9

Designing Bridged and Switched Networks

This chapter discusses the design of flat (nonhierarchical or Layer 2) networks, using devices such as bridges and switches. These devices are not, strictly speaking, internetworking devices but are commonly used in network design, mostly in campus-based local area networks (although they may support wide area interfaces also). We will focus on the following issues:

- Bridging mechanisms and applications
- Switch mechanisms and applications
- Virtual LANs (VLANS)

Previously, we discussed how to build local and campus area networks, and one of the key features introduced was the idea that LAN segments are constrained in both length and the number of stations that may be attached. Many of these LAN technologies also employ shared bandwidth schemes, where overall traffic levels are directly related to the number of nodes added. Regarding IEEE 802.3/Ethernet networks we also discussed the concept of the collision domain, which introduces further constraints on network growth.

Bridging was the first real tool for improving scalability in local area networks, and in the early 1980s bridging technology was (hard to believe now) at the cutting edge. Bridges were also used to build wide area networks, though thankfully these applications are now rare (for reasons we will discuss later). Bridges have been largely eclipsed nowadays by their faster, slicker offspring—switches and general-purpose multiprotocol bridge-routers. Bridging is actually still very much alive but now working in disguise.

9.1 Overview of bridging and switching

Previously, we focused primarily on media characteristics and cabling design issues. The device used for extended and distributing local area segments at the physical layer is referred to as a repeater. Repeater functionality is the most basic transmission facility for data networking and is today integrated into many network products. Figure 9.1 illustrates how repeaters have evolved into increasingly sophisticated bridging and switching products since the early 1980s.

One of the key differences between a repeater and a classical bridge or switch is that the latter two devices are store-and-forward devices (we will discuss the special case of cut-through switching shortly). In Ethernet-style networks, for example, repeaters transfer packets between interfaces while maintaining frame timing on both receive and transmit interfaces. This is achieved by removing leading bits from the preamble field (the so-called bit budget delay) as each new repeater hop is traversed. In effect the preamble field is nibbled away to ensure that the start of frame is synchronized on transmit interfaces. By contrast, bridges and repeaters store incoming frames into buffers, so that they may be processed more intelligently. This means that interfaces on multiport switches and bridges are attached to separate timing domains (or collision domains in the Ethernet sense). As a result of buffering, it is possible (through additional protocol functions) to entirely interconnect different types of local or wide area technology via a bridge or switch; something that is not practical on a repeater.

Performance also improves in bridged or switched Ethernet LANs, because overall traffic levels are lowered through filtering, and collision levels are also lowered (by the act of filtering bad frames and the fact that traffic levels are lower—reducing the probability of collisions). All of this means more bandwidth is available for data.

9.1.1 Bridging

In essence, a bridge is a semiintelligent buffered device that connects two or more similar or dissimilar LAN media types and may also support wide area interfaces (although wide area applications are now becoming far less common since the introduction of high-performance multiprotocol routers). If the bridge connects LAN segments of the same type (say Ethernet), then the task is relatively straightforward. If mixed media or mixed local and remote interfaces are supported, then the task is more complex (the bridge

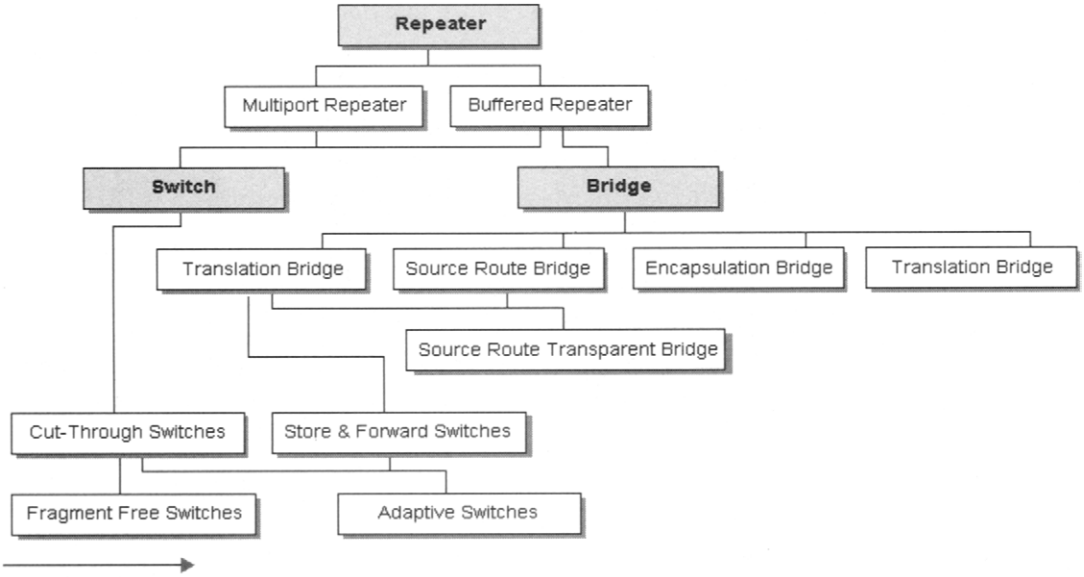

Figure 9.1 *Bridge and switch evolution from humble beginnings.*

must resolve factors such as bit ordering, addressing, MTU, differences in line speeds, etc.). There are also areas here where standards are simply not defined. Unlike a repeater, the main benefit of bridging is that it contains local traffic by filtering out traffic that it knows (through a learning process) to be local. Bridges also extend segments beyond the normal media limitations, since frames are buffered and retimed (on Ethernet networks the preamble will also be reinstated). Bridges also limit the scope of errors to an interface (e.g., short frames [runts] and CRC errors will not be forwarded to other interfaces). They can achieve this because they have much more time than repeaters to examine frames and make sensible filtering decisions.

There are various types of bridges, depending upon the interfaces offered, the media types supported, and the level of functionality supported, as follows:

- Local bridge—supports one or more LAN media types, such as Ethernet, Token Ring, and FDDI.
- Remote bridge—supports one or more LAN interfaces plus one or more WAN interfaces, such as T1/E1, ATM, and Frame Relay.
- Transparent bridge—operates without modifying frames in any way or interacting with end systems. Primarily associated with IEEE 802.3/Ethernet LANs.

- Spanning Tree bridge—a bridge that operates using the IEEE 802.1d Spanning Tree protocol for loop resolution (most standard bridges support some form of Spanning Tree operation).
- Source Routing Bridge (SRB)—a bridge that performs source routing operations by using the Token Ring Routing Information Field (RIF) to make forwarding decisions. Primarily associated with IEEE 802.5/ Token Ring LANs.
- Source Routing Transparent Bridge (SRT)—a bridge that combines both transparent and source routing functions concurrently.
- Source Routing Translation Bridge (SR-TB)—a bridge that performs complete translation at the Data Link and Physical Layers (currently for Token Ring and Ethernet networks only).
- Encapsulation bridge or tunnel bridge—a bridge that performs IP encapsulation of either source route bridging frames or transparent bridge frames in order to perform Layer 3 forwarding.

Bridges examine each incoming frame and, depending upon forwarding decisions, either copy the frame into local buffers or discard it. Forwarding knowledge is based on a process called learning in transparent bridge environments or implicit topology information contained within frames in the source routing environment. Bridges are Layer 2 devices; therefore, forwarding decisions are based on Media Access Control (MAC) addresses; there is no visibility of the Layer 3 IP address space.

9.1.2 Switching

The term switching was originally used to describe wide area technologies such as X.25, Frame Relay, and SMDS; however, LAN switching has evolved from bridging and multiport repeater techniques. LAN switches are designed for high-performance, high port density station attachment, as well as the classic media extension role performed by bridges. Just like bridges, LAN switches are designed to work with existing cable infrastructures transparently, so that they can be installed with minimal disruption to existing networks. Switches also construct MAC address filter tables for use in forwarding decisions, in essentially the same way as bridges. There are various types of switches, depending upon the level functionality supported, including the following:

- Store-and-forward switch—sometimes called classical switching. Effectively this is a high-performance multiport bridge. It has all the

characteristics of a standard bridge (such as buffering complete frames and limiting frame errors to an interface).

- Cut-through switch—a hybrid switch with some bridge and repeater-like functionality. Since the destination MAC address appears first in a frame, this device can make a forwarding decision very quickly by buffering only the first six bytes of the frame (not the whole frame as in a bridge). In effect, forwarding can take place while the frame is still arriving, enabling very high performance. The problems associated with cut-through switches are that the act of forwarding while receiving means that frame errors (such as runts, pygmies, giants, collision fragments, alignment errors, and FCS failures) are also propagated.
- Fragment-free switch—a cut-through switch with additional buffering. Up to 64 bytes are buffered, meaning that some important errors can be avoided (collision fragments, runts, pygmies, etc.). Clearly, problems experienced beyond 64 bytes (such as giants, FCS failures, and alignment errors) cannot be detected, since cut-through switching has already started.
- Adaptive switch—another hybrid switch. Here the switch behaves by default like a cut-through switch but continuously monitors the level of bad frames experienced (and therefore propagated). When bad frame counts reach a threshold, the bridge changes behavior to operate like a classic store-and-forward switch until the error rate drops to acceptable levels.

Another problem with cut-through switching is the lack of Spanning Tree support. If frames are not buffered, it is not possible to operate the Spanning Tree protocol to resolve loops. This can be a serious design issue in a large, flat network. Of course, a combination of cut-through and classical bridges could be used as long as the designer pays attention to detail in configuring the failure modes of the topology (i.e., in normal operating conditions the slower Spanning Tree bridges would all be blocking to maintain topological integrity).

9.1.3 Traffic management

Traffic segmentation

Bridges are a simple and effective way of isolating intrasegment traffic and, therefore, a good way of reducing the traffic seen on each individual LAN segment. This generally improves network response times, as observed by

users. The extent of the improvement depends very much on the volume of intersegment traffic relative to the total traffic, as well as the volume of broadcast and multicast traffic. Bridge placement is, therefore, key to the effectiveness of traffic segmentation. If, for example, you place a bridge between a central host and a client base, where all communication is to and from the host, then you could actually make things worse, since, in effect, you have only added latency. If there is significant interclient traffic, then there will be some benefit, since the host LAN segment will be less utilized and the response time of the host could be improved.

The main use of bridges in network designs is to create partitions (discrete traffic domains) when the traffic on a repeated network begins to affect users or business-critical applications. Bridges may also be used for basic interconnection between remote offices or to transfer protocols that have no Layer 3 address and routing capabilities (such as DEC's LAT). Traditional bridges are now dying out as a useful network device—first, because of the emergence of the router network and second, because the price of switches has come down. Routers generally offer much better security and traffic management features, while switches effectively provide the same functionality but with much higher throughput.

When installed in a repeated network, the first thing a bridge will do is start to manage traffic. It uses fairly simple technology to listen to the network and work out whether to forward packets or drop them. When a bridge forwards a frame, it first makes a complete copy of it in an internal buffer and then queues the frame up on one or more output queues. Bad frames are dropped (such as frames with bad CRCs) and usually error counters are maintained.

Since frames are being buffered, we may improve network reliability by ensuring that bad frames are not propagated. Clearly, however, there is increased latency, since a frame is now copied and queued, but in practice this latency is usually very small.

Broadcast domains and scalability

It is important to understand that bridges propagate broadcasts everywhere, and this can be a serious issue that limits the scalability of bridged networks. For example, on a single LAN segment broadcasts from protocols such as ARP typically represent a small percentage of the overall traffic. As more segments are joined together via bridges, this broadcast traffic is forwarded to every segment and so begins to multiply. Since each device on a bridged network typically has to process broadcasts, this can start to degrade host performance as well as reduce available bandwidth. In order to scale Layer 2

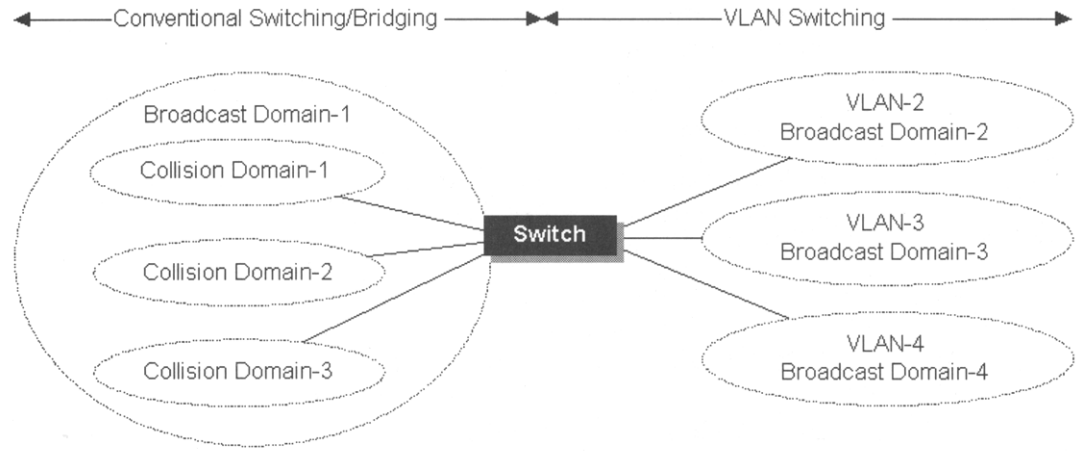

Figure 9.2 *VLAN domains.*

networks more effectively, a technique known as Virtual LANs (VLANs) was introduced to enable broadcast domains to be segregated in flat networks; this concept is illustrated in Figure 9.2. VLANs are described later in Section 9.7. Note that in conventional bridging or switching each interface represents a separate collision domain, but all physical ports belong to a single broadcast domain. VLAN ports, on the other hand, are logical interfaces with a discrete broadcast domain per interface.

Layer 2 access lists

Most bridges today implement a range of user-configurable filtering (Access Control List [ACL]) capabilities to enable the network designer to fine-tune traffic management and impose basic security. For example, you may want to stop hosts on a particular LAN from broadcasting boot requests across the entire subnet. Note that on some Layer 2 products, filtering capabilities may extend beyond the range of the Data Link Layer header. As a general rule you should try to avoid filtering above Level 2 header. This requires expert protocol knowledge, is hard to maintain, and in practice often results in subtle problems, that are hard to debug. High-level filtering is best handled using advanced switching and routing products.

Compression techniques

Traffic compression techniques are often employed on wide area bridge links. Nowadays this is primarily via standard PPP mechanisms. For historical reasons there are a variety of proprietary techniques, several of which will

work only with peer bridges from the same manufacturer. Examples include the following:

- Ethernet packet compression, such as frame header compression (typically compresses 14 bytes to 2 bytes)
- Specific protocol optimizations, such as LAT announcement compression techniques, LAT header compression (typically compresses 18 bytes to 2 bytes)
- Zero fill padding suppression (to reduce minimum packet size)

9.1.4 Load-balancing techniques

In Source Route Bridging (SRB) the ability to send data down multiple paths is built into the architecture. In transparent bridging load-balancing techniques are either dependent on lower-level data-link operations (such as multilink PPP for parallel bridge links) or proprietary implementations, since the use of multiple paths is precluded by the use of the Spanning Tree Algorithm (STA). This is a particular problem for wide area networks, since it may be beneficial for performance, reliability, and cost issues to forward traffic over several wide area links concurrently.

Proprietary techniques generally require peer bridges to be from the same vendor (with very few exceptions) and may be restricted to specific WAN media. Xyplex, for example, employs a technique called Fully Distributed Redundant Bridging (FDRB), which works transparently in cooperation with Spanning Tree bridging. Up to 32 bridge pairs are supported in parallel, and all of these links form a logical group, presented to the network as a single Spanning Tree link. Load balancing is achieved dynamically via a simple hashing algorithm based on flow attributes, using the IP addresses and port numbers to route flows over different bridges. FDRB automatically redistributes flows between bridges, as more or fewer bridge pairs are available. Vitalink also employs a technique called DLS, which enables triangulation between remote sites by allowing traffic to be forwarded down a point-to-point link between bridges. This technique is rather complicated and imposes several constraints on the design that restrict its application [1].

