **Root_distance** - the bridge's estimate of its distance to the presumed root.

The bridges then transmit BPDU with the format:

[Sender_ID|Root_ID|Root_distance]

Where **Sender_ID** is the bridge's ID own ID number.

Initially all bridges think they are the root (**Root_ID=Sender_ID**). A bridge updates its values of **Root_ID** and **Root_distance** based on the messages received to date.

Specifically, assume that a bridge has **Root_ID** = ***R*** and **Root_distance** = ***D***. If it receives a BPDU containing [***S***'|***R***'|***D***'] on a port connected to LAN *i*. Then it does the following:

IF ***R***' < ***R*** THEN set **Root_ID** = ***R***', and **Root_distance** = ***D***' +1, and assume not designated bridge on LAN *i*.

IF ***R'*** = ***R*** and ***D'*** < ***D*** THEN set **Root_distance** = ***D'***+1, and assume not designated bridge on LAN *i*.

IF ***R'***= ***R***, and ***D'***=***D*** and ***S'*** < **Sender_ID**, THEN assume not designated bridge on LAN *i*

ELSE assume it is designated bridge on LAN *i*.

Bridges also update their designation on other LAN's to which they are connected, based on previous messages received on those LANs*, e.g.,* if it had previously received [***S"***|***R"***|***D"***] on LAN *j*, and now **Root_ID** < ***R"***, then the bridge will assume it is now designated on LAN *j*.

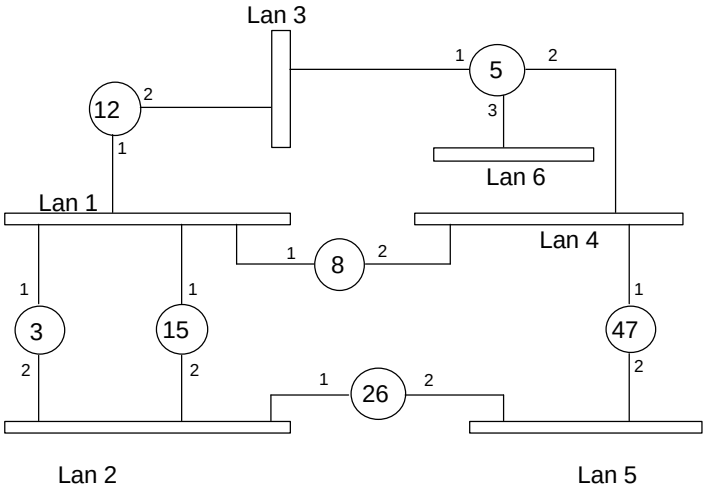
Bridges transmit BPDU's according to the following rules:

1. When a bridge thinks it's the root (**Root_ID** = **Sender_ID**), it periodically transmits BPDU.
2. When a bridge learns it is not root, it only forwards BPDU's (after adding one to the distance).
3. Bridges only forward BPDU's on LANs where it thinks it is the designated bridge.

But note even if a bridge is not designated it still receives all BPDU's transmitted on all LANs it is attached to.

An example of this algorithm is presented next. To simplify this example, we assume that all the bridges are synchronized and send out their BPDUs at the same time. In practice this algorithm would be implemented asynchronously.

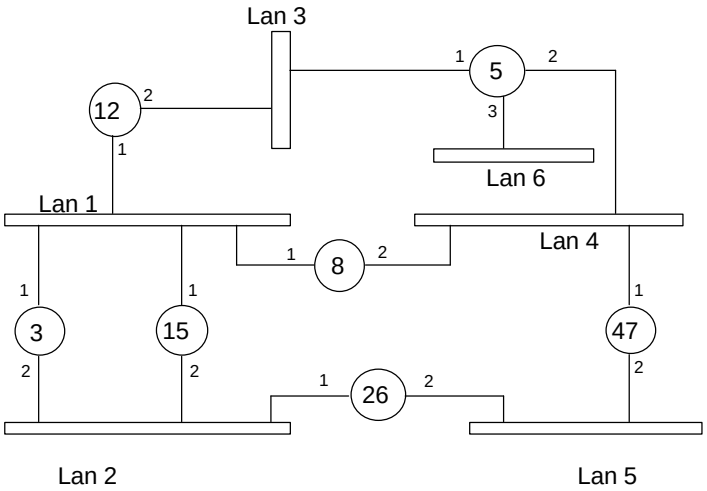
Spanning Tree Example



Initially:

- All bridges assume they are the root
- All bridges assume they are the designated bridge on all LAN's to which they are connected
- All bridges send a BPDU to that effect.

