CHAPTER 21

Configuring Source-Route Bridging

Our bridging software includes Source-Route Bridging (SRB) capability. A source-route bridge connects multiple physical Token Rings into one logical network segment. When the network segment bridges only Token Ring media to provide connectivity, it is called source-route bridging. When the network bridges Token Ring, and some sort of non-Token Ring media is introduced into the bridged network segment, it is called remote source-route bridging.

The source-route bridging feature enables our router/bridge to simultaneously act as a Level 3 router and a Level 2 source-route bridge. Thus, protocols such as IPX or XNS can be routed on Token Rings, while other protocols such as SNA or NetBIOS are source-route bridged.

For a complete description of the commands mentioned in this chapter, refer to the "Source-Route Bridging Commands" chapter of the *Router Products Command Reference* publication. For historical background and a technical overview of source-route bridging, see the *Internetworking Technology Overview* publication.

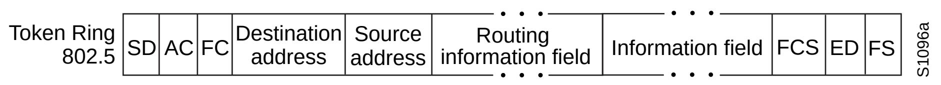
Source-Route Bridging Overview

Source-route bridging technology is a combination of bridging and routing functions. A source-route bridge is allowed to make routing decisions based upon the contents of the Media Access Control (MAC) frame header. Keeping the routing function at the MAC or Level 2 layer allows the higher-layer protocols to execute their tasks more efficiently and allows the LAN to be expanded without the knowledge of the higher-layer protocols.

As designed by IBM and the IEEE 802.5 committee, source-route bridges connect extended Token Ring LANs. A source-route bridge uses the Routing Information Field (RIF) in the IEEE 802.5 MAC header of a datagram (see Figure 1-1) to determine which rings or Token Ring network segments the packet must transit. The source station inserts the RIF into the MAC header immediately following the source address field in every frame, giving this style of bridging its name. The destination station reverses the routing field to reach the originating station.

The information in a RIF is derived from explorer packets generated by the source node. These explorer packets traverse the entire source-route bridge network, gathering information on the possible paths the source node might use to send packets to the destination.

Figure 1-1 IEEE 802.5 Token Ring Frame Format



Unlike transparent spanning-tree bridging, which requires time to recompute topology in the event of failures, source-route bridging allows multiple, active paths through the network, which provides for more timely switches to alternate routes in the event of failure. Most importantly, source-route bridging places the burden of transmitting frames with the end stations by allowing them to determine the routes the frames take.

Cisco's Implementation of Source-Route Bridging

Cisco's source-route bridging software implementation includes the following features:

- Provides configurable fast-switching software for source-route bridging.
- Provides for a local source-route bridge that connects two or more Token Ring networks.
- Provides *ring groups* to configure a source-route bridge with more than two network interfaces. A ring group is a collection of Token Ring interfaces in one or more routers that are collectively treated as a *virtual ring*.
- Provides two types of explorer packets to collect Routing Information Field (RIF) information. An *all-routes* explorer packet, which follows all possible paths to a destination ring, and a *spanning-tree* explorer packet, which follows a statically configured limited route (spanning tree) when looking for paths.
- For remote source-route bridging (RSRB), provides for multiple router/bridges separated by non-Token Ring segments. Three options are available:
  - — Encapsulate the Token Ring traffic inside IP datagrams passed over a TCP connection between two router/bridges.
  - — Use Fast Sequenced Transport (FST) to transport remote source-route bridging packets to their peers without TCP or UDP header or processor overhead.
  - — Use MAC-layer encapsulations over a single serial line, Ethernet, Token Ring, or FDDI ring connected between two routers attached to Token Ring networks.
- Provides for configurable limits to the size of the TCP backup queue.
- Provides a dynamically determined RIF cache based on the protocol; also allows you to manually add entries to the RIF cache.
- Provides for filtering by MAC address, Link Service Access Point (LSAP) header, and protocol type.
- Provides for filtering of NetBIOS frames either by station name or by a packet byte offset.
- Provides for translation into transparently bridged frames to allow source-route stations to communicate with nonsource-route stations (typically on Ethernet).
- Provides support for the SRB MIB variables as described in the IETF draft "Bridge MIB" document, "Definition of Managed Objects for Bridges," by E. Decker, P. Langille, A. Rijsinghani, and K. McCloghrie, June 1991. Only the SRB component of the Bridge MIB is supported.
- Provides support for the Token Ring MIB variables as described in RFC 1231, "IEEE 802.5 Token Ring MIB," by K. McCloghrie, R. Fox, and E. Decker, May 1991. Cisco implements the mandatory tables (Interface Table and Statistics Table) but not the optional table (Timer Table) of the Token Ring MIB. The Token Ring MIB has been implemented for the 4/16-Mb Token Ring cards that can be user adjusted for either 4- or 16-Mb transmission speeds (CSC-1R, CSC-2R, CSC-R16M, or CSC-C2CTR).

SRB Configuration Task List

You can perform the following tasks to configure source-route bridging.

- Configure Source-Route Bridging
- Configure Remote Source-Route Bridging
- Configure Bridging Routed Protocols
- Configure Translation between SRB and TB Environments
- Configure NetBIOS Support
- Configure LAN Network Manager Support
- Secure the SRB Network
- Tune the SRB Network
- Establish SRB Interoperability with Specific Token Ring Implementations
- Monitor and Maintain the SRB Network

This chapter describes how to perform these tasks. At the end of the chapter are source-route bridging configuration examples.

Configure Source-Route Bridging

A source-route bridge connects two or more Token Ring networks using only Token Ring media. This means that no non-Token Ring media have been introduced into the networks over which you will be bridging.

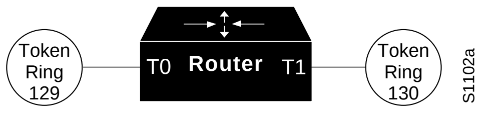
When a router is configured as a source-route bridge, bridged traffic does not pass across non-Token Ring media, and only those protocols that are not being routed are source-route bridged. For example, if IPX routing is enabled on the router that is configured for source-route bridging, IPX datagrams will not be source-route bridged. However, datagrams for other nonrouted protocols will be source-route bridged.