9.1.5 Multicast support

One of the design issues raised by multiport bridges and switches in a multicast environment is a phenomenon referred to as multicast leakage. We know that a bridge must propagate broadcasts and multicasts to all ports

other than the receiving port. In the case of IP multicasts there are several Layer 3 protocols used to distribute multicasts efficiently, so that specific multicasts are directed only to those users who require them. All of this good work is wasted if a Layer 2 switch or bridge blindly forwards these frames to its interfaces without knowledge of user subscriptions. A special class of multicast-aware bridges and switches has emerged to resolve this problem by implementing additional protocols and functionality that can transparently assist the device in acquiring multicast group status information. The three key protocols of interest are, as follows:

- Generic Attribute Registration Protocol (GARP)
- GARP Multicast Registration Protocol (GMRP)
- GARP VLAN Registration Protocol (GVRP)

GARP

The Generic Attribute Registration Protocol (GARP) is designed to allow GARP applications to propagate status information throughout the network. GARP is standardized under IEEE 802.1P. Currently there are two GARP applications: GMRP and GVRP. A GARP application makes declarations or withdrawals in order to communicate the status via a number of attributes. At present there are five attribute-specific messages and a general message supported, as follows:

- JoinEmpty (operator type 0)—a device is declaring an attribute value. At this stage the device has not registered the attribute or is interested to see if other participants wish to declare.
- JoinIn (operator type 1)—a device is declaring an attribute value and has either registered the attribute or does not care if other participants wish to declare.
- LeaveEmpty (operator type 2)—the opposite actions of JoinEmpty.
- LeaveIn (operator type 3)—the opposite actions of JoinIn.
- Empty (operator type 4)—a device is simply interested to see if other participants wish to declare.
- LeaveAll (generic)—indicates that registrations will be terminated shortly, either by specifying all attributes (by setting attribute type to 0) or by identifying the specific attributes. If participants wish to remain registered, they must explicitly rejoin.

Each GARP application is allocated a unique group multicast MAC address referred to as the GARP Application Address. GARP does, however,

use the same LLC address as Spanning Tree, relying on the combination of the unique MAC address and a specific GARP to ensure that the protocol is correctly handled on reception.

GMRP

The GARP Multicast Registration Protocol (GMRP) enables multicast group membership information to be propagated throughout a bridged or switched network. GARP is also standardized under IEEE 802.1P. In operation GMRP is similar to the combination of IGMRP (which supports dynamic group membership activities) and MOSPF (which supports multicast distribution). GMRP carries two types of information, as follows:

- Group membership information—is used to indicate whether a participant wishes to subscribe to or unsubscribe from a specific multicast group. Upon receipt of this message a multicast-aware switch or bridge can decide whether to graft or prune a port onto the distribution tree for a specific multicast group address.
- Service requirement information—specifies that the default behavior of the receiving interface should be either to forward all groups or to forward unregistered groups. This mechanism is used to enable GMRP-aware switches to interact with GMRP-unaware switches.

GMRP uses the MAC destination multicast address 01-80-c2-00-00-20. GMRP defines two attribute types: the group attribute contains group membership information, and the service requirement contains service requirement information for all groups or all unregistered groups.

GVRP

The GARP VLAN Registration Protocol (GVRP) supports the distribution of VLAN registration over a bridged or switched network. GARP is standardized under IEEE 802.1Q. GVRP uses the MAC destination multicast address 01-80-c2-00-00-21. GVRP operates in a manner similar to GARP/GMRP except that it distributes 12-bit VLAN Identifiers (VIDs) rather than 48-bit multicast group MAC addresses. VLANs are described in Section 9.7.

9.1.6 Applicability in internetwork design

Bridges and switches are tools that alleviate congestion in local area networks (Ethernet, Token Ring, and FDDI) by reducing traffic and preserving valuable bandwidth. The relative simplicity and utility of these devices,

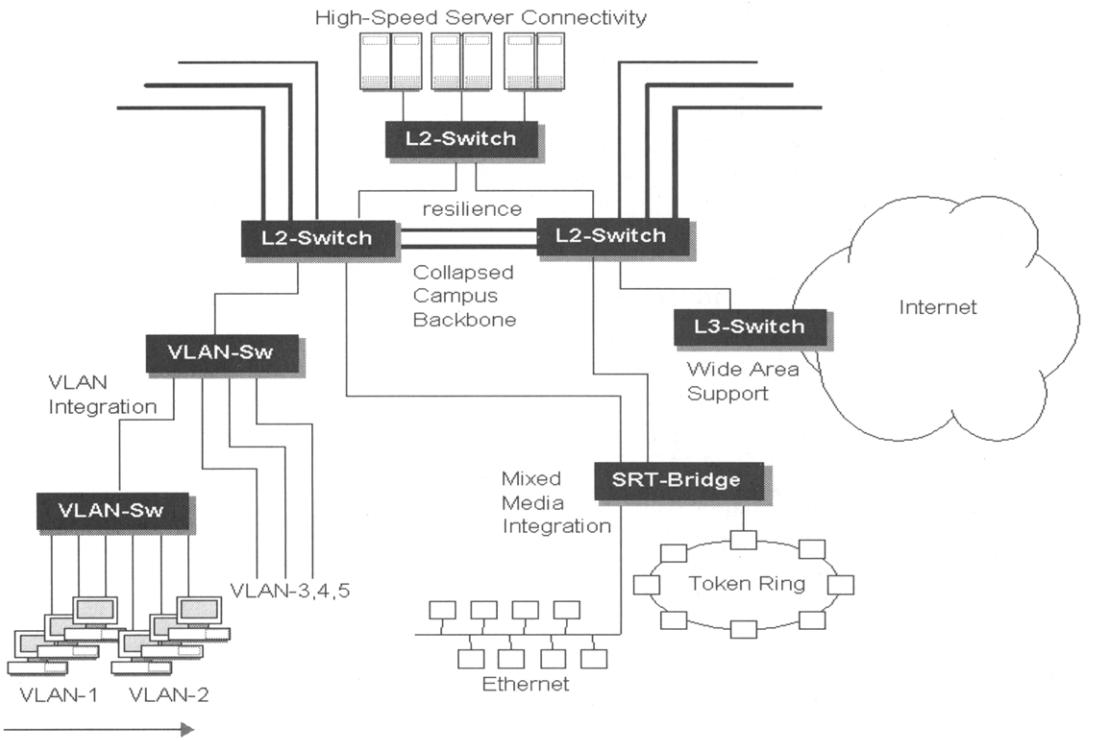

Figure 9.3 *Typical applications of bridges and switches.*

as well as their maturity, performance, and reliability, make them useful in many network designs. Protocol transparency is also a useful asset in some networks; there are still several LAN protocols in use that were not developed with routing in mind (such as NetBIOS and DEC LAT). For this reason bridging functionality is commonly employed within multiprotocol routers to handle any nonroutable protocols.

On the downside, since neither bridges nor switches are aware of Network Layer addressing, there is no concept of network hierarchy. Therefore, bridged or switched networks are not inherently scalable, and their ability to direct traffic efficiently is far from optimal. Figure 9.3 illustrates common applications of bridges and switches today. Bridging and switching applications are generally limited to the following:

- Startup or pilot networks
- Interconnecting local area networks in campus environments
- Simple, remote office interconnect
- High-speed server attachment

- Collapsed local area backbones
- Bulk high-performance user attachment

In order to impose some form of hierarchy at Layer 2, a technology known as Virtual LANs (VLANs) was introduced in the late 1980s. Essentially, VLANs introduced an addressing scheme between Layer 2 and Layer 3, which operates transparently as far as users are concerned.

Classical bridges have declined in popularity since the early 1990s. In the wide area, the flexibility, resilience, and better traffic management capabilities of multiprotocol bridge routers mean that bridges are now rarely used for LAN interconnect. In campus networks, LAN switches have effectively displaced bridges almost entirely. Switches typically offer much greater port density, flexible media support, a low price per port, and wire speed forwarding rates on a per-user basis. While specialist bridges, such as those required for protocol translation, are still in use in many IBM environments, switches have gone from strength to strength, incorporating additional functions such as Layer 2, Layer 3, Layer 4, LAN, WAN, Virtual LAN (VLAN), and even security functions.

9.2 Transparent bridging

Transparent bridges were first developed at Digital Equipment Corporation (DEC) in the early 1980s. This work was subsequently adopted by the IEEE and ISO, culminating in the IEEE 802.1 and ISO 8802-1 standards. An IEEE 802.1d transparent bridge is characterized by store-and-forward operation, the ability to listen to all traffic on all ports, the ability to learn and cache topology information transparently, and the ability to resolve topology loops through the implementation of the Spanning Tree algorithm. Transparent bridges are commonly used in Ethernet/IEEE 802.3 networks to connect LAN segments. Transparent bridging may also be used to connect Token Ring LAN segments, although this application is not common.

9.2.1 Operation

Transparent bridging relies on the principle that a node should be able to transmit a frame to another node on a LAN without any knowledge of the location of that node. Transparent bridges are completely transparent to network hosts, and this makes their use in network designs relatively simple. When transparent bridges are powered on, they begin to build up a picture of the network topology by listening to packets on the network and learn-

ing which source addresses are associated with which physical interfaces. To provide transparent operation, a bridge will assume that the intended destination of a packet could be on any interface until it has proof. This means that any frame that has a destination address not recognized by the bridge will be forwarded to all interfaces other than the receiving interface. In essence a transparent bridge acts like a buffered repeater until it can start to make smart decisions, and only the source address of a packet is proof of location.

Transparent bridges are constantly listening and learning. Each frame received at an interface will cause the bridge to either make a new association of a source MAC address with an interface (meaning a new entry in a filter table) or cause an aging timer to be refreshed for an existing table entry. This filtering database acts as a forwarding table for any received frames. Whenever a new packet is received, these tables are examined for any entries matching the destination address. If the destination address is matched against a particular interface, then the packet will be forwarded to that interface only; otherwise, it will be sent to all interfaces. Packets are never sent out on the interfaces on which they are received. To summarize, a packet received at an interface will go through the following process:

- If the destination address is broadcast or multicast, then the packet will always be forwarded to all interfaces except the receiving interface.
- If the destination address is not in the database, the frame is forwarded to each interface except the receiving interface.
- If the destination address is in the database and the frame was received on an interface associated with the destination address, the frame is discarded.
- If the destination address is in the database and the frame was received on an interface not associated with the destination address, the frame is forwarded to the associated interface for this destination address in the database.

Figure 9.4 illustrates how a transparent bridge will build up its filtering database. Let's assume that the network has been inactive for a long time and there are no entries in the bridge filter tables. In the morning the bridge will start to build a new picture of the topology, as follows:

- At 8:30 A.M. Node-A sends a packet destined for Node-B. When the bridge receives this packet on interface 1, it learns from the source address that Node-A is reachable via interface 1. It then adds Node-A to its filter tables, associating it with interface 1 and setting a timer

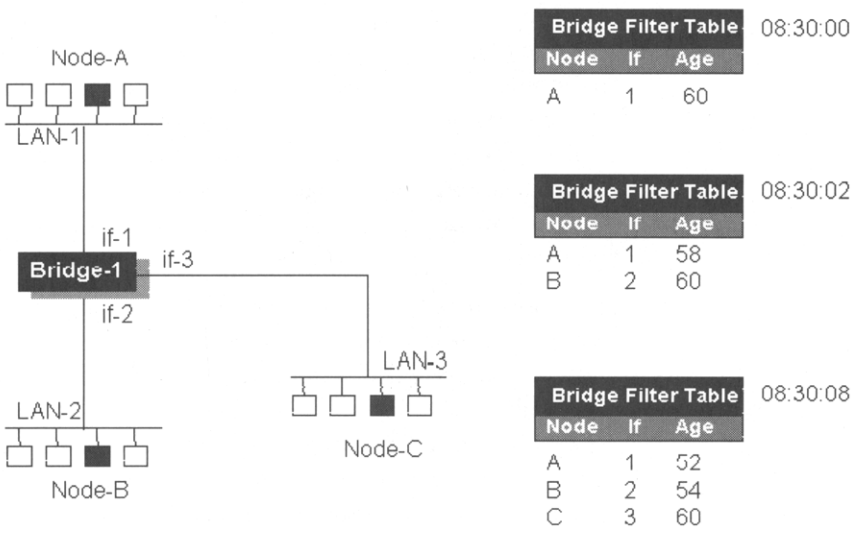

Figure 9.4 *Transparent bridging operation.*

(let's assume a 60-second timer for this example). The packet is then forwarded to both interface 2 and interface 3, since the bridge has no idea where Node-B is at this point.

- Two seconds later, at 8:30:02 A.M., Node-B sends a reply to Node-A. When the bridge receives this packet, it learns from the source address that Node-B is reachable via interface 2. It then adds Node-B to its filter tables, associating it with interface 2 and setting a timer. The bridge examines the destination address in the packet and scans its filter tables, finding a match on interface 1. The bridge then forwards this packet only to interface 1. Note that by now the Node-A filter table entry is aging.
- Six seconds later, Node-C sends a packet to Node-D. When the bridge receives this packet, it learns from the source address that Node-C is reachable via interface 3. It then adds Node-C to its filter tables, associating it with interface 3 and setting a timer. The bridge examines the destination address in the packet and scans its filter tables, finding no match. The bridge then forwards this packet only to interfaces 1 and 2. If Node-C continues to send packets to this address, they will continue to be forwarded out of both interfaces, and Node-C's aging timer will simply be reset.

The principle that the bridge's memory of the topology is constantly being eroded is important. First, it is possible that the topology will change, and therefore a bridge must be capable of responding to such changes. Sec-

ond, filter tables consume memory and there is typically a finite limit on the number of learning table entries (usually a power of 2, such as 1,024, 2,048, or 4,096). On large, flat networks the number of hosts may exceed the number of available table space, so it is useful to age out any hosts that have not communicated for some time. Clearly, if you have a very large network and limited bridge memory, then you could have a real problem, since any destination addresses that are not in the tables cause packets to be flooded out of multiple interfaces. For the same reason it is not a good idea to set this aging timer too low. To avoid unnecessary surges of traffic, especially on large or busy networks, transparent bridges implement a simple state machine that starts off by blocking all traffic and precharges its filter tables.