After lst set of BPDU's



Bridge 3

- receives claims of root-hood from Bridges 15, 8, 26, 12
- keeps **Root_ID** = 3, **Root_distance** = 0.
- maintains its belief that it is designated on LAN 1 and LAN 2.

Bridge 5

- received BPDU's from Bridges 12, 8, 47
- keeps **Root_ID** = 5, **Root_distance** = 0.
- maintains its belief that it is designated on LAN's 3, 4, and 6

Bridge 12

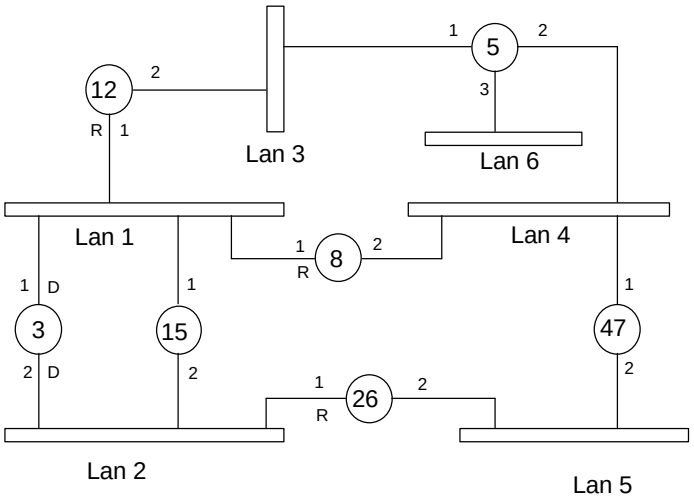
- received claims of root-hood from Bridges 3, 15, 8, 5
- sets **Root_ID** = 3, **Root_distance** = 1.
- sets Bridge 3's as designated on LAN 1
- sets Port 1 as root port
- sets port 2 as designated port on LAN 3

NOTE: Subsequent BPDU will not be sent toward root by Bridge 12.

Bridge 15

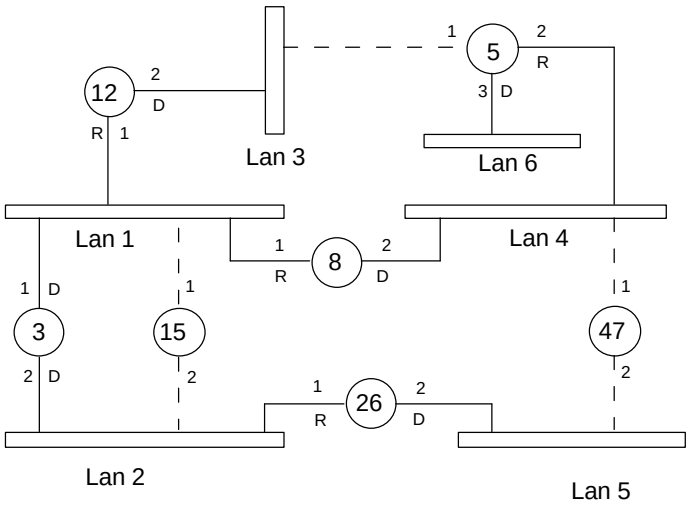
- receives claims of roothood from 3, 12, 8, and 26
- sets **Root_ID** = 3, **Root_distance** = 1.
- sets bridge 3 as designate bridge on LAN 1 and LAN 2
- Since it is connected to no other LAN, it goes into a listen mode.

Spanning Tree Example After 1st Iteration



- Bridges 3, 15, 8, 26, 12 know correct root
- Bridges 5 and 47 think bridge 5 is the root.
- Next, bridges 3 and 5 will send out BPDU's; the other bridges will forward these on ports that are not towards the root or blocked.
- In future rounds, Bridge 15 will not send out or forward any BPDU's

Spanning Tree Example



Final spanning tree (solid lines)

Recovery from Loss of Bridge

Bridges can also detect and recover from the loss of a bridge. To accomplish this, the root node is required to send out BPDU's at regular intervals. The other bridges forward these according to the above rules. Each bridge has timer that times out if it doesn't get an update from root. When a bridge times-out, it assumes it is the root bridge again, and starts the algorithm over.

Summary of Medium Access Control:

- Static approaches - TDMA, FDMA
- Dynamic approaches -
    - ♦ Contention based -
        - o Aloha, CSMA, CSMA/CD (Ethernet)
        - o Wireless - MACA, MACAW, (802.11)
    - ♦ Non-contention based -
        - o Reservations
        - o Token rings - (802.4, 802.5, FDDI)
- Interconnection of LANs - bridges/LAN switches
    - ♦ Transparent bridges
    - ♦ Source routing bridges