You can configure the router for source-route bridging by performing any of the first three tasks and optionally, one or both of the last two tasks:

- Configure a dual-port bridge
- Configure multiple dual-port bridge
- Configure a multiport bridge using a virtual ring
- Enable forwarding and blocking of spanning-tree explorers
- Limit the maximum SRB hops

A dual-port bridge is the simplest possible source-route bridging configuration. When configured as a dual-port bridge, the router serves to connect two Token Ring LANs. One LAN is connected through one port (Token Ring interface), and the other LAN is connected through the other port (also a Token Ring interface). Figure 1-2 shows a dual-port bridge.

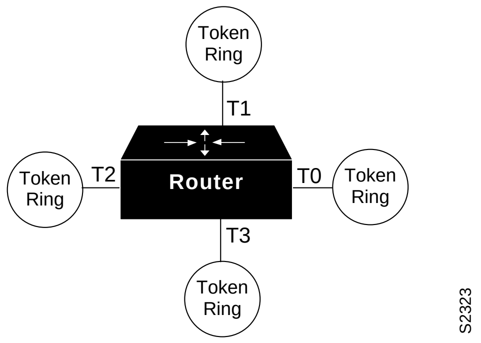
Figure 1-2 Dual-Port Bridge



A dual-port bridge is a limitation imposed by IBM Token Ring chips; they can only process two ring numbers. If you have a router with two or more Token Ring interfaces, you can work around the two-ring number limitation. You can configure your router as multiple dual-port bridges or as a multiport bridge using a virtual ring.

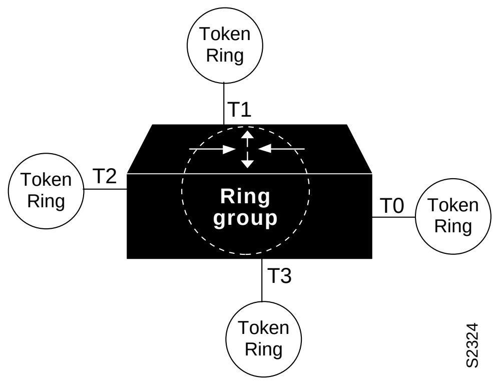
You can define several separate dual-port bridges in the same router. However, the devices on the LANs cannot have any-to-any connectivity; that is, they cannot connect to every other device on the bridged LANs. Only the devices connected to the dual-port bridge can communicate with one another. Figure 1-3 shows two separate dual-port bridges (T0-T2 and T1-T3) configured on the same router.

Figure 1-3 Multiple Dual-Port Bridges



A better solution for overcoming the two-ring number limitation of IBM Token Ring chips is to configure a multiport bridge using a virtual ring. A virtual ring on a multiport bridge allows the router to interconnect three or more LANs with any-to-any connectivity; that is, connectivity between any of the devices on each of the three LANs is allowed. A virtual ring creates a logical Token Ring internal to the router that causes all the Token Rings connected to the router to be treated as if they are all on the same Token Ring. The virtual ring is called a *ring group.*

Figure 1-4 Multiport Bridge using a Virtual Ring



To take advantage of this virtual ring feature, each Token Ring interface on the router must be configured to belong to the same ring group. For information about configuring a multiport bridge using a virtual ring, see the "Configure a Multiport Bridge using a Virtual Ring" section later in this chapter.

Configure a Dual-Port Bridge

The router's Token Ring software is written such that usually, minimal configuration is necessary. To configure a dual-port bridge, you must perform the first task and optionally one or both of the other tasks:

- Enable SRB on the appropriate Token Ring interface
- Enable the forwarding of spanning-tree explorers (strongly recommended)
- Limit the maximum number of source-route bridge hops

For multiple dual-port source-route bridges, you would repeat this task for each Token Ring interface that is part of a dual-port bridge. If you wanted your network to use only source-route bridging, you could connect as many of these routers via Token Rings as you needed. Remember, to use source-route bridging requires you bridge only Token Ring media.

Enable SRB on the Appropriate Token Ring Interface

A router equipped with Token Ring cards is by default a Token Ring host, and source-route bridging is disabled by default. To configure a dual-port bridge that connects two Token Rings, you must enable source-route bridging on each of the Token Ring interfaces that connect to the two Token Rings. To enable source-route bridging, perform the following task in interface configuration mode for each of the Token Ring interfaces:

Task	Command
Enable local source-route-bridging on a Token Ring interface.	**source-bridge** *local-ring bridge-number target-ring*

Note Ring numbers need to be unique across interfaces and networks, so that when you enable source-route bridging over an interface, the local and target rings are defined. Each node on the network will know if it is the target of explorer packets sent on the network.

Configure a Multiport Bridge using a Virtual Ring

To configure a source-route bridge to have more than two network interfaces, you must perform the following tasks in the specified order:

- Define a *ring group*
- Enable source-route-bridging and assign a ring group to a Token Ring interface

Once you have completed these tasks, the router acts as a multiport bridge not as a dual-port bridge.

Note Ring numbers need to be unique across interfaces and networks.

Define a Ring Group in SRB Context

Because all IBM Token Ring chips can only process two ring numbers, we have implemented the concept of a ring group or virtual ring. A ring group is a collection of Token Ring interfaces in one or more routers that share the same ring number. This ring number is used just like a physical ring number, showing up in any route descriptors contained in packets being bridged. Within the context of a multiport bridge that uses source-route bridging rather than remote source-route bridging (RSRB), the ring group resides in the same router. See the "Configure Remote Source-Route Bridging" section to compare ring groups in the SRB and RSRB context.

A ring group must be assigned a ring number that is unique throughout the network. It is possible to assign different Token Ring interfaces on the same router to different ring groups, if, for example, you plan to administer them as interfaces in separate domains.

To define or remove a ring group, perform one of the following tasks in global configuration mode:

Task	Command
Define a ring group.	**source-bridge ring-group** *ring-number*
Remove a ring group.	**no source-bridge ring-group** *ring-number*

Enable SRB and Assign a Ring Group to an Interface

After you have defined a ring group, you must assign that ring group to those interfaces you plan to include in that ring group. An interface can only be assigned to one ring group. To enable any-to-any connectivity among the end stations connected through this multiport bridge, you must assign the same target ring number to all Token Ring interfaces on the router.