The bridge state machine

IEEE 802.1d transparent bridges implement the Spanning Tree Protocol (STP), which relies on a simple state machine to determine how to communicate with peers, learn addresses, prevent loops, and recover from topology failures. The basic operational states are illustrated in Figure 9.5. Note that state 4, in particular, differentiates Spanning Tree bridges from simple transparent bridges (now quite rare). Note that these states also refer to ports

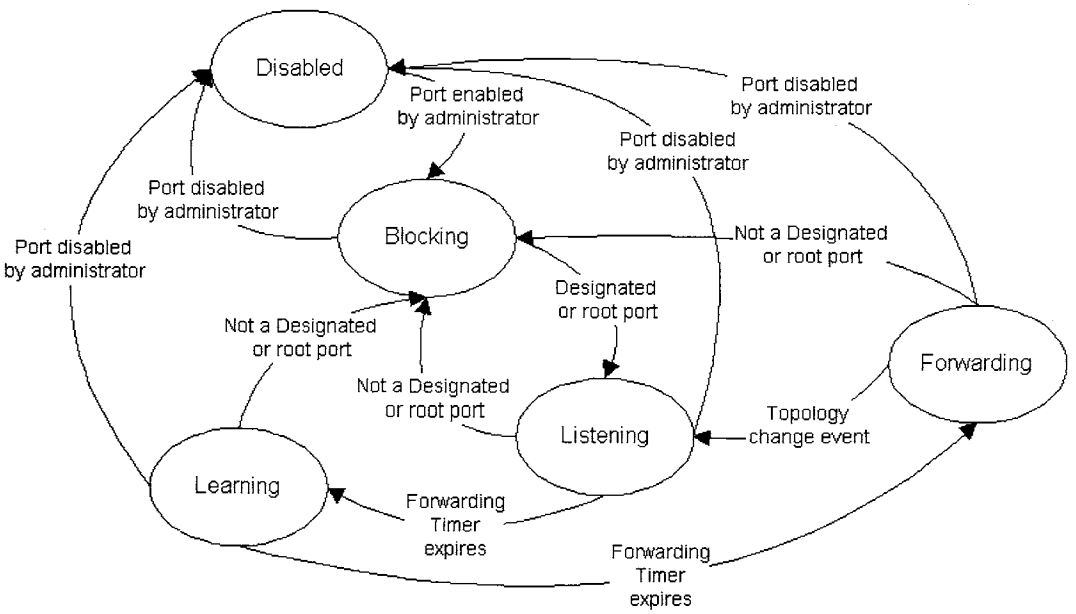

Figure 9.5 *Spanning Tree state transition diagram for a bridge port.*

(interfaces). On a multiport bridge or switch the device may be in several states concurrently on different ports.

In states 1, 2, and 4 the network is effectively broken for all users who need to reach destinations across a bridge (local communications on a segment, for example, are still active). A timer called the forward delay timer (a form of hold-down timer) is implemented to ensure that user data frames do not circulate while a bridge is reconfiguring. This timer is normally configurable by a network administrator (typically the default is 15 seconds, maximum 255 seconds). The main features of the various states are highlighted as follows:

- Listening—A port enters this state if it has been classified as either a designated or root port. In this state STP is still configuring, and a forward delay timer is initialized. Ports may receive and act upon bridge management frames (i.e., Hello-BPDUs) but will not pass or process user data frames. The listening state allows each bridge to take part in an election process, where a master, or root, bridge will be elected. Once elected the root bridge begins sending out Hello messages at regular intervals, and all other bridges will align themselves with the new root bridge; some will forward on all ports while others may decide to block certain ports to stop loops forming. Once the forward delay timer has expired, the bridge will enter a learning state. Any ports deemed to be not designated or root at this point would go into the blocking state.
- Learning—In the learning state a forward delay timer is again initialized. The bridge will then receive and process all user data frames but will still not forward them. This enables the bridge to precharge its filter database by caching any source address to port mappings it learns (i.e., each new source MAC address observed by the bridge is placed in a table together with the ID of the receiving port). Upon expiration of the forward delay timer, the bridge will enter either the forwarding or the blocking state.
- Forwarding—The bridge becomes fully activated. User traffic is forwarded using the filter tables to determine which ports to send packets to. Even when the bridge starts to forward frames, it is always updating this filter database with new or updated information on station locations. A bridge makes essentially three forwarding decisions, as follows:
  - Local destination—If the bridge knows that the destination station is on the same segment as the source address in the frame, it

will discard the frame (there is no need to transmit it, since both stations can see each other's traffic).

- Remote destination—If the bridge knows that the destination station is on another segment, it forwards the frame only to the interface attached to that segment.
- Unknown destination—If the bridge has no knowledge of where the destination station resides (e.g., if that station has not transmitted any data so far), then the bridge forwards the frame on all segments except the interface attached to the source segment and any ports in the blocking state (a technique known as flooding).

- Blocking—In this phase only bridge management frames are received; no user traffic is forwarded, and no bridge management frames are forwarded, since the port is neither a designated nor a root port. Once ports are determined to be either root or designated ports the port enters the listening state.
- Disabled—The port is disabled through software by the system administrator.

The precharging stage (learning) is important in busy networks. During the forwarding state the bridge knows nothing of the whereabouts of end-station devices. If the bridge were to go straight into a forwarding condition, then it would have no choice but to send all packets to every other port except the received port (rather like a multicast or broadcast packet). On a busy network this would result in an unhealthy surge of traffic until sufficient knowledge is acquired to discard packets intelligently.

Loop avoidance

IEEE 802.1d–compliant transparent bridges require that there be only a single active path between any two LANs in an internetwork. This ensures that frames do not loop around the topology and multiply. Transparent bridges rely on the Spanning Tree Protocol (STP) to resolve potential loops and provide automatic topology recovery from network or bridge failures. Without STP, the transparent bridge algorithm fails when there are multiple paths between bridges connected subnetworks. Consider the bridging loops created in Figure 9.6.

In Figure 9.6(a), we have a bridging loop created by two non-STA bridges. Let's suppose that Node-A transmits a frame destined for Node-B. Both Bridge-1 and Bridge-2 receive the frame from Network-1 and correctly conclude that Node-A is on Network-1. Both bridges cache Node-A's source address against if-1. Node-B subsequently receives two copies of

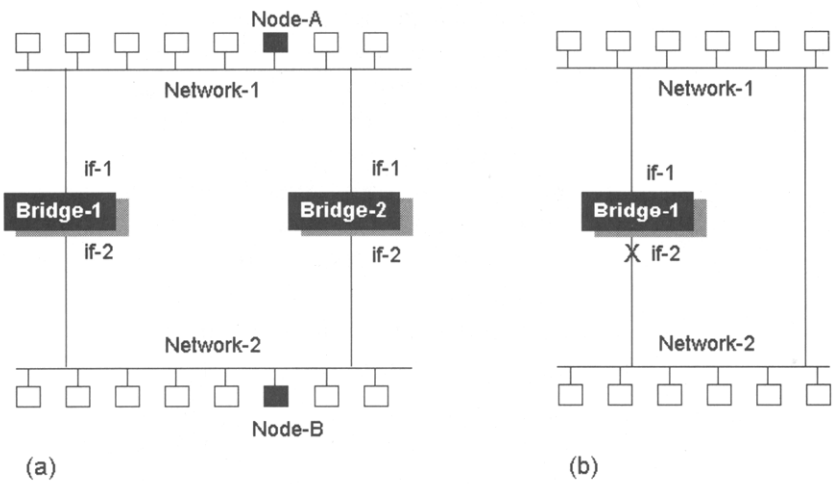

Figure 9.6
(a) Bridging loop created by two non-STA bridges. (b) Loop created with a simple Spanning Tree topology.

Node-A's frame, so we have doubled the traffic on Network-2 and possibly confused Node-B. Worse still, both bridges now see the frame propagated from their peer on their Network-2. At this point we should expect both bridges to immediately change their internal tables so that Node-A is now cached on if-2, and the entry associated with if-1 should be flushed (you might want to check this with your bridges, since I have come across some that do not!). If this is the case, when Node-B replies to Node-A's frame, both bridges receive and subsequently drop the reply, since their mapping tables now indicate that the destination address (Node-A) is local (i.e., on the same interface as the source address). So we will not see a reply at Node-A, since we have lost communication (even though we have plenty of it physically). Note that the situation could be worse; if the bridges do not flush old address-interface associations, then traffic could simply loop and multiply exponentially, causing a network meltdown.

In addition to basic connectivity issues, we can see some even more crippling problems with broadcasts and multicasts. Assume that Node-A's initial frame is an ARP broadcast. After the initial replication of the frame, both bridges will forward the frames continuously, using all available network bandwidth and inhibiting the transmission of other packets on both segments. We call this a broadcast storm, the networking equivalent of acoustic feedback.

In Figure 9.6(b) we see a single Spanning Tree bridge connecting two LANs. The fact that there is a parallel loop means that a loop is created. In this example the bridge will automatically elect itself as root, detect the

loop, and close down one of its ports (by placing it into a blocking state). The process by which this is achieved is explained in the next section.

Spanning Tree Algorithm (STA)

The Spanning Tree Protocol (STP) uses a Spanning Tree Algorithm (STA) to eliminate logical loops in a bridged network topology, while providing automatic rerouting around failures to preserve the network integrity. You may remember that a Spanning Tree is a loop-free subset of a graph (i.e., the network topology), which includes all nodes. Graph theory states: For any connected graph consisting of nodes and edges connecting pairs of nodes, there is a Spanning Tree of edges that maintains the connectivity of the graph but contains no loops.

The STP algorithm works by automatically configuring a loop-free subset of the network topology, a single Spanning Tree. It achieves this by placing all bridge interfaces, which, if forwarding, would create loops, into a standby, or blocking, state. Blocked ports can be switched to forward in the event of link failure to provide an alternate path through the internetwork. STP is relatively simple in that, unlike more sophisticated routing protocols such as RIP and OSPF, it does not distribute sufficient information for bridges to know what the actual topology is. In effect, it works by aligning bridges based on relatively simple messages passed from an elected master (called the root bridge) and implements fairly long hold-down timers to ensure that all bridges have sufficient time to realign themselves after a failure.

Spanning Tree bridges communicate with their neighbors using configuration messages called Bridge Protocol Data Units (BPDUs). Bridges are constantly listening on all ports to each other's transmissions. All bridge topology decisions are made locally; there is no central authority on network topology or administration. The Spanning Tree calculation occurs when the bridge is powered up and whenever a topology change is detected. The calculation requires communication between the Spanning Tree bridges, which is accomplished through configuration messages (BPDUs). Configuration messages contain information identifying the bridge that is presumed to be the root (root identifier) and the distance from the sending bridge to the root bridge (root path cost). Configuration messages also contain the bridge and port identifier of the sending bridge and the age of information contained in the configuration message.

The Spanning Tree algorithm calls for each bridge to be assigned a unique identifier. Typically, this identifier is one of the bridge's MAC

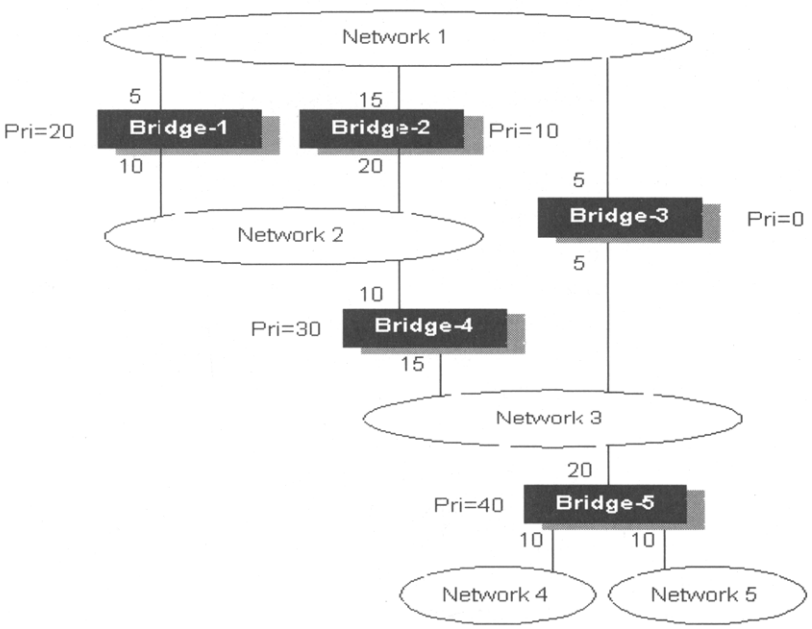

Figure 9.7 *Starting topology—only STA BPDUs are circulating at this time.*

addresses plus a priority. Each port in every bridge is also assigned a unique (within that bridge) identifier (typically, its own MAC address). Finally, each bridge port is associated with a path cost. The path cost is the cost of transmitting a frame onto a LAN through that interface. In Figure 9.7, path costs are noted on the lines emanating from each bridge. Path costs are usually defaulted but can be assigned manually by network administrators.

The first activity in the Spanning Tree algorithm is the election of the root bridge, which is the bridge with the lowest value bridge identifier. The root bridge transmits periodic Hello-BPDUs.

Next, the root port on all other bridges is determined. A root port is the port through which the root bridge can be reached (i.e., the upstream port) with the least aggregate path cost. This value (the least aggregate path cost to the root) is called the root path cost. All bridges append their path costs to the BPDUs received on their root ports and send them out of their downstream ports. At this stage the network still has loops, but there is still no regular traffic being forwarded other than BPDUs.

Finally, designated bridges and their designated ports are determined. A designated bridge is the bridge on each LAN that provides the minimum path cost back to the root bridge. A LAN's designated bridge is the only bridge allowed to forward frames to and from the LAN for which it is the

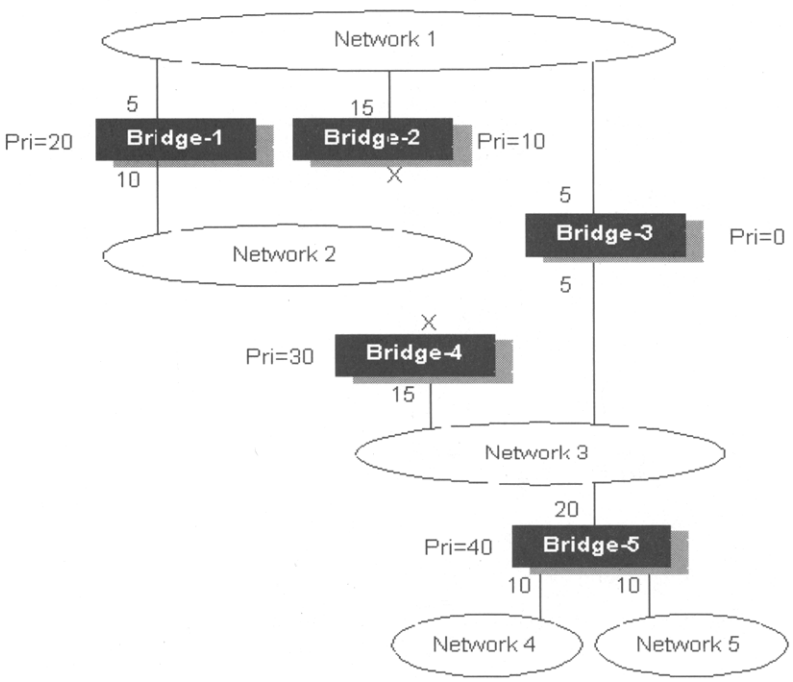

Figure 9.8 *Loop-free bridged network after Bridge-3 is elected root.*

designated bridge. A designated port is the root port on a designated bridge. All highest-cost downstream ports are blocked, leaving only the least-cost paths available.

The result is a single Spanning Tree with lowest-path costs aligned toward the current root bridge. It is important to note that if a different root bridge is subsequently elected, then, depending where that bridge is, the entire topology could change considerably. In the example network in Figure 9.7, Bridge-3 has the lowest priority and so is elected root (all bridges attached to the same LANs defer when they hear BPDUs of lower priority). The resulting topology is shown in Figure 9.8.

In some cases, two or more bridges can have the same root path cost. For example, in Figure 9.7, Bridges-2 and -4 both have a root path cost of 15. If Bridge-1 were to fail, then these two bridges would tie, and the lowest priority bridge would win (Bridge-2).

Bridges exchange configuration messages at regular intervals (typically two seconds). If a bridge fails, then neighboring bridges will quickly detect the absence of configuration messages and initiate a Spanning Tree recalculation. In Figure 9.9 we see a different way of representing the topology.