To enable SRB and assign a ring group to an interface, perform the following task in interface configuration mode:

Task	Command
Enable source-route-bridging and assign a ring group to a Token Ring interface.	**source-bridge** *local-ring bridge-number target-ring*

Enable the Forwarding and Blocking of Spanning Tree Explorers

When trying to determine the location of remote destinations on a source-route bridge, the source device will need to send explorer packets. Explorer packets are used to collect routing information field (RIF) information. The source device can send spanning-tree explorers or all-routes explorers. Note that some older IBM devices only generate all-routes explorer packets, but many newer IBM devices are capable of generating spanning-tree explorer packets.

A spanning-tree explorer packet is an explorer packet that is sent to a defined group of nodes that comprise a statically configured spanning tree in the network. In contrast, an all-routes explorer packet is an explorer packet that is sent to every node in the network on every path.

Forwarding all-routes explorer packets is the default. However, in complicated source-route bridging topologies, using this default can generate an exponentially large number of explorers that are traversing the network. The number of explorer packets becomes quite large because duplicate explorer packets are sent across the network to every node on every path. Eventually each explorer packet will reach the destination device. The destination device will respond to each of these explorer packets. It is from these responses that the source device will collect the RIF and determine which route it will use to communicate with the destination device. Usually, the route contained in the first returned response will be used.

The number of explorer packets traversing the network can be reduced by sending spanning-tree explorer packets. Spanning-tree explorer packets are sent to specific nodes; that is, to only the nodes on the spanning tree, not to all nodes in the network. You must manually configure the spanning-tree topology over which the spanning-tree explorers are sent. You do this by configuring which interfaces on the routers will forward spanning-tree explorers and which interfaces will block them.

To enable forwarding of spanning-tree explorers on an outgoing interface, perform the following task in interface configuration mode:

Task	Command
Enable the forwarding of spanning-tree explorer packets on an interface.	**source-bridge spanning**

Note While enabling the forwarding of spanning-tree explorer packets is not an absolute requirement, it is strongly recommended in complex topologies. Configuring an interface to block or forward spanning-tree explorers has no effect on how that interface handles all-routes explorer packets. All-routes explorers can always traverse the network.

To block forwarding of spanning tree explorers on an outgoing interface, perform the following task in interface configuration mode:

Task	Command
Block spanning-tree explorer packets on an interface.	**no source-bridge spanning**

Limit the Maximum SRB Hops

You can minimize explorer storms if you limit the maximum number of source-route bridge hops. For example, if the largest number of hops in the best route between two end stations is six, it might be appropriate to limit the maximum source-route bridging hops to six as one way to eliminate unnecessary traffic. This setting affects spanning-tree explorers and all-routes explorers sent from source devices.

To limit the number of SRB hops, perform the following task in interface configuration mode:

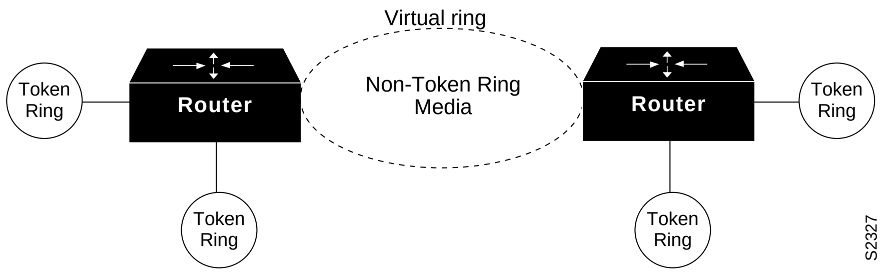
Task	Command
Limit the maximum number of source-route bridge hops.	**source-bridge max-hops** *count*

Configure Remote Source-Route Bridging

While source-route bridging involves bridging between Token Ring media only, remote source-route bridging (RSRB) involves multiple router/bridges separated by non-Token Ring network segments. Figure 1-5 shows an RSRB topology. The virtual ring can extend across any non-Token Ring media supported by the RSRB such as serial, Ethernet, FDDI, and WANs. The type of media you select will determine the way you set up RSRB.

Note If you will be bridging across Token Ring media, it is recommended that you do not use RSRB. Use SRB instead.

Figure 1-5 Remote Source-Route Bridged Topology



There are four ways to set up RSRB as follows:

- Direct encapsulation. This method uses an HDLC-like encapsulation to pass frames over a single physical network connection between two routers attached to Token Rings. Use this method when you are running source-route bridge traffic over point-to-point, single-hop, serial or LAN media. Although this method does not have the flexibility of the TCP approach, it provides the best performance of the three methods because it involves less overhead.
- IP encapsulation over a Fast Sequenced Transport (FST) connection. This method involves encapsulating the source-route bridged traffic inside IP datagrams passed over a Fast Sequenced Transport (FST) connection between two router/bridges. While not as fast as direct encapsulation, it outperforms IP encapsulation over a TCP connection because it has lower overhead. However, this method does not allow for local acknowledgment, nor is it suitable for use in networks that tend to reorder frame sequences.
- IP encapsulation over a TCP connection. This method involves encapsulating the source-route bridged traffic inside IP datagrams passed over a TCP connection between two router/bridges. While this method offers lower performance than the other two methods, it is the appropriate method to use under the following conditions:
    - — You plan to connect Token Ring networks across arbitrary media including Ethernets, FDDI, serial interfaces, X.25 networks, and so forth.
    - — You plan to connect Token Ring networks across a multiprotocol backbone network.
    - — You plan to load share over multiple, redundant paths. Using this topology, when a path fails there is no need for hosts to retransmit explorer packets. IP routing handles the network reconfiguration transparently to the Token Ring hosts.
- IP encapsulation over a TCP connection with Local Acknowledgment. This method involves encapsulating the source-route bridged traffic inside IP datagrams passed over a TCP connection between two router/bridges with Local Acknowledgment enabled. This method is appropriate when you have LANs separated by wide geographic distances and you want to avoid time delays, multiple retransmissions, or loss of user sessions.

Note IP encapsulation over a TCP connection should only be used within complex meshed networks to support connections between peers that are separated by multiple hops and have the potential of using multiple paths and where performance is not an issue. Direct encapsulation should be used in point-to-point connections. In such point-to-point configuration, using TCP would add too much unnecessary processing overhead.

Configure RSRB Using Direct Encapsulation

To configure a remote source-route bridge to use a point-to-point serial line or a single Ethernet, or a single FDDI hop, you must perform the following tasks:

- Define a ring group
- Identify the remote peers
- Enable source-route bridging on the appropriate interfaces

For an example of how to configure RSRB using direct encapsulation, see the “Examples of RSRB Using Direct Encapsulation” section later in this chapter.

Define a Ring Group in RSRB Context

In our implementation of RSRB, whenever you connect Token Rings using non-Token Ring media, you must treat that non-Token Ring media as a virtual ring by assigning it to a ring group. Every router/bridge with which you wish to exchange Token Ring traffic must be a member of this same ring group. These other router/bridges are referred to as remote peer bridges. The ring group is therefore made up of interfaces that reside on separate routers.

A ring group must be assigned a ring number that is unique throughout the network. It is possible to assign different interfaces on the same router to different ring groups, if, for example, you plan to administer them as interfaces in separate domains.

To define or remove a ring group, perform one of the following tasks in global configuration mode:

Task	Command
Define a ring group.	**source-bridge ring-group** *ring-number*
Remove a ring group.	**no source-bridge ring-group** *ring-number*

Identify the Remote Peers (Direct Encapsulation)

The interfaces that you identify as remote peer bridges must be serial, Ethernet, FDDI, or Token Ring interfaces. On a serial interface, you must use HDLC encapsulation. To identify remote-peer bridges, perform the following task in global configuration mode:

Task	Command
Define the ring-group and identify the interface over which to send source-route bridging traffic to another router/ bridge in the ring group.	**source-bridge remote-peer** *ring-group* **interface** *interface-name* [*mac-address*]

You must specify one **source-bridge remote-peer** command for each peer router that is part of the virtual ring. You must also specify one **source-bridge remote-peer** command to identify the IP address of the local router.

Note You use one instance of the **source bridge remote peer** command for each interface you configure for remote source-route bridging. If you use the *mac-address* argument, be sure that it is the MAC address on the remote interface that is directly connected to the system that is being configured. It should not be the MAC address of the Token Ring interface on the remote peer.

Enable SRB on the Appropriate Interfaces

You must enable source-route bridging on each of the interfaces through which source-route bridging traffic will pass. The value of the target ring parameter for the **source-bridge** command should be the ring group number you have assigned to the interface. To enable SRB on an interface, perform the following task in interface configuration mode:

Task	Command
Enable source-route bridging on an interface.	**source-bridge** *local-ring bridge-number target-ring*

Set the Keepalive Interval of the Remote Source-Route Bridging Remote Peer

To set the keepalive interval of the remote source-route bridging remote peer, perform the following task in interface configuration mode:

Task	Command
Set the keepalive interval.	**source-bridge keepalive** *seconds*

Configure RSRB Using IP Encapsulation over an FST Connection

To configure a remote source-route bridge to use IP encapsulation over an FST connection, you must perform the following tasks:

- Set up an FST peer name and assign an IP address.
- Identify the remote peers.
- Enable SRB on the appropriate interfaces.

Note FST encapsulation preserves the dynamic media-independent nature of IP routing to support SNA and NetBIOS applications.

For an example of how to configure RSRB over an FST connection, see the "Example of RSRB Using IP Encapsulation over an FST Connection" section later in this chapter.

Set Up an FST Peer Name and Assign an IP Address

To set up an FST peer name and provide an IP address to the local router, perform the following task in global configuration mode:

Task	Command
Set up an FST peer name and provide the local router with an IP address.	**source-bridge fst-peername** *local-interface-address* [**lf** *size*] [**version** *number*]

In our implementation of RSRB, whenever you connect Token Rings using non-Token Ring media, you must treat that non-Token Ring media as a virtual ring by assigning it to a ring group. Every router/bridge with which you wish to exchange Token Ring traffic must be a member of this same ring group. Therefore, after you set up an FST peer name, define a ring group. For more information about defining a ring group, see the "Define a Ring Group in the RSRB Context" section earlier in this chapter.

Identify the Remote Peer s (FST Connection)

All of the router/bridges with which you want to exchange Token Ring traffic are referred to as remote peer bridges. The remote peers can be at the other end of an FST connection. To identify the remote peers, perform the following task in global configuration mode:

Task	Command
Identify your peers and specify an FST connection.	**source-bridge remote-peer** *ring-group* **fst** *ip-address*

You must specify one **source-bridge remote-peer** command for each peer router that is part of the virtual ring. You must also specify one **source-bridge remote-peer** command to identify the IP address of the local router.

Note You must use one instance of the **source bridge remote peer** command for each interface you configure for remote source-route bridging. The *ip-address* argument should be the IP address the router tries to reach.

Enable SRB on the Appropriate Interfaces

You must enable source-route bridging on each of the interfaces through which source-route bridging traffic will pass. The value of the target ring parameter for the **source-bridge** command should be the ring group number you have assigned to the interface. To enable source-route bridging, perform the following task in interface configuration mode:

Task	Command
Enable local source-route-bridging on a Token Ring interface.	**source-bridge** *local-ring bridge-number target-ring*

Performance Considerations when Using FST in a Redundant Network Topology

FST is fast switched when it receives or sends frames from Ethernet, Token Ring or FDDI interfaces. It is also fast switched when sending and receiving from serial interfaces configured with the HDLC encapsulation. In all other cases, FST is slow switched.

In cases where FST is fast switched, in either the routers configured for FST or in the routers contained within the IP “cloud” between a pair of FST peers, only one path will be used at a given time between the two FST peers. This provides an extremely high likelihood that frames will not arrive at one peer in a different sequence than are sent from a remote peer. In the very rare cases where this can happen, the FST code on the receiving peer will discard the out-of-order frame. As such, the Token Ring end hosts will rarely lose a frame over the FST router cloud, and performance levels will remain adequate.

The same conditions hold true for any slow-switched topology that provides only a single path between the peers. For example, a single X.25 network cloud would fall under this category. Similarly, if two slow-switched paths are of very different costs such that one always will be chosen over the other, the chances of having frames received out of sequence are also rare.

However, if two or more slow-switched paths of equal cost exist between the two routers (such as two parallel X.25 networks), the routers alternate in sending packets between the two or more equal-cost paths. This results in a high probability of frames arriving out of sequence at the receiver. In such cases, the FST code will dispose of every out-of-sequence packet, leading to a large number of drops at the router. This requires that the end hosts retransmit frames, greatly reducing overall throughput.