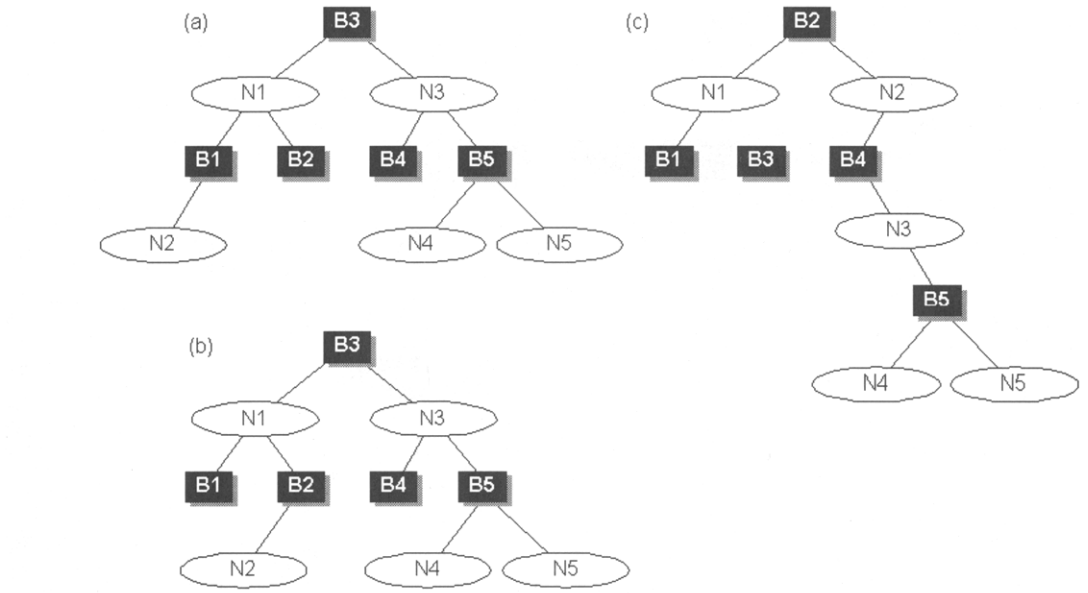

Figure 9.9 *Changes in Spanning Tree topology. (a) Stable topology with Bridge-3 as root. (b) Stable topology after Bridge-1 loses its interface to Network 1. (c) Stable topology after Bridge-3 dies, forcing the election of Bridge-2 as root bridge.*

Figure 9.9(a) shows the same stable topology shown in Figure 9.8. Figure 9.9(b) shows the result of Bridge-1 losing its interface with Network 2. Figure 9.9(c) illustrates a completely new topology, resulting from Bridge-3 dying. In this case Bridge-2 takes over (next lowest priority) and the whole network is realigned.

For this reason it is important to consider the order in which roots are to be prioritized to minimize disruption throughout the network. If bridges with priorities similar to the root are located closest to the root, then peripheral bridges do not necessarily have to reorient themselves if the root is subsequently reelected. For example, in Figure 9.9(c), Bridge-4 has to swap designated and root ports completely after Bridge-2 becomes root.

BPDU frame format

Field definitions

- Protocol identifier—Set to zero.
- Version—Set to zero.
- Message type—Set to zero.

- Flag—A one-byte field, of which only the first two bits are used. The topology change (TC) bit signals a topology change. The Topology Change Acknowledgment (TCA) bit is set to acknowledge receipt of a configuration message with the TC bit set.
- Root ID—Identifies the root bridge by listing its two-byte priority followed by its six-byte ID.
- Root path cost—Contains the cost of the path from the bridge sending the configuration message to the root bridge.
- Bridge ID—Identifies the priority and ID of the bridge sending the message.
- Port ID—Identifies the port from which the configuration message was sent. This field allows loops created by multiple attached bridges to be detected and dealt with.
- Message age—Specifies the amount of time since the root sent the configuration message on which the current configuration message is based.
- Maximum age—Indicates when the current configuration message should be deleted.
- Hello time—Provides the time period between root bridge configuration messages.
- Forward delay—Provides the length of time that bridges should wait before transitioning to a new state after a topology change. If a bridge transitions too soon, not all network links may be ready to change their state, and loops can result.

Transparent bridges exchange configuration messages and topology change messages. Configuration messages are sent between bridges to establish a network topology. Topology change messages are sent after a topology change has been detected to indicate that the STA should be rerun. The IEEE 802.1d configuration message format is shown in Figure 9.10.

Topological change messages consist of only four bytes. They include a protocol identifier field, which contains the value 0; a version field, which contains the value 0; and a message type field, which contains the value 128. An example Hello-BPDU is shown in Figure 9.11. For further information on the Spanning Tree algorithm and protocol the interested reader is referred to reference [1].

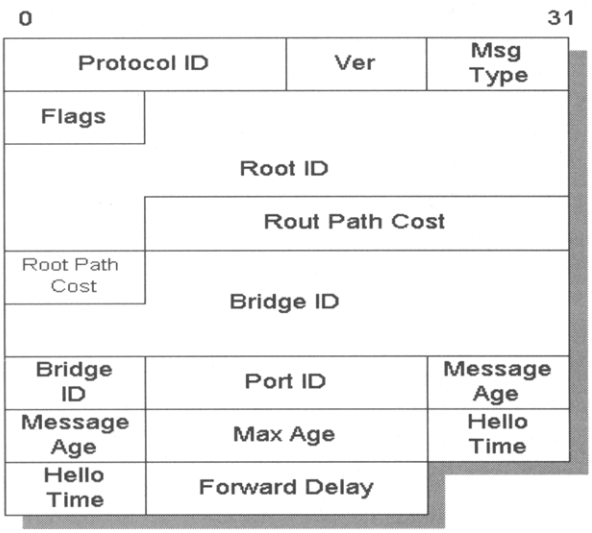

Figure 9.10 *Transparent bridge configuration message format.*

Timers

The key timers used to control Spanning Tree protocol operations are Forwarding_Delay, Message_Age, Root_Hello_Time, and Max_Age. A Spanning Tree BPDU is considered valid only if:

Message_Age + Root_Hello_Time < Max_Age

So with a Root_Hello_Time of 2 seconds and a Max_Age of 20 seconds, the Message_Age must be less than 18 seconds for the frame to remain

```
File:PPLOT1.SYC      Type:SNIFFER_SYC  Mode:•••••••  Records:480
================================================================================
Frame     : 1                Len      : 39               Error    : None
T Elapsed: 12:58:03:331      T Delta  : 00:00:00:003
-------------------------------[WAN]-------------------------------------------
Dst Addr : DTE               Src Addr : DCE
-------------------------------[ppp encap]-------------------------------------
Address  : FF                Control  : 03
-------------------------------[ppp]-------------------------------------------
Type     : PID               Protocol : 802.1D Hello
-------------------------------[ieee stp]---------------------------------------
STA Pri  : 0                 STA Ver  : 0                BPDU Type: CONFIG_BPDU
Flags    : 0                 Root Pri : 256              Root ID  : This Bridge
Root Cost: 00000000          Brdge Pri: 256              Bridge ID: 08008701ff84
Port ID  : 0x8002            Msg Age  : 0     secs       Max Age  : 20    secs
T_Hello  : 2     secs        Forw Del : 15    secs
================================[data:   0]=====================================
```

Figure 9.11 *Typical STA Hello-BPDU.*

valid. Note that even though bridges throughout the network may be configured with different local Hello times (this is not recommended), the Hello interval used by all bridges in a Spanning Tree is determined by the active root bridge. All bridges synchronize with the Root_Hello_Time passed in its Hello-BPDUs.

The minimum forwarding delay should be consistent with the formula:

$2 \times (\text{Forwarding\_Delay} - 1 \text{ sec}) = \text{Max\_Age}$

So a Forwarding_Delay of 15 seconds would result in a Max_Age of 28 seconds.

The DEC Spanning Tree algorithm

The Spanning Tree protocol was originally developed by Digital Equipment Corporation, one of the founding Ethernet vendors. DEC's algorithm was subsequently revised by the IEEE 802 committee and published in the IEEE 802.1d specification. It is important to note that the DEC algorithm and the subsequent IEEE 802.1d algorithm are not the same, nor are they compatible. If both algorithms are implemented in the same network, then you will need to be extra careful to avoid some potentially nasty looping problems. Key points to understand are as follows:

- Typically the two algorithms use different destination multicast addresses by default, and, if both are activated, they will be invisible to each other (causing potential loops on the network, which may be difficult to debug).
- It is possible to run both algorithms together if you really must, but you should only do this by isolating the parts of the topology to be controlled by each algorithm with a well-defined interface between the two domains. Specifically, do not allow Spanning Tree domains to overlap.
- You should also check the filter configuration on non-DEC bridges, since there may be default filters configured to block DEC multicast addresses, and this can cause problems when troubleshooting loops in multivendor bridged environments.

9.2.2 Design guidelines

The following guidelines are provided:

- Maximum bridges in series—Bridges do incur some delay, and therefore there are guidelines to the number you should install in series in

a network design. For transparent bridges such as Ethernet bridges the recommended maximum number in series is seven.

- Bridge placement—I have seen several real examples of bridges being positioned where they either do more harm than good or no good at all.
  - In general you should aim to get at least a rough idea of where the major sources and sinks of traffic are, and use bridges to isolate users and devices into sensible partitions.
  - Placing a bridge in between a server and a user group generally does nothing but slow things down. This could be a valid thing to do if there is significant client-to-client traffic or significant or sensitive server-to-server traffic and you want to protect each segment and thereby lower the overall traffic. Another possible justification is the case where the device is a multiport switch (each port representing a user or server), where internal filtering processes enable traffic to be forwarded quickly and efficiently.
- Bridge timers—In general you should avoid changing the default timers unless absolutely necessary, or you are prepared to document and manage your network proactively. On a network where topology convergence is paramount, you may want to speed up Spanning Tree reconfigurations by shortening the Hello timer and decreasing the listening and learning periods. One important trade-off here is that on busy networks decreasing the learning period significantly can result in a large traffic surge once the forwarding bridges kick in.
- Control root bridge election—On any reasonably sized bridge network (i.e., more than just a couple of bridges), you should proactively configure which bridges will become root. Try to aim for a bridge in the center of the topology (i.e., a bridge with roughly the same number of hops to all other bridges).
- Control backup root bridges—Often overlooked, but nevertheless important is the election of a new root bridge when the primary bridge fails. In this case try to ensure that you nominate bridges closest to the primary root bridge in the topology; this way if the main root bridge dies many of the remaining bridges will maintain the same orientation with the new root. This will speed up convergence after a root bridge failure by minimizing the impact of the new election process on all other bridges.
- STA bridges should not be underpowered, since they are doing considerably more than just forwarding traffic. There have been examples

of early STA bridges with insufficient processing power to handle heavy traffic conditions and STA concurrently; the results are potentially chaotic, leading to network meltdown (imagine if a bridge starts to drop BPDUs or cannot forward them in time under load; we would see oscillations in the Spanning Tree, loops, and potentially broadcast storms).

- Potential address conflicts must be dealt with prior to joining LANs via bridges. When two LANs are interconnected via bridges, they effectively become one single address space. If you have poorly administered IP addressing schemes on one or both of these LANs, you could waste considerable time tracking down duplicate address issues. Either resolve possible problems beforehand, or ensure that addressing is handled centrally (from a common DHCP server, for example).
- Interoperability—In the LAN environment most bridges should work with each other quite well. There may be some issues with older bridges running prestandard Spanning Tree, or bridges running proprietary Spanning Tree, but most of the time you should not see any issues. In the WAN environment different vendors' bridges often will not interoperate, or interwork, with reduced functionality. Typically the only choice for interoperability is via the PPP protocol over a serial link. In general it is advisable to run bridges in pairs, with the same vendor at each end of the wide area link.

9.3 Source Route Bridging (SRB)

Source Route Bridging (SRB) is covered by IEEE 802.5d and implemented by IBM and several of the major bridge vendors for use over Token Ring LANs. SRB is a Layer 2 function (the term routing is somewhat misleading) and operates between two Token Ring LANs, connected via either a local or a wide area link. Both 4 Mbps and 16 Mbps ring speeds are supported. SRB is far more popular in Token Ring environments than either translation bridging or Source Route Transparent (SRT) bridging (described later).

Source routing is fundamentally different from transparent bridging in that the onus is placed on hosts (rather than the bridges) to provide and maintain routing information when transmitting frames. This enables source routing to offer more flexibility and several distinct advantages over transparent bridging. For example, source routing enables multiple active routes to be used to forward traffic rather than the single Spanning Tree, increasing both the throughput and overall efficiency of the network (each

host uses a specific path between source and destination, but different source-destination pairs may use different paths concurrently). Source routing also allows a host to determine the MTU for any given path.

9.3.1 Operation

In source route bridging the node transmitting a frame specifies the path that should be taken over an internetwork and includes the path in every frame sent in a special field called the Routing Information Field (RIF), illustrated in Figure 9.12. This is in contrast to transparent bridging, where the bridges make the routing decision. In order to forward packets efficiently, the transmitting node must first establish the best path to a specific destination, using a discovery process. To facilitate this, networks and bridges are configured with a 12-bit and 4-bit address, respectively. Bridge IDs can be reused within the same extended LAN as long as the bridges with the same ID are not running in parallel.

The Routing Information Field (RIF)

The RIF (see Figure 9.12) is an extension of the source MAC address field. The RIF starts with a 16-bit Route Control Field (RCF) header, followed by zero or more 16-bit Route Descriptor Fields (RDFs—sometimes called a route designator). The RCF contains key information on the type of frame being sent and how the information should be handled. The RDF contains pairs of ring and bridge numbers (12 and 4 bits, respectively). The size of the RIF, and hence the number of RDF ring-bridge pairs, is theoretically constrained by the length field within the RCF header. The length field is 5 bits wide, allowing up to 32 bytes in total. Subtracting the 2 bytes of the RCF itself leaves 30 bytes, equivalent to 15 RDF fields of 2 bytes each. In

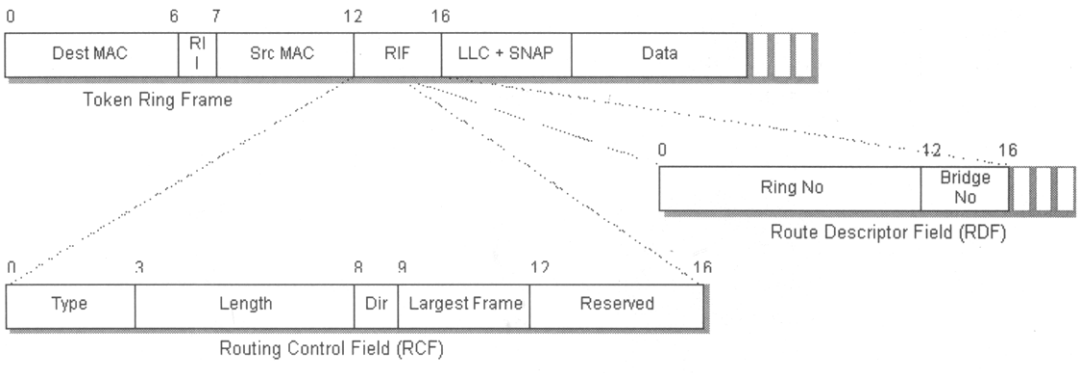

Figure 9.12 *Structure of the RIF field.*

practice the number of RDFs is constrained to meet the seven-bridge constraint on network diameter (primarily a limitation imposed by IBM, which effectively controls the SR bridging standards).