When such parallel paths exist, it is strongly recommended that one be chosen over the other as the preferred path. This can be done by specifying a higher bandwidth for the path that contains the direct connections to the two or more parallel paths on the router.

Do not use FST when the probability routinely exists for frames to lose their ordering in your network. If you have a network where frames are routinely reordered, it is far better to use the TCP protocol for remote source-route bridging, because it provides the overhead necessary to bring frames back in order on the receiving router. FST, to remain fast, does not provide for such a mechanism, and will toss out-of-order frames.

Configure RSRB Using IP Encapsulation over a TCP Connection

To configure a remote source-route bridge to use IP encapsulation over a TCP connection, you must perform the following tasks:

- Identify the remote peers.
- Enable SRB on the appropriate interfaces.

Identify the Remote Peer (TCP Connection)

In our implementation, whenever you connect Token Rings using non-Token Ring media, you must treat that non-Token Ring media as a virtual ring by assigning it to a ring group. Every router/bridge with which you wish to exchange Token Ring traffic must be a member of this same ring group. For more information about defining a ring group, see the "Define a Ring Group in the RSRB Context" section earlier in this chapter.

To identify the remote peers, perform the following task in global configuration mode:

Task	Command
Identify the IP address of a peer in the ring group with which to exchange source-bridge traffic using TCP.	**source-bridge remote-peer** *ring-group* **tcp** *ip-address*

You must specify one **source-bridge remote-peer** command for each peer router that is part of the virtual ring. You must also specify one **source-bridge remote-peer** command to identify the IP address of the local router.

Note You must use one instance of the **source bridge remote peer** command for each interface you configure for remote source-route bridging. The *ip-address* argument is the IP address the router tries to reach.

Enable SRB on the Appropriate Interfaces

The value of the target ring parameter for the **source-bridge** command should be the ring group number.

To enable SRB on an interface, perform the following task in interface configuration mode:

Task	Command
Enable local source-route-bridging on a Token Ring interface.	**source-bridge** *local-ring bridge-number target-ring*

Configure RSRB Using TCP and LLC2 Local Acknowledgment

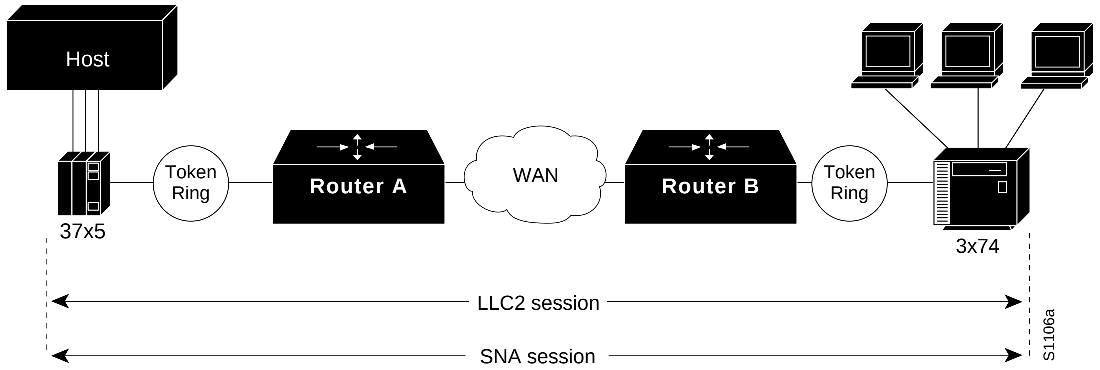
If you configure your remote source-route bridge to use IP encapsulation over a TCP connection, you also can enable LLC2 Local Acknowledgment. The following explains how this feature can be useful.

LLC2, or Logical Link Control–Type 2, an ISO standard data link level protocol used in Token Ring networks, was designed to ensure reliable transmission of data across LAN media with minimal or predictable time delays. With the advent of remote source-route bridging (RSRB) and wide area network (WAN) backbones, LANs are now separated by wide, geographic distances spanning countries and continents. As a result, these LANs have time delays that are longer than LLC2 allows for bidirectional communication between hosts. The Local Acknowledgment capability in router/bridges supporting remote source-route bridging addresses the problem of unpredictable time delays, multiple retransmissions, and loss of user sessions.

In a typical LLC2 session, when host A sends a frame to host B, the sending host A expects host B to respond positively or negatively in a certain amount of predefined time commonly called the *T1 time*. If host A does not receive an acknowledgment of the frame it sent to host B within the T1 time, it will retry a few number of times (normally 8 to 10). If there is still no response from host B, host A will drop the session.

Figure 1-6 illustrates an LLC2 session. A 37x5 on a LAN segment can communicate with a 3x74 on a different LAN segment separated via a wide area backbone network. Frames are transported between Router A and Router B using remote source-route bridging. However, the LLC2 session between the 37x5 and the 3x74 is still end-to-end; that is, every frame generated by the 37x5 traverses the backbone network to the 3x74, and the 3x74, on receipt of the frame, acknowledges it.

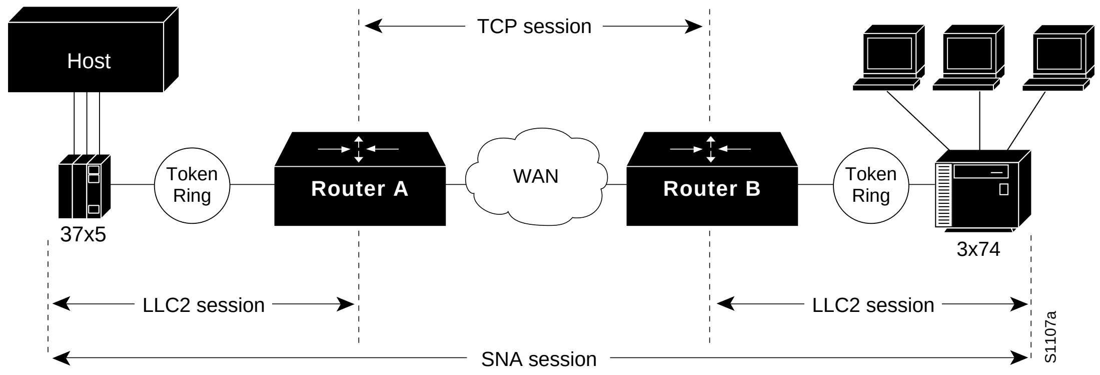
Figure 1-6 LLC2 Session Without Local Acknowledgment



On backbone networks consisting of slow serial links, the T1 timer on end hosts could expire before the frames have a chance to reach the remote hosts, causing the end host to retransmit. This results in duplicate frames reaching the remote host at the same time as the first frame also reached the remote host, albeit slowly. These frame duplications break the LLC2 protocol, resulting in the loss of sessions between the two IBM machines.

One way to solve this time delay problem is to increase the timeout value on the end nodes to account for the maximum transit time between the two end machines. However, in networks consisting of hundreds or even thousands of nodes, every machine would need to be reconfigured with new values. With Local Acknowledgment for LLC2 turned on, the LLC2 session between the two end nodes would not be end-to-end, but instead, terminates at two local routers. Figure 1-7 shows the LLC2 session with the 37x5 ending at Router A and the LLC2 session with the 3x74 ending at Router B. Both Router A and Router B execute the full LLC2 protocol as part of Local Acknowledgment for LLC2.

Figure 1-7 LLC2 Session with Local Acknowledgment



With Local Acknowledgment for LLC2 enabled in both routers, Router A acknowledges frames received from the 37x5. The 37x5 still thinks that the acknowledgments it receives are from the 3x74. Router A looks like the 3x74 to the 37x5. Similarly, Router B acknowledges frames received from the 3x74. The 3x74 thinks that the acknowledgments it receives are from the 37x5. Router B looks like the 3x74 to 37x5. Because the frames no longer have to travel the WAN backbone networks to be acknowledged, but instead are locally acknowledged by routers, the end machines do not time out, resulting in no loss of sessions.

The advantages of enabling Local Acknowledgment for LLC2 include the following:

- Local Acknowledgment for LLC2 solves the T1 timer problem without having to change any configuration on the end nodes. The end nodes are unaware that the sessions are being locally acknowledged. In networks consisting of hundreds or even thousands of machines, this is a definite advantage. All the frames acknowledged by the router appear to the end hosts to be coming from the remote IBM machine. In fact, by looking at a trace from a protocol analyzer, one cannot say whether a frame was acknowledged by the local router or by a remote IBM machine. The MAC addresses and the RIFs generated by the routers are identical to those generated by the remote IBM machine. The only way to find out whether a session is locally acknowledged is to use either a **show local-ack** command or a **show source-bridge** command on the router.
- All the supervisory (RR, RNR, REJ) frames that are locally acknowledged go no farther than the router. Without Local Acknowledgment for LLC2, *every* frame traverses the backbone. With Local Acknowledgment, only data (I-frames) traverse the backbone, resulting in less traffic on the backbone network. For installations in which customers pay for the amount of traffic passing through the backbone, this could be a definite cost-saving measure. A simple protocol exists between the two *peers* to bring up or down a TCP session.

To configure a remote source-route bridge to use IP encapsulation over a TCP connection, you must perform the following tasks:

- Enable LLC2 Local Acknowledgment between two remote peers
- Enable SRB on the appropriate interfaces

Enable LLC2 Local Acknowledgment between Two Remote Peer Bridges

In our implementation, whenever you connect Token Rings using non-Token Ring media, you must treat that non-Token Ring media as a virtual ring by assigning it to a ring group. Every router/bridge with which you wish to exchange Token Ring traffic must be a member of this same ring group. For more information about defining a ring group, see the "Define a Ring Group in the RSRB Context" section earlier in this chapter.

To enable LLC2 Local Acknowledgment, perform the following task in global configuration mode:

Task	Command
Enable LLC2 Local Acknowledgment on a per-remote-peer basis.	**source-bridge remote-peer** *ring-group* **tcp** *ip-address* [**local-ack**]

The **local-ack** keyword specifies that LLC2 sessions destined to a specific remote peer are to be locally acknowledged. You must use one instance of the **source bridge remote peer** command for each interface you configure for remote source-route bridging.

Enable SRB on the Appropriate Interfaces

The value of the target ring parameter for the **source-bridge** command should be the ring group number.

To enable SRB on an interface, perform the following task in interface configuration mode:

Task	Command
Enable local source-route-bridging on a Token Ring interface.	**source-bridge** *local-ring bridge-number target-ring*

For an example of how to configure RSRB with Local Acknowledgment, see the "Example of RSRB with Local Acknowledgment" section later in this chapter.

Notes on Using LLC2 Local Acknowledgment

LLC2 Local Acknowledgment can only be enabled with TCP remote peers (as opposed to LAN or direct serial interface remote peers), because the routers need the reliable transmission of TCP to provide the same reliability that an LLC2 LAN end-to-end connection provides. Therefore, the direct media encapsulation options for the **source-bridge remote-peer** command cannot be used.

If the LLC2 session between the local host and the router terminates in either router, the other will be informed to terminate its connection to its local host.

If the TCP queue length of the connection between the two routers reaches 90 percent of its limit, the routers will send Receiver-not-Ready (RNR) messages to the local hosts until the queue limit is reduced to below this limit.

The configuration of the LLC2 parameters for the local Token Ring interfaces can affect overall performance. Refer to the "Configuring LLC2 and SDLC Parameters" chapter of this manual for more details about fine-tuning your network through the LLC2 parameters.

Note As previously stated, Local Acknowledgment for LLC2 is meant only for extreme cases in which communication is not possible otherwise. Because the router must maintain a full LLC2 session, the number of simultaneous sessions it can support before performance degrades depends on the mix of other protocols and their loads also running in it.

The routers at each end of the LLC2 session execute the full LLC2 protocol, which could result in some overhead. The decision to turn on Local Acknowledgment for LLC2 should be based on the speed of the backbone network in relation to the Token Ring speed. For LAN segments separated by slow-speed serial links (for example, 56 kbps), the T1 timer problem could occur more frequently. In such cases, it may be wise to turn on Local Acknowledgment for LLC2. For LAN segments separated by a FDDI backbone, backbone delays will be minimal; in such cases, Local Acknowledgment for LLC2 should not be turned on. Speed mismatch between the LAN segments and the backbone network is one criterion to be used in the decision to use Local Acknowledgment for LLC2.

There are some situations (such as host B dying between the time host A sends data and the time host B receives it), when host A would believe, *at the LLC2 layer*, that data was received that actually was not, because the router acknowledges that it received data from host A before it knows that host B can actually receive the data. But because both NetBIOS and SNA have error recovery in situations where an end device goes down, these higher-level protocols will resend any missing or lost data. These transaction request/confirmation protocols exist above LLC2, so they are not affected by tight timers, as is LLC2. They also are transparent to Local Acknowledgment.