Field definitions

- Routing control field (16 bits)
  - Type (3 bits)—The routing type field identifies the frame as either an explorer frame or an explicitly routed frame. Values (in binary) are as follows:

    001 Specifically Routed Frame (SRF)

    100 All Routes Explorer (ARE)

    110 Spanning Tree Explorer, All Routes Explorer return

    111 Spanning Tree Explorer, Spanning Tree return
  - Length (5 bits)—Total length of the RIF field in bytes. The minimum value is 2 (just the RCF present), and the maximum value is 30. In practice some implementations limit the maximum value to support only seven RDFs, which determine the radius on a Token Ring network.
  - Direction (1 bit)—Used to indicate in which direction to read the RDF entries. Cleared equals left to right and vice versa.
  - Largest frame (3 bits)—Indicates the MTU size supported over a particular path, in binary this can be:

    000 Up to 516 bytes, IEEE 802.2 LLC and ISO 8473 CLNP

    001 Up to 1,500 bytes, IEEE 802.3 Ethernet

    010 Up to 2,052 bytes, an 80 × 24 character screen plus control information

    011 Up to 4,472 bytes, IEEE 802.5 Token Ring 4 Mbps and FDDI

    100 Up to 8,144 bytes, IEEE 802.4 Token Bus

    101 Up to 11,407 bytes, IEEE 802.5 Token Ring 4 Mbps burst errors unprotected

    110 Up to 17,800 bytes, IEEE 802.5 Token Ring 16 Mbps

    111 Up to 65,535 bytes, Interpreted as ANY size and used in an all routes broadcast frame.
  - Reserved (4 bits)—Unused

- Route descriptor (or route designator) field—A 16-bit field containing ring/bridge ID pairs
  - Ring number (12 bits)—The ring ID. This is configurable on all SRBs.
  - Bridge number (4 bits)—The bridge ID

  If a frame is received by a bridge with the RDF field full, it must be discarded.

Since there was no place for the Route Information Indicator (RII) bit in the original IBM specifications, the bit selected is the group/individual bit of the source MAC address field. Clearly, if this were set in the destination address, it would indicate a multicast frame, but in the source address field there is no such convention, so it was commandeered for this use by IBM. If the RII bit is set, an SRB will forward the frame; if cleared, then an SRB will assume that the two stations communicating are local to the ring.

Host-to-host communications

When a Token Ring station wishes to transmit data to another station for which it does not have routing information, it will first issue a probe, in the form of an Exchange Identification (XID) frame, onto the local ring. For example, in Figure 9.13 Node-A wishes to transmit data to Node-B, so Node-A sends out a local XID probe onto network 100. In this case the XID frame circulates the ring and is returned back to Node-A without any positive indication that Node-B has seen it (i.e., the AR and AC bits are not set). Node-A must therefore assume that Node-B resides on a remote network, and a route discovery process is initiated.

The next step varies depending upon the implementation, since the standards were never entirely clear on the various applications of Spanning Tree explorer frames. In essence an explorer frame must be sent, addressed to the destination node, to discover a path to that destination. As the explorer frame crosses bridges and networks, it grows in size by aggregating ring numbers and bridge IDs in the Routing Information Field (RIF). The explorer frame can also collect other useful information on a path, such as the minimum MTU required. The difference in implementations lies in the type of explorer issued and whether or not Spanning Tree is enabled on intermediate bridges, as described in the following scenarios:

- Scenario 1—In this scenario the generally accepted doctrine is that the source node sends out an All Routes Explorer (ARE) with a length set to two bytes (since there are no router descriptor fields

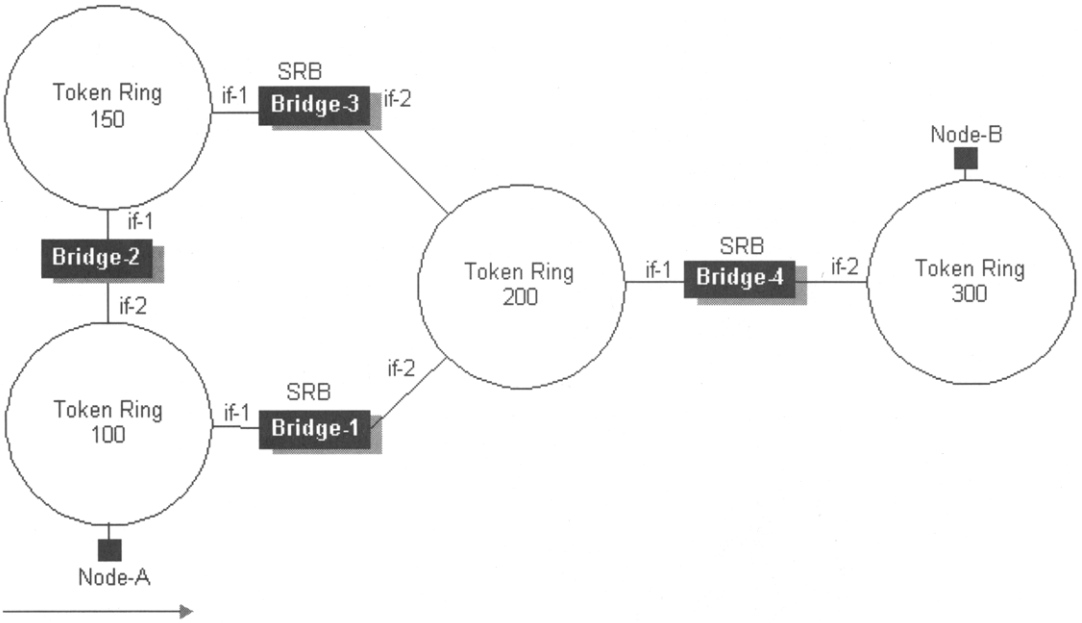

Figure 9.13 *Source route bridging.*

within the RIF field at this stage). In this case Node-B should receive multiple copies of the discovery frame, since there are multiple paths from Node-A to Node-B. Once ARE frames reach the destination, Node-B caches any routes (from the RIF fields), and then returns each frame as a specifically routed frame with the direction bit in the RCF flipped to 1 to indicate that the route should be interpreted backwards. In this way a frame is returned down each available path back to the source.

- Scenario 2—In this scenario (typically the IBM implementation) the transmitting node issues a single-route broadcast (i.e., RCF type 6—Spanning Tree Explorer, All Routes Explorer return). In this case Node-B should, therefore, receive only one copy of the discovery frame. When explorer frames arrive at Node-B, it must respond with a discovery response frame marked all-routes broadcast. This will contain the most significant bit; the Route Information Indicator (RII), set in the source MAC address field; and a single RDF entry with the bridge number set to 0, and the ring number set to 300. The direction bit (Dir) is normally flipped in the RCF for the return path. Since the response frame is marked all-routes broadcast, it will pass through all bridges and rings on its way to Node-A. Each bridge must

insert its bridge and associated ring number, so in effect the route is built up in reverse (contrary to scenario 1).

In both scenarios Node-A will receive several response frames, and it is Node-A's responsibility to cache the most efficient route based on predetermined criteria. In our example Node-A will receive frames indicating two paths to Node-B, as follows:

- Route 1—Ring 100 to Bridge-1 to Ring 200 to Bridge-4 to Ring 300
- Route 2—Ring 100 to Bridge-2 to Ring 150 to Bridge-3 to Ring 200 to Bridge-4 to Ring 300

The IEEE 802.5 specification does not mandate the criteria Node-A should use in choosing a route, but it does include several suggestions, as follows:

- First frame received
- The response with the minimum number of hops
- The response with the largest allowed frame size
- Combinations of the above criteria

In most cases, the path contained in the first frame received will be used, and in this case we are most likely to choose Route 1, all bridge and link performances being equal. Once a route is cached at the hosts, the same path will be used for all subsequent traffic between them. In practice stations do not cache routing information for long periods to keep routing table sizes small and ensure that topology information is reasonably dynamic. If both hosts were on the same ring, then the Route Information Indicator (RII) bit is cleared to stop traffic leaking through bridges unnecessarily.

Spanning Tree operations

We described the Spanning Tree Algorithm (STA) in Section 9.2. Spanning Tree may be disabled by default in some SRB implementations. When enabled, however, there is a subtle difference between the way that the STA works in an SRB environment. Unlike transparent bridging, in SRB networks STA ports in the blocking state only block non–source-routed traffic and Spanning Tree Explorer frames. This enables source-routed traffic to employ the RIF field to provide alternate paths and maximize bandwidth efficiency, as described previously. This also means that broadcast and multicast traffic can be forwarded down a single multicast tree, rather than duplicating such traffic through all paths.

Extended source route bridging

BAY Networks, Inc. (now part of Northern Telecom) has implemented an extension to basic source route bridging in its router product line. Extended source routing enables end systems to support both local source route bridging and internetwork routing and eliminates the seven-hop bridge restrictions imposed by implementations such as that offered by IBM. Routers use source route Explorer packets to build the routing tables dynamically, and each router can reset the hop count to zero as it forwards packets onto the next router. This protocol improves reliability, improves scalability, and decreases network response time.

9.3.2 Design guidelines

In a Token Ring environment there are several reasons why a network designer would consider extended the LAN via bridging. These include the following:

- To create separate traffic domains—The classic justification for installing any bridge; this has minimal impact on the design.
- To avoid accumulated jitter—In Token Ring LANs frames travel down different sections of the cable at slightly different speeds and this results in jitter. To counteract this Token Ring networks limit the number of hosts that can be attached to a LAN (260 for STP or fiber and 72 for UTP, although some vendors may support larger numbers). When a bridge is installed, the jitter is effectively zeroed across bridged interfaces.
- To transport nonroutable protocols over extended LANs—Token Ring environments are typically users of IBM protocols, such as SNA, LU6.2, and NetBIOS. None of these protocols incorporates a routing layer.

The following guidelines are provided when designing SRB extended LANs:

- Bridge IDs—Bridge IDs are a limited resource (a 4-bit quantity). You can use more than 16 bridge IDs by using duplicate bridge IDs, so long as the bridges with the same ID are not running directly in parallel (i.e., not connecting the same two rings directly). In smaller networks it is recommended that you choose unique bridge IDs to simplify the design and avoid routing issues.
- Ring numbers—Ring numbers do not have to be unique within a Token Ring LAN; however, where multiple bridges are attached to a

ring, all bridge ports must use the same ring number. Within a single bridge all ports should be assigned different ring numbers.

- Maximum bridges in series—In implementations where the network radius is constrained to bridge seven hops, this limitation should not be exceeded. Nodes more than seven hops away will not be able to communicate.
- Shortest paths in source routing are response based. In source routing, the shortest path is determined by the fastest response to an Explorer frame. Regardless of the number of hops involved, the fastest response will be chosen.
- Parallel bridges are supported. So long as bridge IDs can be differentiated between two rings it is possible to support multiple bridge paths. Since the paths chosen are based on response time, this can allow both resilient links and some measure of load sharing to be implemented between Token Ring LANs (session based rather than packet based). Source routing enables logical meshing of the traffic.
- Source routing does not scale. Aside from the seven-hop limit, broadcasts from protocols such as ARP, RARP, and BOOTP will propagate throughout the extended LAN, so SRB networks have the same inherent scalability problems of transparent bridging.
- Be aware of performance trade-offs. SRB adds more traffic overhead than conventional bridging, since each frame will contain complete routing information as well as protocol data. On the plus side, SRB bridges have to do very little work when making frame-forwarding decisions.

The designer should be aware that there are potential problems with the A and C bits (MAC layer acknowledgments) when interconnecting with transparent bridges, since the standards have never quite resolved these issues for Token Ring bridges (the FDDI bridging standards do). These issues are discussed in the next section.

9.4 Source Route Transparent Bridging (SRT)

The IEEE 802.1 committee identified the need for source route bridges to interoperate with transparent bridges in the same internetwork. The result was a standard called Source Route Transparent Bridging (SRT). Note, however, that SRT bridges operate only between two Token Ring LANs and are covered in an unapproved IEEE 802.1m standard. SRT bridges do not perform conversion between Token Ring and Ethernet LANs.

9.4.1 Operation

The basic operation of SRT bridges is quite simple. The bridge inspects all received frames and checks the Routing Information Indicator (RII). If the RII is set, the bridge acts as a source route bridge and inspects the Routing Information Field (RIF). If not set, the bridge operates in transparent mode and forwards frames based on their MAC destination address (using information built up in the learning tables). In order to achieve this the Token Ring chipset in the bridge has to be in mixed mode.

SRT bridges do not allow source route bridge nodes to communicate with transparent bridge nodes; even though an SRT bridge can understand both source route bridging and transparent bridging, it does not perform translation between frame types (in effect an SRT bridge has a split personality). SRT bridges also introduce a problem with the A (address recognized) and C (frame copied) bits in the Token Ring frame status field. Source routing bridges use the RIF fields to decide how to set A and C, but bridges in transparent mode do not look at the RIF. This leads to several possible solutions, none of which is perfect, as follows:

- Never set A and C. This means that any frames forwarded by the bridge for remote nodes will return around the ring and be interpreted by the sender as unacknowledged. The sender will continue to retransmit and may subsequently get a response. The sending node may then get very confused, since the reply is good, but there is no record of the receiver acknowledging and copying the original frame (since A and C were never set). Needless to say this can cause a variety of problems.
- Always set A and C. If both the source and destination nodes are on the same ring, this will result in a duplicate address error being sent to the ring error monitor (often the bridge itself) for every frame modified.
- Conditionally set A and C. This is the smarter way of doing things but still problematic. The bridge will set A and C if it forwards frames with the knowledge that the destination node is not on the local ring. The problem is that it can learn of a station's whereabouts only by listening for traffic, and if the receiver has not transmitted any packets to date, then there is the possibility that the bridge forwards the frame, sets A and C, and the local host receives the frame. Again, a duplicate address event is generated, but the scale of the problem is much smaller.

The IEEE 802.1m standard suggests that A and C are always set by the bridge. The error reports for duplicate addresses can be disabled, but this will need to be configured for all rings in the network.

Spanning Tree

Spanning Tree on the transparent bridge interfaces runs independently from the source route Spanning Tree and supports all the traditional functionality. This means that multiple SRT bridges can be placed in parallel, but be aware that any interring source route traffic passing over the transparent network will only go down the active path, and load balancing is not supported.

9.4.2 Design guidelines

The following guidelines are provided when designing SRT extended LANs:

- Applications—Because of the many limitations of SRT, in real life these bridges are often used in source route only mode. SRT bridges can be used to enable interring communication between hosts whose drivers cannot support source route Explorer traffic. Figure 9.14 shows an example where SRT is required between ring 400 and 200.

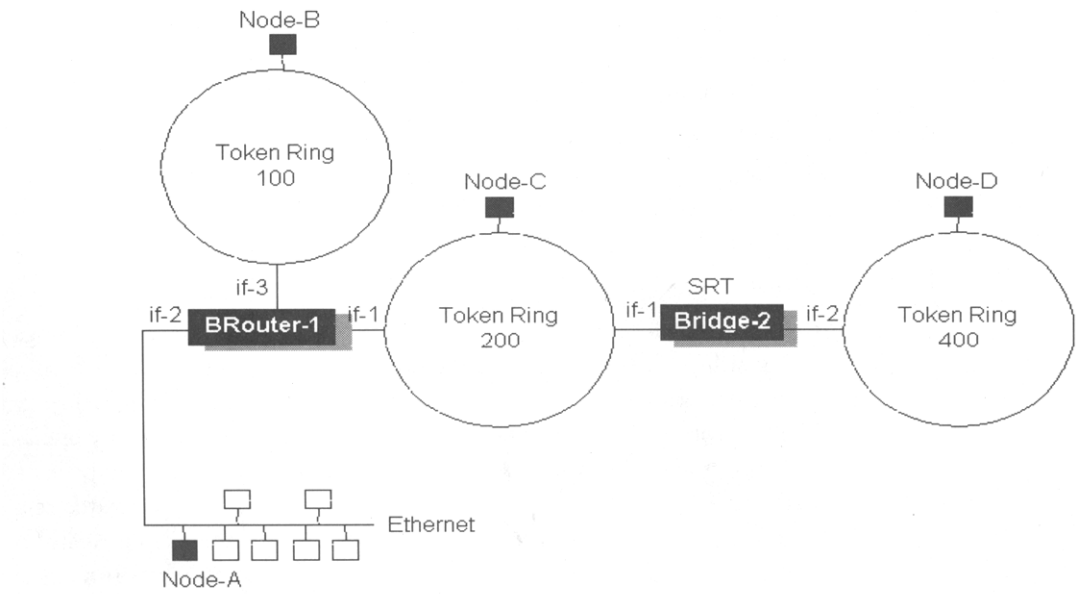

Figure 9.14 *SRT bridges enable interring communication.*

Traffic between ring 100 and 200 does not require SRT; frames have the RII bit set and are source routed by Brouter-1. Traffic between rings 100, 200, and the Ethernet LAN does not require SRT, since the RII bit will be cleared; frames will be sent transparently. Traffic between ring 200 and 400 is source routed as normal. SRT bridging is required where node D, on ring 400, has old drivers that do not support source route Explorer frames.