If you are using NetBIOS applications, note that there are two NetBIOS timers—one at the link level and one at the next higher level. Local Acknowledgment for LLC2 is designed to solve session timeouts at the link level only. If you are experiencing NetBIOS session timeouts, you have two options:

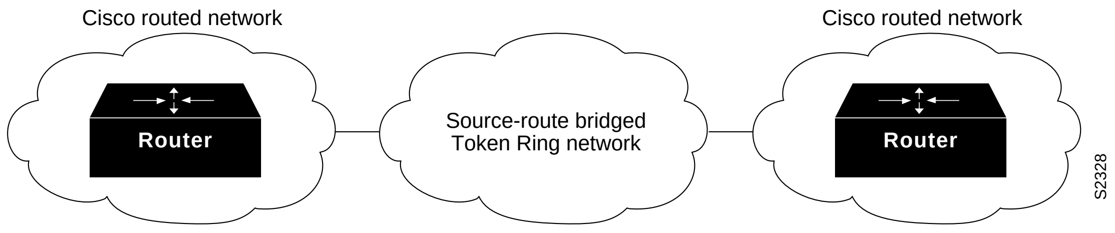
- Experiment with increasing your NetBIOS timers.
- Avoid using NetBIOS applications on slow serial lines.

Configure Bridging Routed Protocols

Source-route bridges use MAC information, specifically the information contained in the Routing Information Field (RIF), to bridge packets. A RIF contains a series of ring and bridge numbers that represent the possible paths the source node might use to send packets to the destination. Each ring number in the RIF represents a single Token Ring in the source-route bridged (SRB) network and is designated by a unique 12-bit ring number. Each bridge number represents a bridge that is between two Token Rings in the SRB network and is designated by a unique 4-bit bridge number. The information in a RIF is derived from explorer packets traversing the source-route bridged network. Without the RIF information, a packet could not be bridged across a source-route bridged network. For more information about RIFs and their format, refer to the *Internetworking Technology Overview* publication.

Unlike source-route bridges, Level 3 routers use protocol-specific information (for example Novell IPX or XNS headers) rather than MAC information to route datagrams. As a result, the router software default for routed protocols is to not collect RIF information and to not be able to bridge routed protocols. However, if you want the router to bridge routed protocols across a source-route bridged network, the router must be able to collect and use RIF information to bridge packets across a source-route bridged network. You can configure the router to append RIF information to routed protocols so that routed protocols can be bridged. Figure 1-8 shows a network topology in which you would want to use this feature.

Figure 1-8 Topology for Bridging Routed Protocols across a Source-Route Bridged Network



To configure the router to bridge routed protocols, you must perform the first task, and optionally, one or both of the other tasks as follows:

- Enable use of the RIF
- Configure a static RIF entry
- Configure the RIF timeout interval

Enable Use of the RIF

You can configure the router so that it will append RIF information to the routed protocols. This allows routed protocols to be bridged across a source-route bridged network. The routed protocols that you can bridge are as follows:

- Apollo Domain
- AppleTalk
- ISO CLNS
- DECnet
- IP
- IPX
- VINES
- XNS

You would enable use of the RIF only on Token Ring interfaces on the router.

To configure the router to append RIF information, perform the following task in interface configuration mode:

Task	Command
Enable collection and use of RIF information.	**multiring** {*protocol-keyword* | **all** | **other**}

For an example of how to configure the router to bridge routed protocols, see the “Example of Bridging Routed Protocols” section later in this chapter.

Configure a Static RIF Entry

If a Token Ring host does not support the use of IEEE 802.2 TEST or XID datagrams as explorer packets, you might need to add static information to the RIF cache of the router/bridge.

To configure a static RIF entry, perform the following task in global configuration mode:

Task	Command
Enter static source-route information into the RIF cache.	**rif** *MACaddress RIF-string* {*interface-name* | **ring-group** *ring*}

Configure the RIF Timeout Interval

RIF information that can be used to bridge routed protocols is maintained in a cache whose entries are aged.

To configure the number of minutes an inactive RIF entry is kept in the cache, perform the following task in global configuration mode:

Task	Command
Specify the number of minutes an inactive RIF entry is kept.	**rif timeout** *minutes*

Configure Translation between SRB and TB Environments

Source-Route Translational Bridging (SR/TLB) is a router software feature that allows you to combine source-route and transparent bridged networks without the need to convert all of your existing source-route bridges to source-route transparent (SRT) nodes. As such, it provides a cost-effective connectivity path between Ethernets and Token Rings, for example.

Note When you are translationally bridging, you will have to route routed protocols and translationally bridge all others, such as LAT.

Overview of SR/TLB

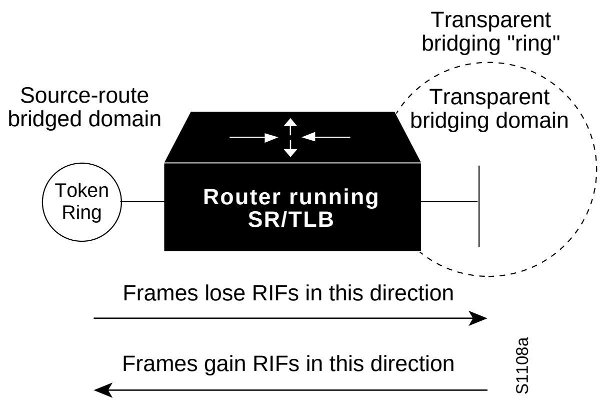
You can bridge packets between a source-route bridging domain and a transparent bridging domain. Using this feature, a software “bridge” is created between a specified virtual ring group and a transparent bridge group. To the source-route station, this bridge looks like a standard source-route bridge. There is a ring number and a bridge number associated with a ring that actually represents the entire transparent bridging domain. To the transparent bridging station, the bridge represents just another port in the bridge group.

When bridging from the source-route (typically, Token Ring) domain to the transparent bridging (typically, Ethernet) domain, the source-route fields of the frames are removed. The RIFs are cached for use by subsequent return traffic.