- Plug-and-play mode—SRT bridges do not need to have ring or bridge IDs configured if they are to operate only in transparent mode (i.e., as transparent Token Ring bridges). In small networks this can be very attractive, since they are quick and easy to deploy.
- SRT and SR bridges in parallel—If you design an SRT bridge and SR bridge in parallel, then the SR bridge will operate only in backup mode (assuming that hosts on the rings can send source route Explorer frames). Since all hosts transmit their first frame with the RII bit clear, the SRT bridge will always forward the traffic.
- End-to-end paths must be consistent—SRT bridges do not perform any frame translation between transparent and source route formats. A pure translation mode frame can never pass through a pure SRB ring, and the end-to-end path must be supported by SRT bridges.

9.5 Translation bridging

Given the huge range of issues that translation bridges need to resolve, some commentators have classified translation bridges as an engineering triumph over common sense. The basic idea with translation bridging is that different MAC layers must be completely translated so that interworking between different media types on different interfaces is entirely transparent and requires no setup by the user. Stations on either side of the bridge communicate as if they were directly connected. Considering that we are talking about media such as Ethernet, Token Ring, and FDDI, each with different MTU sizes, frame formats, and bit ordering, this is no mean feat, especially when you consider that this must all be achieved at wire speed forwarding rates. A translation bridge must deal with a number of difficult issues when converting between different media, as follows:

- Interface speeds
- Bit ordering (Big Endian or Little Endian)
- MAC address formats

- MTU sizes
- Routing mechanism (source routing, transparent, etc.)
- Frame formats (type/length, LSAPs, SNAP)
- Protocol irregularities (e.g., embedded addressing)
- Spanning Tree operations

We will now look at how Token Ring and Ethernet networks can be connected and how FDDI and Ethernet networks can be connected via translation bridges. Note that there is no current standard for translation between FDDI and Token Ring, so most network designers will use either encapsulation bridging or native source route bridging techniques.

9.5.1 FDDI-to-Ethernet translation

In this scenario, translation bridging passes packets between FDDI and Ethernet so that on either medium the packets appear to be from a locally attached device. This operation is covered by the IEEE 802.1h standard. Figure 9.15 shows an example network with an FDDI backbone and two Ethernet LANs attached via translation bridges. IEEE 802.h operates the same Spanning Tree for Ethernet and Token Ring. In Figure 9.15 both the FDDI and Ethernet interfaces act as independent transparent tree bridges, compatible with IEEE 802.1.

In this example, hosts A, B, C, and D can all communicate transparently. Frames in transit between the Ethernet LANs are placed on the FDDI ring and forwarded between the two bridges. To comply with FDDI MAC procedures, these transit frames must be removed from the ring; however, since the frames do not have source MAC addresses that match any device

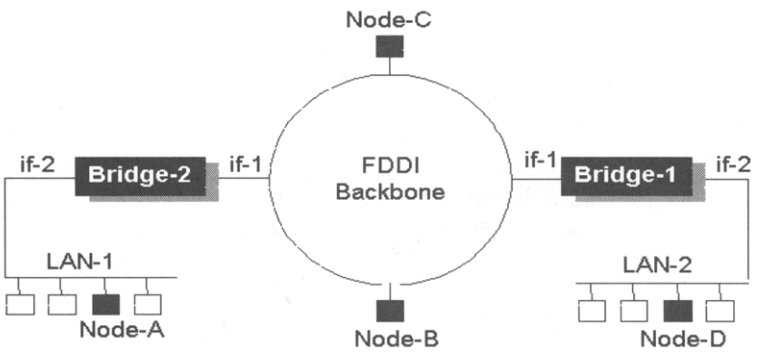

Figure 9.15 *Ethernet-to-FDDI translation bridging.*

on the ring, they cannot be removed in the normal way (usually this would be done by the sending device on the ring). To complicate matters further, the standards in this area specify only that frames should be removed but do not specify the mechanism. In practice, bridges either maintain a database of source addresses for frames transmitted or transmit a marker frame on the ring to indicate when the bridge should remove frames.

As with any IEEE 802.1–compliant bridge, all interfaces of the bridges operate in mixed learning mode. On the FDDI interfaces this requires a very high filtering rate from the FDDI chipset, so these bridges typically use Content Addressable Memory (CAM) to optimize forwarding lookups. The maximum rate for an FDDI ring is 446,000 fps using 28-byte frames. On real networks the frame size is typically much larger and frame rates lower, since FDDI is usually employed as a backbone interconnect between LANs for remote file transfer client/server interaction. Another issue is the size of the learning table, which tends to be large on such bridges, since they are typically used as backbone devices and stored in CAM. Table size can limit scalability, since bridge networks have a flat address space, so some high-end vendors allow the tables to be extended.

Design issues

There are several nontrivial issues to be resolved in the translation process, including the following:

- Bit ordering—Ethernet uses noncanonical bit ordering (least significant bit first), and FDDI uses canonical ordering (least significant bit last). Translation is resolved by the bridge internally and is not a concern for the network designer. However, one potential area of conflict arising from bit reversal is address translation.
- Address resolution—ARP and RARP pose particular problems in this environment and require specific software enhancements in the bridges to cope. The problem here is that these protocols carry addressing information in their data fields, and, since bit ordering is reversed between media types, this addressing information becomes reversed on a byte level. For example, 1.1.1.1 becomes 128.128.128.128 when bit reversed. Either the bridge must explicitly deal with ARP and RARP by reversing the relevant data fields, or the drivers must automatically translate addresses recovered from data fields according to the bit order preference of the interface. The latter option is recommended by RFC 1042 [2] and adopted by 802.1h bridges, which somewhat simplifies matters.

- Encapsulation—Frame formats on IEEE 802.3 and FDDI differ in several ways. A native EtherType encapsulated protocol would typically be translated into SNAP format on the FDDI, which is straightforward enough. Things may get problematic when the frame comes back off the FDDI onto the LAN, since it could be converted back into either EtherType, IEEE 802.2 encapsulation (length field plus LSAPs), or SNAP. Typically the bridge will allow you to configure the preferred encapsulation per interface. In reality you may have an application that uses IEEE 802.2 encapsulation at the server end, uses SNAP encapsulation on the backbone, and uses EtherType encapsulation at the client. Debugging such a session might prove interesting.
- MTU size—MTU size is an issue between Ethernet and FDDI. FDDI supports up to 4,352-byte frames, whereas Ethernet supports only 1,518-byte frames. Bridges cannot handle fragmentation and reassembly, so the only real course of action is to limit the MTU size on the end systems, which is consistent with IEEE 802.3h and RFC1042 [2].
- Mixed-speed buffering—FDDI and basic Ethernet operate at different speeds (100 Mbps versus 10 Mbps). Bridges do not offer any form of flow control but should have large buffers available to cope with most reasonable conditions. Under sufficient load an 802.1h bridge will silently discard packets (typically from FDDI to Ethernet). For example, a Sun workstation directly attached to an FDDI backbone could easily outpace an Ethernet-attached receiver with a UDP stream (TCP will typically back off according to the prevailing conditions).

MTU inconsistencies are a typical problem for the network designer. For example, the FDDI hosts in Figure 9.15 would have their MTU reduced to 1,500 bytes. This resolves the problem but can degrade performance on the FDDI; in particular, it penalizes high-bandwidth applications between native FDDI devices unnecessarily. The only other sensible option would be to use routing between the FDDI and Ethernet LANs and let the routing protocol use MTU Discovery to resolve the different MTU sizes optimally.

Encapsulation is also a likely area of conflict. Most translation bridges will by default translate all FDDI 802.2/SNAP frames to EtherType format (except for AppleTalk EtherTypes). On Ethernet it is, therefore, important not to have interfaces configured in 802.2./SNAP format; otherwise, communications will be lost. On older networks there may also be the issue with Novell raw frame-type encapsulation to contend with (a broken form of LLC whereby a length field is used instead of the EtherType field, but no

LSAPs). Another potential issue here is when routers and bridges are used on the same ring. Routers should be configured to use both the FDDI 802.2 and FDDI 802.2/SNAP encapsulations if stations on the Ethernet segments are configured to use different Ethernet encapsulation types.

9.5.2 Token Ring to Ethernet translation

This type of bridging is often called Source Route Translation Bridging (or SRTB) and is synonymous with the IBM 8209 translation bridge [3]. IBM 8209 is a two-port device that allows Ethernet and Token Ring hosts to communicate transparently using nonroutable protocols such as LU6.2 and NetBIOS. This, in effect, allows IBM SNA networks to deploy Ethernet LANs for workgroup users, while connecting SNA hosts onto Token Ring via cluster controllers and Front-End Processors (FEPs). Table 9.1 presents an Ethernet to Token Ring translation bridging example.

In Figure 9.16 we see two Token Ring LANs connected via a source routing bridge and an Ethernet LAN attached via a source route translation bridge. On Bridge-1, interface 2 must use mixed learning mode to build up its forwarding tables and operates as a standard Spanning Tree transparent bridge. On interface 1 Bridge-1 must use source routing; forwarding is determined by sending all-routes Explorer frames and caching the RIF responses. On the Token Ring interface the Spanning Tree protocol is used to create a single path for Explorer traffic. Note that the two Spanning Tree processes are independent of each other; there is no interaction. Spanning Tree BPDUs from either side are carried over each media type. The bridge uses its internal mapping database to make forwarding decisions between the two different media types. From the perspective of an Ethernet node the Token Ring networks appear as if they were a single Ethernet segment. From the Token Ring perspective the Ethernet appears to be a single virtual ring segment.

Table 9.1 *Ethernet to Token Ring Translation Bridging*

Ethernet				Token Ring			
if	NodeID	Encap	Age	if	NodeID	RIF	Age
2	A	V2	32	1	C		24
2	B	802	12	1	D		6
				1	E	[200]+[1]+[400]	45
				1	F	[200]+[1]+[400]	17

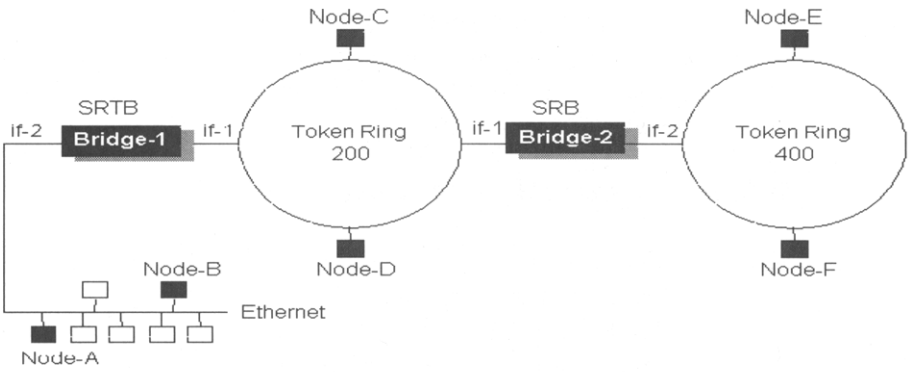

Figure 9.16 *Ethernet to Token Ring translation database mapping for Bridge-1.*

Design issues

- Bit ordering—Ethernet uses noncanonical bit ordering (least significant bit first), and Token Ring uses canonical ordering (least significant bit last). Translation is resolved by the bridge internally and is not a concern for the network designer.
- MTU size—As with FDDI translation, MTU size is potentially a major problem between Ethernet and Token Ring. Token Ring supports up to 4,500 bytes at 4 Mbps and 17,749 bytes at 16 Mbps, whereas Ethernet supports only 1,518-byte frames. Fragmentation and reassembly are not sensible solutions for bridging, so we should limit the MTU size on the end systems (potentially degrading performance for some Token Ring hosts). IBM 8209 gets around this issue by dynamically overwriting the 3-bit largest frame field within the Token Ring Explorer frame's Route Control Field (RCF) with the binary value 001 (indicating 1,500 bytes). This field specifies the maximum frame size to be used, and since it is set dynamically there is no need to configure all hosts to this value.
- Encapsulation—Frame formats on Ethernet/802.3 and Token Ring differ in several ways. EtherType frames being forwarded from Ethernet to the Token Ring interface are SNAP encoded. Going back the other way the bridge will examine its internal database and see which encapsulation the host is using before converting back into EtherType or IEEE 802.3 encapsulation. If the bridge does not know the correct encapsulation (e.g., the destination host may have aged out), then it can transmit duplicate frames with both types until the database is updated.

- Address resolution—As with FDDI we have the same problem with ARP and RARP. In this case IBM 8209 checks all packets of type 0x806 (ARP) or 0x8035 (RARP) and dynamically changes the bit order of embedded addressing data (located at fixed offsets) according to the bit order used on the appropriate interfaces. Novell IPX also embeds host MAC addresses in the Layer 3 address field (avoiding the need to ARP). Since these addresses appear in several places and under several different encapsulations, it would be very tedious to dig all these addresses out and convert on the fly. IBM 8209 enables the user to configure a feature called NetWare Awareness, which basically means that the bridge will not convert the MAC addresses portion of NetWare frames received. This can cause further problems, since frames on one side of the bridge will appear inverted and in some cases will appear as multicasts. There is no easy solution to this; in some cases the network administrator may have to identify specific problem MAC addresses and configure new soft addresses in those hosts. At the time of writing IBM 8209 did not forward AppleTalk, DECnet (Phase IV and Phase V), or OSI packets, since these would require similar fixes. Clearly, it would be advantageous for any IBM 8209–compatible products to implement multiprotocol routing to handle these protocols appropriately.

There is no real standard for translation bridging between Token Ring and Ethernet; it is an industry standard defined effectively by IBM 8209. The IEEE specifications 802.1d, 802.5d, and 802.1h cover part of the functionality required, but since IBM 8209 is neither a transparent bridge nor a source route bridge, there are features not specified in the standards. Several major bridge vendors have implemented the SRTB because of the requirement to interwork between source route bridge domains and transparent bridge domains. Some of the key vendors are IBM, Cisco, Proteon, and BAY. Many of these devices are multiport devices with support for wide area interfaces (remember that 8209 has only two LAN interfaces). In wide area translation devices there is an additional issue to solve—namely, where to perform the translation. Most devices from the main manufacturers will perform conditional translation on either incoming or outgoing interfaces according to the source and destination media types.

9.5.3 Token Ring to FDDI translation

In practice these devices are rarely deployed in networks today, although this functionality may be embedded in multiprotocol bridge router plat-

forms. The translation process is generally far simpler than for FDDI to Ethernet translation, since FDDI and Token Ring share many similarities, not the least of which are bit order and encapsulation types. Clearly, MTU size is a potential issue, with 16-Mbps Token Ring supporting an MTU nearly four times larger than FDDI (17,749 bytes versus 4,500 bytes). To resolve this problem either Token Ring stations must be configured to use the FDDI MTU, or routing must be enabled to support fragmentation.

Another major area of conflict is the use of Source Route Bridging (SRB) and the presence of the RIF field in Token Ring frames. To resolve this problem translation bridges typically implement both a forwarding database (for transparent bridging) and a routing database (for source route bridging) and map between the two environments. In this case the source-routed environment treats the whole of the transparent environment as a single ring, with its own ring ID. Implementations may vary in this area, and you should consult the manufacturer's documentation on this matter.

9.6 Encapsulation/tunnel bridging

Encapsulation bridges come in a number of forms; we will briefly review the more important ones here.

9.6.1 Data Link Switching (DLSw)

Encapsulation (or tunnel) bridges can be used to enable Source Route Bridging (SRB) domains or transparent bridging domains to communicate across an IP internetwork. This feature is generally referred to as Data Link Switching (DLSw). The encapsulation bridge encapsulates bridged frames (from either an SRB or TB domain) inside IP packets and forwards them to the destination IP address, via standard Layer 3 IP routing protocols. The destination IP address is, in fact, another encapsulation bridge. The destination target encapsulation bridge removes the IP encapsulation to reveal the original frame, and then forwards these frames to its attached SRB domain or TB domain, as if the domain were local. From the SRB perspective the IP network is perceived as a single LAN segment, and only one hop is added to cross this IP network. The end-to-end hop count must not exceed the seven-hop count limitation imposed by some SRB implementations (this includes the number of hops from the source node to the source IP encapsulation bridge, plus one hop to cross the IP network, plus the number of hops from the destination IP encapsulation bridge to the destination node).

9.6.2 FDDI encapsulation bridging

Encapsulation bridges were the first types of bridges used to interconnect Ethernet and FDDI LANs, since the standards available at that time were too immature to consider translation. In effect, the entire Ethernet frame is simply encapsulated inside an FDDI frame, passed over the ring, and then deencapsulated by a remote FDDI bridge before forwarding onto another Ethernet LAN. Most FDDI encapsulation bridges run proprietary protocols over the FDDI ring between bridges to ensure that bridges are aware of each other. They may also advertise forwarding tables between bridges.

Encapsulation avoids many of the pitfalls and complications of translation bridging but at the expense of any communication between Ethernet and FDDI-attached stations (only Ethernet-to-Ethernet communications are supported, with the FDDI treated as a high-speed transit LAN). This could be a nonstarter if high-speed servers are attached to the FDDI ring and need to be accessible by Ethernet clients. Interoperability is also clearly one of the main concerns for a network designer; bridges typically need to be supplied from a single vendor. As a result this method of bridging is rarely used for this application, with translation bridges being the preferred option.

9.7 Virtual LANs (VLANs)

This section examines the design issues of using LAN switches when implementing virtual LANs (VLANs) and switched internetworks. VLANs are mainly applicable for campus network designs, but they are unlikely to be used exclusively. A good campus internetworking solution must combine the benefits of both routers and switches. The main benefits of incorporating switches in campus network designs (as opposed to pure bridged or routed environments) include the following:

- Higher bandwidth
- Better performance
- Lower costs
- Simplified configuration

VLAN switches impose a logical hierarchy on flat networks (unlike bridges), but they require routers to deal with interdomain traffic. VLANs resolve several major problems in flat networks, including the following:

- Scalability—A flat LAN (i.e., nonhierarchical) is a single broadcast domain and, hence, inherently not scalable because broadcast traffic becomes a much more significant proportion of the bandwidth as the number of hosts and services increases. VLANs resolve scalability problems by breaking this broadcast domain into several smaller broadcast domains called virtual LANs (essentially similar to IP subnets but based on logical addresses transparent to the user, rather than IP addresses). Broadcasts are, therefore, localized within each VLAN, and this logical segmentation is commonly referred to as VLAN communication. For example, in a broadcast domain consisting of 100 users, if the broadcast traffic is intended only for 30 of the users, then placing those 30 users on a separate VLAN will better optimize bandwidth. When compared with switches, routers have to do more processing on incoming packets. As the volume of traffic increases, so does latency, which results in performance degradation. The use of VLANs reduces the number of routers required, since VLANs create broadcast domains using Layer 2 switches instead of routers.
- The ability to form virtual workgroups—The ability to move users around easily, transparently, and cost-effectively has become an essential requirement for many enterprise networks. Since physical hardware addresses are typically fixed, the other problem VLANs are designed to resolve is how to handle moves and changes in an enterprise environment without having to constantly reassign addresses or reconfigure user workstations and ports (or worse still, reconfigure bridge filter tables). Consider the situation where the whole finance department is to be relocated to another building on a campus at short notice, and one-third of them will remain in the original office for a week. To create virtual LANs, a VLAN ID is associated with each interface on the switch, so when a user workstation is plugged into that interface, the VLAN ID is automatically associated with the user but centrally controlled by the network administrator. In this way two users located next to one another could be physically connected to the same switch but logically connected to different sets of resources. This technology would, for example, enable virtual teams to be set up without the need to physically locate everybody in the same physical workspace.
- Simplified administration—Some sources report that as much as 70 percent of network costs is a result of adds, moves, and changes of users in the network. Every time a user is moved the administrator potentially has to rewire, change station addressing, reconfigure hubs

and routers, and so on. VLANs simplify most of these tasks. If a user is moved within a VLAN, reconfiguration of routers is unnecessary. In addition, depending on the type of VLAN, other administrative work can be reduced or eliminated. Despite this saving, VLANs do add an extra layer of administrative complexity.

- LAN security—Sensitive data may be compromised by the broadcast nature of a local area network, and installing firewalls at the departmental layer may be viewed as prohibitively expensive. In such cases, placing only those users who can have access to that data on a VLAN improves security. VLANs can also be used to control broadcast domains, set up firewalls, restrict access, and inform the network manager of an intrusion.

9.7.1 Operation

VLANs are a relatively new abstraction of the way traffic domains are created in LAN environments. Essentially it was thought that a transparent mechanism was required to offer Layer 3 functionality within Layer 2 environments. Switches and routers each play an important role in VLAN design.

- Switches provide connectivity and communication for individual VLAN users, while routers provide inter-VLAN communication. VLAN switches free the designer from the physical constraints imposed by shared-hub architectures (within each wiring closet). On a VLAN users and ports are grouped logically across the LAN, irrespective of their actual location or physical connectivity.
- Routers provide inter-VLAN communication, as well as VLAN access to shared resources such as servers and external services. In addition routers may also provide firewall functionality, broadcast suppression, policy-based control, broadcast management, and route processing and distribution.

LAN switches are typically used to segment networks into virtual workgroups, and each workgroup VLAN is typically associated with a VLAN ID—for example, VLAN 3. When a VLAN switch or bridge receives data from a workstation, it tags these data with a VLAN identifier indicating the VLAN from which these data originated. This is called explicit tagging. It is also possible to determine to which VLAN data received belong by using implicit tagging. In implicit tagging the frame is not tagged, but the source VLAN is determined based on other information such as the port on which

these data arrived. Note that it is not always necessary to have a router to provide access to shared resources. For example, a server port on a security VLAN switch may be configured with several VLAN IDs, thus providing service to more than one group of users. The overlapping VLANs are still effectively separated within the switch, but each has access to the server. Note that the overlapping VLANs configured at the server port cannot communicate directly (there is no path between different VLANs other than through the server).

Figure 9.17 illustrates several VLAN design concepts. VLANs are defined at the switch port level; physical location is transparent. In this example Cisco's Inter-Switch Link (ISL) protocol is used between switches to multiplex VLANs over higher-speed campus backbone links (this could equally be another vendor-specific protocol such as Cabletron's ISMP). Spanning Tree is run on a per-VLAN basis, and this can be configured

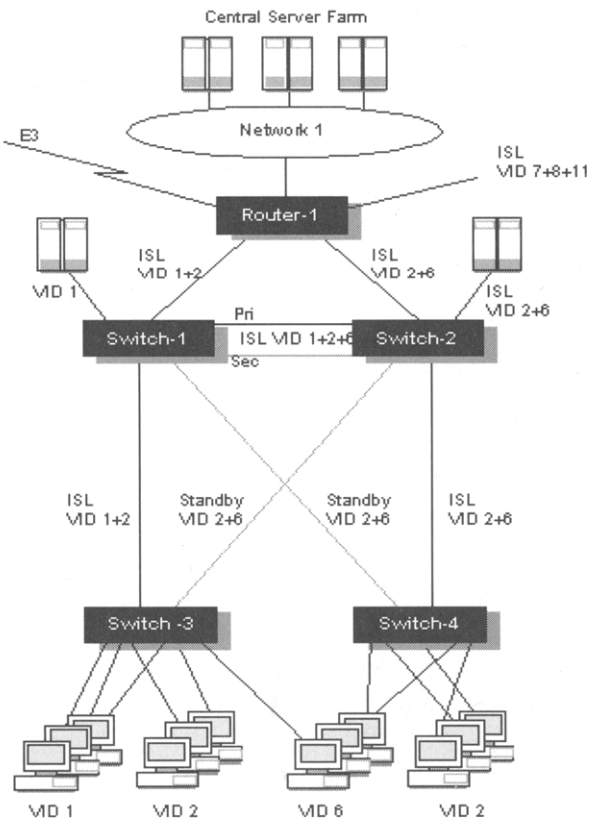

Figure 9.17 *VLAN design concepts.*

either on physical links (for high resilience) or over existing ISL links. The router enables inter-VLAN communications and access to the central server farm and external services. Some servers are directly attached to switches, either tied to a specific VLAN or via NIC cards that support multiple VLANs. Note the primary and backup ISL links between Switch-1 and -2. This ensures that VLAN 2 is contiguous and does not require the router in the forwarding path. These links could also be configured with alternative VLAN IDs to load-balance and maintain both links live.

In general the majority of end systems are not aware of tagging mechanisms (such as 802.1Q [4]). One of the key advantages of VLANs is that they are totally transparent to the end user, so tagging is not normally seen outside of VLAN trunk ports, which connect Inter-Switch VLANs (ISVLANs). In general, switches will use tagging on dedicated trunk ports and will strip the tags before forwarding the frame into a user segment. Another reason for this is that tagging increases the frame size and may, therefore, create an illegal frame (i.e., exceeding the MTU), which would be rejected by most end-station stack drivers (particularly if using 802/LLC or SNAP encapsulations).

Tagging can be based on the port from which it came, the source MAC address, the source network address, or some other field or combination of fields. VLANs are classified based on the method used. To be able to do the tagging of data using any of the methods, the switch would have to keep an updated database containing a mapping between VLANs and whichever field is used for tagging. For example, if tagging is by port, the database should indicate which ports belong to which VLAN. This database is called a filtering database. Switches would have to be able to maintain this database and also to make sure that all the bridges on the LAN have the same information in each of their databases. The switch determines where data are to go next based on normal LAN operations. Once the switch determines where these data are to go, it now needs to determine whether the VLAN identifier should be added to these data and sent. If these data are to go to a device that is VLAN-aware, the VLAN identifier is added to these data. If these data are to go to a device that has no knowledge of VLANs, the bridge sends these data without the VLAN identifier.

In order to understand how VLANs work, we need to look at the types of VLANs; the types of connections between devices on VLANs; the filtering database that is used to send traffic to the correct VLAN; and tagging, a process used to identify the VLAN originating data.

Types of VLAN

VLAN membership can be classified by port, MAC address, and protocol type, as follows:

- Layer 1 VLAN: Membership by port—Membership in a VLAN can be defined based on the ports that belong to the VLAN. For example, in a VLAN bridge with four ports, ports 1, 2, and 4 could be assigned to VLAN 1, and port 3 could be assigned to VLAN 2. One disadvantage of this method is that user mobility may be compromised if the number of spare switch ports configured with the appropriate VLAN ID is small. If a user moves to a different location away from the assigned bridge, the network manager may have to reconfigure VLAN ports if free ports are in short supply and not configured with the correct VLAN ID.
- Layer 2 VLAN: Membership by MAC address—Here, membership in a VLAN is associated with the MAC address of the workstation. The switch tracks the MAC addresses that belong to each VLAN. Since MAC addresses form a part of the workstation's network interface card, when a workstation is moved, no reconfiguration is needed to allow the workstation to remain in the same VLAN. This is unlike Layer 1 VLANs where membership tables must be reconfigured. The main problem with this method is that VLAN membership must be assigned initially. In networks with thousands of users, this is no easy task. Also, in environments where notebook PCs are used, the MAC address is associated with the docking station and not with the notebook PC. Consequently, when a notebook PC is moved to a different docking station, its VLAN membership must be reconfigured.
- Layer 2 VLAN: Membership by protocol type—VLAN membership for Layer 2 VLANs can also be based on the protocol type field found in the Layer 2 header. For example, IP could be associated with VLAN 1, IPX could be associated with VLAN 3, and so on.
- Layer 3 VLAN: Membership by IP subnet address—Membership is based on the Layer 3 header. The network IP subnet address can be used to classify VLAN membership. For example, subnet 193.127.34.128 could be associated with VLAN 1, and subnet 140.5.64.0 with VLAN 2. Although VLAN membership is based on Layer 3 information, this has nothing to do with network routing and should not be confused with router functions. In this method, IP addresses are used only as a mapping to determine membership in VLANs. No other processing of IP addresses is done. In Layer 3

VLANs, users can move their workstations without reconfiguring their network addresses. The only problem is that it generally takes longer to forward packets using Layer 3 information than using MAC addresses.

- Higher-layer VLANs—It is also possible to define VLAN membership based on applications or service, or any combination thereof. For example, File Transfer Protocol (FTP) applications can be executed on one VLAN and Telnet applications on another VLAN.

Types of connection

Devices on a VLAN can be connected in three ways, based on whether the connected devices are VLAN-aware or VLAN-unaware. Recall that a VLAN-aware device is one that understands VLAN memberships (i.e., which users belong to a VLAN) and VLAN formats. The three connection methods are as follows:

- Trunk link—All the devices connected to a trunk link, including workstations, must be VLAN-aware. All frames on a trunk link must have a special header attached. These special frames are called tagged frames.
- Access link—An access link connects a VLAN-unaware device to the port of a VLAN-aware bridge. All frames on access links must be implicitly tagged (untagged). The VLAN-unaware device can be a LAN segment with VLAN-unaware workstations, or it can be a number of LAN segments containing VLAN-unaware devices (legacy LAN).
- Hybrid link—This is a combination of the previous two links. This is a link where both VLAN-aware and VLAN-unaware devices are attached. A hybrid link can have both tagged and untagged frames, but all the frames for a specific VLAN must be either tagged or untagged.

It should be noted that the network could have a combination of all three types of link.

VLAN protocols

VLAN switches require mechanisms to discover neighbor and topological information. VLAN switches also require a mechanism to ensure that VLAN traffic passed between them is distributed to the relevant interfaces. Since VLAN IDs are transparent to users, this requires that the IDs be inserted and removed from the LAN protocol stack by VLAN equipment.

There are several approaches that can be taken, and typically this involves tagging frames with VLAN information. There are a number of standard and proprietary protocols used to support VLAN operations. These include the following:

- IEEE 802.1Q tagging
- An adaptation of the IEEE 802.10 security protocol—Cisco et al. proprietary
- VLAN Hello protocol—Cabletron proprietary
- Virtual LAN Link State Protocol (VLSP)—Cabletron proprietary
- The Inter-Switch Link (ISL) protocol—Cisco proprietary
- VLAN Trunk Protocol (VTP)—Cisco proprietary

This list is not exhaustive, and a brief overview is provided in the following list:

- IEEE 802.1Q—IEEE 802.1Q reflects a move toward developing a common set of standards for VLAN products. In the past, VLAN standards were incomplete, and products typically interoperated via proprietary protocols, meaning that a customer wanting to install VLANs would have to purchase all products from the same vendor. IEEE 802.1Q is a VLAN tagging protocol that effectively provides the same functionality as Cisco's ISL. 802.1Q requires that a tag be inserted between the source MAC address and the EtherType or length field. The tag also carries a flag to indicate a Token Ring frame, so that Token Ring frames can be carried in native format over Ethernet, without requiring 802.1H translation. The 802.1Q draft standard defines Layer 1 and Layer 2 VLANs only. Protocol type–based VLANs and higher-layer VLANs have been allowed for but are not defined in the standard. Note that 802.1Q defines only the VLAN ID; 802.1P defines VLAN membership. Both 802.1P and 802.1Q share common tagging. Most of the major vendors now support these standards and either have released or are planning on releasing products based on them.
- IEEE 802.10—Several vendors have adapted the IEEE 802.10 security protocol so that VLAN IDs can be carried within LAN frames between the MAC and LLC headers. This information is inserted and stripped off by IEEE 802.10–compliant switches at the ingress and egress switch nodes, respectively. End systems send and receive standard frames so that VLAN operation is handled transparently within the switched cloud.

- VLAN Hello protocol—The VLAN Hello protocol is defined by Cabletron in RFC2641 [5]. This RFC is informational. VLAN Hello is part of the InterSwitch Message Protocol (ISMP), which provides interswitch communication between switches running Cabletron's SecureFast VLAN (SFVLAN) product RFC2643 [6]. Switches use the VLAN Hello protocol to discover their neighboring switches and establish the topology of the switch fabric.
- Virtual LAN Link State Protocol (VLSP)—Developed by Cabletron and defined in RFC2641 [5]. This RFC is informational. VLSP is part of the InterSwitch Message Protocol (ISMP), which provides interswitch communication between switches running Cabletron's SecureFast VLAN (SFVLAN) product RFC2643 [6]. VLSP is used to determine and maintain a fully connected mesh topology graph of the switch fabric. Each switch maintains an identical database describing the topology. Call-originating switches use the topology database to determine the path over which to route a call connection. VLSP provides support for equal-cost multipath routing and recalculates routes quickly in the face of topological changes, utilizing a minimum of routing protocol traffic. VLSP is derived from the OSPF link-state routing protocol.
- Inter-Switch Link (ISL) protocol—This protocol was developed by Kalpana, Inc. (a pioneer of switches) and subsequently acquired by Cisco. ISL is designed to carry VLAN information over 100-Mbps Ethernet switch-to-switch or switch-to-router links (called trunks) between access and distribution layer switches in a campus environment. ISL maintains VLAN information as traffic flows between switches. With ISL, an Ethernet frame is encapsulated with a 30-byte header that contains a 2-byte VLAN ID. ISL can multiplex multiple VLANs over a single link or distribute VLANs over multiple links. Some server NICs support native ISL operation and so enable application servers and file servers to participate in multiple VLANs directly with the switch.
- VLAN Trunk Protocol (VTP)—Cisco has implemented the proprietary VLAN Trunk Protocol (VTP), which is used to automate VLAN tagging in multimedia (ATM, Ethernet, FDDI, etc.) switching environments. VTP is a switch-to-switch and switch-to-router protocol that automatically configures VLANs over campus networks regardless of the physical media types. VTP also enables managers to move users easily without service disruption by allowing the addition, deletion, and renaming of VLANs on a campus-wide basis without

the need to manually configure each switch. VTP also enables new switches and routers to be automatically configured with VLAN information as they are brought online.

Frame processing

A bridge, on receiving data, determines to which VLAN data belong either by implicit or explicit tagging. In explicit tagging a tag header is added to data. The bridge also keeps track of VLAN members in a filtering database, which it uses to determine where data are to be sent. Following is an explanation of the contents of the filtering database and the format and purpose of the tag header [4].

Filtering database

Membership information for a VLAN is stored in a filtering database. The filtering database consists of the following types of entries:

- Static entries—Static information is added, modified, and deleted by management only. Entries are not automatically removed after some time (aging), but must be explicitly removed by management. There are two types of static entries: Static filtering entries specify for every port whether frames to be sent to a specific MAC address or group address and on a specific VLAN should be forwarded or discarded, or should follow the dynamic entry. Static registration entries specify whether frames to be sent to a specific VLAN are to be tagged or untagged and which ports are registered for that VLAN.
- Dynamic entries—Dynamic entries are learned by the bridge and cannot be created or updated by management. The learning process observes the port from which a frame with a given source address and VLAN ID (VID) is received and updates the filtering database. The entry is updated only if all the following three conditions are satisfied: the port allows learning, the source address is a workstation address and not a group address, and there is space available in the database.

Entries are removed from the database by the aging-out process, where, after a certain amount of time specified by management (10 to 1,000,000 seconds), entries allow automatic reconfiguration of the filtering database if the topology of the network changes. There are three types of dynamic entry, as follows:

- Dynamic filtering entries.—specify whether frames to be sent to a specific MAC address and on a certain VLAN should be forwarded or discarded.

- Group registration entries—indicate for each port whether frames to be sent to a group MAC address and on a certain VLAN should be filtered or discarded. These entries are added and deleted using Group Multicast Registration Protocol (GMRP). This allows multicasts to be sent on a single VLAN without affecting other VLANs.
- Dynamic registration entries—specify which ports are registered for a specific VLAN. Entries are added and deleted using Generic Attribute Registration Protocol (GARP) and Generic VLAN Registration Protocol (GVRP), as described in Section 9.1.5.

GVRP is used not only to update dynamic registration entries but also to communicate the information to other VLAN-aware bridges. In order for VLANs to forward information to the correct destination, all the bridges in the VLAN should contain the same information in their respective filtering databases. GVRP allows both VLAN-aware workstations and bridges to issue and revoke VLAN memberships. VLAN-aware bridges register and propagate VLAN membership to all ports that are a part of the active topology of the VLAN. The active loop-free network topology is determined via the Spanning Tree algorithm. Once an active topology (which may comprise several VLANs) is determined, the bridges determine an active topology for each VLAN. This may result in a different topology for each VLAN or a common topology for several VLANs. In either case, the VLAN topology will be a subset of the active topology of the network.

Tagging

VLAN operations require that frames carry some form of identification to indicate the VLAN to which they belong. To achieve this, a field called a tag header is added to standard LAN frames between Layer 2 and Layer 3. The tag format is specified by IEEE 802.1Q [4] and IEEE 802.3ac [7]. The tagged frames that are sent across hybrid and trunk links; it is the responsibility of VLAN-aware devices to tag and untag frames accordingly so that these operations are transparent to the user. There are two formats of the tag header, as follows:

- Ethernet tag header—In Ethernet frames the tag header comprises a two-byte Tag Protocol Identifier (TPID) and a two-byte Tag Control Information (TCI) field. The TPI is transposed over the standard EtherType field with a special value of 0x8100, and the EtherType field is shifted so that it follows the TCI field.
- Token Ring and FDDI tag header—In Token Ring and FDDI networks the tag headers comprise a SNAP-encoded TPID and TCI.

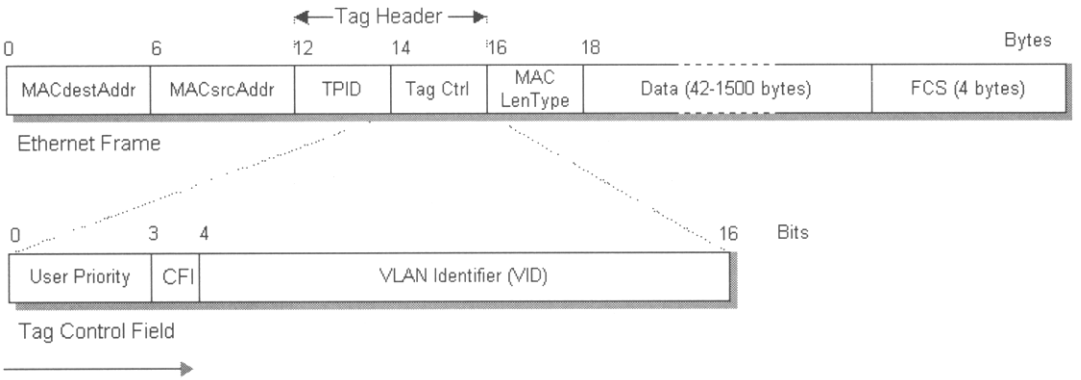

Figure 9.18 *VLAN tagging with Ethernet.*

The SNAP header is set to 0xAAAA03, and the SNAP PID is set to 0x000000. These are followed by the TPI, which is set to 0x8100, and then the TCI.

The format of the Ethernet tag header is shown in Figure 9.18.

TCI field definitions

- User priority (3 bits)—enables priority information to be encoded in the frame. Eight levels of priority are allowed (zero is the lowest priority and seven is the highest priority).
- CFI (1 bit)—used to indicate that all MAC addresses present in the data field are in canonical format (used to decode encapsulated frames that may be present in the data field, such as a Token Ring frame). This field is interpreted differently depending on whether it is an Ethernet-encoded tag header or a SNAP-encoded tag header.
  - In SNAP-encoded TPID the field indicates the presence or absence of the canonical format of addresses.
  - In Ethernet-encoded TPID, it indicates the presence of the source routing information field after the length field. This field indicates routing on Ethernet frames.
- VID (12 bits)—an unsigned binary value used to uniquely identify the VLAN to which the frame belongs. There can be a maximum of $(2^{12} - 1)$ VLANs, with the exception of the following reserved values:
  - 0x000—The NULL VID. Although no VID is used, this allows priority to be encoded in nonpriority LANs.
  - 0x001—The default VID. This is the default VID set on switch ports, which may be reconfigured.

- 0xFFF—Reserved for implementation use. Has significance inside a VLAN-aware device but should not be used externally.

9.7.2 Design guidelines

Network designers can use switches and routers to evolve their shared-media networks to a campus switched internetwork architecture. Typically, this evolution involves the following stages:

- Microsegmentation—existing hubs and routers are maintained; however, a LAN switch is added to enhance connectivity to improve performance.
- High-speed backbone technology—is added (such as ATM, Fast Ethernet, or Gigabit Ethernet), together with routing between switches. LAN switches perform switch processing and provide dedicated bandwidth to the desktop and to shared-media hubs. Backbone routers are attached to either Fast Ethernet or ATM switches. The increase in backbone bandwidth matches the increased bandwidth in the wiring closet.
- Router distribution—between the LAN switches in the wiring closet and the high-speed core switch. The network backbone is now strictly a high-speed transport mechanism with all other devices, such as the distributed routers, at the periphery.
- End-to-end switching—with integral VLANs and multilayer switching capability. Layer 2 and Layer 3 integrated switching is distributed across the network and is connected to the high-speed core.

However, despite the advantages of VLAN networks they can prove hard to manage and optimize. If logical VLANs are widely distributed across a physical LAN infrastructure, then intra-VLAN traffic needs to be dispersed to all physical connections. In a large, well-distributed campus VLAN this can lead to serious traffic overheads and subsequent performance degradation. Other design issues with VLANs include the following:

- Virtual workgroup resources—One of the problems with virtual workgroups is that shared resources may not be conveniently located for all users. A virtual workgroup may be spread over multiple floors and between buildings, whereas the main printing and server resources may be housed centrally. Without duplicating resources this is an unavoidable issue.
- Server farms—Another problem with virtual workgroups is the implementation of centralized server farms. Server farms can be prob-

lematic if servers cannot access more than one VLAN. In such a case, there are three options: The server farm would sit on a single VLAN, and all other VLANs needing to access the server would have to go through a router, with reduced performance; a special server NIC card would need to be installed to support multiple VLAN Ids; overlapping VLAN IDs would need to be configured at the server switch ports.

- Multiple Spanning Trees—Standard switching and bridging can often result in nonoptimal forwarding of packets, since every packet must travel down the minimum Spanning Tree, regardless of the physical proximity of the source and destination node. Routers generally offer more optimal paths, since each path is typically a least-cost path for each source and destination pair. To counter this, some VLAN vendors (such as Cisco) provide support for improved routing and redundancy in switched environments by supporting a single Spanning Tree per VLAN. Note that standard Spanning Tree bridges can implement separate Spanning Trees, but this requires the configuration of different Spanning Tree multicast addresses and very careful topological implementation to avoid loops.

VLANs can provide major benefits in key areas such as LAN administration, security, and network management in enterprise LANs. They require a fundamental change in the way that LANs are designed and managed, and there are several issues network designers should consider prior to large-scale VLAN deployment. VLANs have been heavily promoted by the main enterprise equipment vendors (such as Cisco, Cabletron, and 3Com). They are more commonly implemented in larger enterprises (such as financial institutions). However, as router technology improves in performance (approaching wire speed forwarding rates) and switches begin to incorporate more and more Layer 3 functionality, fewer organizations will need to implement large, flat networks, and, consequently, VLANs are likely to be less attractive for the network designer in the future.

9.8 Summary

In this chapter, we discussed the following:

- Repeaters are Physical Layer devices, which simply forward frames over extended LANs and offer no traffic management capabilities. Bridges and switches use Layer 2 MAC addressing to make forwarding decisions and in so doing perform a limited form of traffic management. A number of different switching models have emerged,

including classic store-and-forward switching, cut-through switching, adaptive switching, and fragment-free switching.

- Since neither bridges nor switches are aware of Network Layer addressing, there is no concept of network hierarchy, and as such networks designed with these devices are not scalable and are limited in their ability to engineer traffic efficiently. Their use is primarily for extended local area networks and simple remote office interconnects. A new logical network architecture called the Virtual LAN (VLAN) has emerged in recent years, which enables hierarchy in bridged or switched networks.
- The Spanning Tree algorithm is implemented in bridges to provide automated loop avoidance. In practice it is slow to converge and potentially expensive to deploy, especially in wide area networks.
- The IBM 8209 translation bridge is a hybrid device, not fully covered by the standards, which enables Ethernet and Token Ring users to coexist. It uses brute-force techniques to convert frame formats from one media type to another. Typically there are protocols that are unsupported for bridging across these devices.
- Encapsulation bridging for Ethernet-to-FDDI applications is simple but limited to Ethernet-to-Ethernet communications only, with the FDDI ring used merely for transit. The use of proprietary protocols between bridges means that this form of bridging is now rarely used. The preferred method is translation bridging, which enables both FDDI and Ethernet/Token Ring stations to communicate directly.
- VLANs allow the formation of virtual workgroups, better security, improved performance, simplified administration, and reduced costs. VLANs are formed by the logical segmentation of a network and can be classified into Layers 1, 2, 3, and higher layers. Only Layers 1 and 2 are specified in IEEE 802.1Q. Tagging and the filtering database allow a bridge to determine the source and destination VLAN for received data.
- VLANs, if implemented effectively, offer an effective and transparent mechanism to scale flat networks. However, as router technology improves in performance (approaching wire speed forwarding rates) and switches begin to incorporate more and more Layer 3 functionality, fewer organizations will need to implement large, flat networks, and, consequently, VLANs are likely to be less attractive for the network designer in the future.

References

[1] R. Perlman, *Interconnections, Bridges, and Routers* (Reading, MA: Addison-Wesley, 1993).

[2] Standards for the Transmission of IP Datagrams over IEEE 802 Networks, RFC 1042 (February 1988).

[3] A. Latif, E. J. Rowlance, and R. H. Adams, "The IBM 8209 LAN Bridge," *IEEE Network* (May 1992).

[4] IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks, IEEE Std 802.1Q-1998 (1998).

[5] Cabletron's VLAN Hello Protocol Specification, version 4, RFC 2641 (August 1999).

[6] Cabletron's SecureFast VLAN Operational Model, RFC 2643 (August 1999).

[7] ISO/IEC 15802-3 Information Technology, Telecommunications and Information Exchange between Systems, Local and Metropolitan Area Networks, Specific Requirements, Supplement to Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications, Frame Extensions for Virtual Bridged Local Area Network (VLAN) Tagging on 802.3 Networks, IEEE Std 802.3ac-1998 (supplement to IEEE 802.3 1998 edition) (1998).