When bridging from the transparent bridging domain to the source-route bridged domain, the router/bridge checks the packet to see if it has a multicast or broadcast destination or a unicast (single host) destination. If it is multicast, the packet is sent as a spanning-tree explorer. If it is a unicast destination, the router/bridge looks up the path to the destination in the RIF cache. If a path is found, it will be used; otherwise, the router/bridge will send the packet as a spanning-tree explorer.

An example of a simple topology is shown in Figure 1-9.

Figure 1-9 Example of a Simple SR/TLB Topology


Transparent bridging "ring"
Source-route bridged domain
Transparent bridging domain
Token Ring
Router running SR/TLB
Frames lose RIFs in this direction
Frames gain RIFs in this direction
S1108a

Note The spanning-tree protocol messages used to prevent loops in the transparent bridging domain are *not* passed between the source-route domain and the transparent bridging domain. Therefore, you must not set up multiple paths between the source-route and transparent bridging domains.

The following notes and caveats apply to all uses of SR/TLB:

- Multiple paths cannot exist between the source-route bridged domain and the transparent bridged domain. Such paths can lead to data loops in the network, because the spanning-tree packets used to avoid these loops in transparent bridged networks do not traverse the source-route bridged network.
- Some devices, notably PS/2s under certain configurations running OS/2 Extended Edition Version 1.3, do not correctly implement the “largest frame” processing on RIFs received from remote source-route bridged hosts. The maximum Ethernet frame size is smaller than that allowed for Token Ring. As such, bridges allowing for communication between Ethernet and Token Ring will tell the Token Ring hosts, through the RIF on frames destined to the Token Ring, that hosts on the Ethernet cannot receive frames larger than a specified maximum, typically 1472 bytes. Some machines ignore this run-time limit specification and send frames larger than the Ethernet can accept. The router and any other Token Ring/Ethernet bridge has no choice but to drop these frames. To allow such hosts to successfully communicate across or to an Ethernet, you must configure their maximum frame sizes manually. For the PS/2, this can be done through Communications Manager.
- Any access filters applied on any frames apply to the frames as they appear on the media to which the interface with the access filter applies. This is important because in the most common use of SR/TLB (Ethernet and Token Ring connectivity), the bit ordering of the MAC addresses in the frame is swapped. Refer to the SR/TLB examples in the “SRB Configuration Examples” section of this chapter.

Caution Bridging between dissimilar media presents several problems that can prevent communication from occurring. These problems include bit order translation (or usage of MAC addresses as data), maximum transmission unit (MTU) differences, frame status differences, and multicast address usage. Some or all of these problems may be present in a multimedia bridged LAN and prevent communication from taking place. Because of differences in the way end nodes implement Token Ring, these problems are most prevalent when bridging between Token Rings and Ethernets or between Token Ring and FDDI LANs.

We currently know that problems occur with the following protocols when bridged between Token Ring and other media: Novell IPX, DECnet Phase IV, AppleTalk, VINES, XNS, and IP. Further, problems can occur with the Novell IPX and XNS protocols when bridged between FDDI and other media. We recommend that these protocols be routed whenever possible.

To enable SR/TLB, you must perform the following task:

- Enable bridging between transparent bridging and SRB.

In addition, you can also:

- Enable translation compatibility with IBM 8209 bridges
- Use SR/TLB for 0x80d5 processing
- Use SR/TLB for other IBM LLC2 environments

Enable Bridging between Transparent Bridging and SRB

Before enabling bridging, you must have completely configured your router using multiport source-bridge and transparent bridging. Once you have done this, establish bridging between transparent bridging and source-route bridging by performing the following task in global configuration mode:

Task	Command
Enable bridging between transparent bridging and source-route bridging.	**source-bridge transparent** *ring-group pseudo-ring bridge-num tb-group* [*oui*]

Enable Translation Compatibility with IBM 8209 Bridges

To transfer data between IBM 8209 Ethernet/Token Ring bridges and routers running the SR/TLB software (to create a Token Ring backbone to connect Ethernets), perform this task on each Token Ring interface in interface configuration mode:

Task	Command
Move data between IBM 8209 Ethernet/ Token Ring bridges and routers running translational bridging software.	**ethernet-transit-oui standard**

Enable Token Ring LLC2 to Ethernet Conversion

The routers support the following types of Token Ring to Ethernet frame conversions:

- Token Ring LLC2 to Ethernet 802.3 LLC2 (standard)
- Token Ring LLC2 to Ethernet Type II (0x80d5 processing)

For most non-IBM hosts, Token Ring LLC2 frames can be translated in a straightforward manner into Ethernet 802.3 LLC2 frames. This is the default conversion on routers.

However, many Ethernet-attached IBM devices use nonstandard encapsulation of LLC2 on Ethernet. Such IBM devices, including PS/2s running OS/2 Extended Edition and RT-PCs, do not place their LLC2 data inside an 802.3 format frame, but rather place it into an Ethernet Type 2 frame whose type is specified as *0x80d5*. This nonstandard format is called 0x80d5, named after the type of frame. This format is also sometimes called *RT-PC Ethernet format* because these frames were first widely seen on the RT-PC. Hosts using this nonstandard 0x80d5 format cannot read the standard Token Ring LLC2 to Ethernet 802.2 LLC frames.

The format of all these frames is given in the *Internetworking Technology Overview* publication.

To enable Token Ring LLC2 to Ethernet LLC2 conversion, you can perform one or both of the following tasks:

- Enable 0x80d5 Processing
- Enable Standard Token Ring LLC2 to Ethernet LLC2 Conversion

Enable 0x80d5 Processing

You can change the router's default translation behavior of translating Token Ring LLC to Ethernet 802.3 LLC, to translate Token Ring LLC2 frames into Ethernet *0x80d5 format* frames. To enable this nonstandard conversion, perform the following task in global configuration mode:

Task	Command
Change the router's Ethernet/Token Ring translation behavior to translate Token Ring LLC2 frames into Ethernet *0x80d5 format* frames	**source-bridge enable-80d5**

Enable Standard Token Ring LLC2 to Ethernet LLC2 Conversion

After you change the router's translation behavior to perform Token Ring LLC2 frames into Ethernet 80d5 format frames, some of the nonIBM hosts in your network topology may use the standard Token Ring conversion of Token Ring LLC2 to 802.3 LLC2 frames. If this is the case, you can change the translation method of those hosts to use the standard translation method on a per-DSAP basis. The translation method for all the IBM hosts would still remain as Token Ring LLC2 to Ethernet 0x80d5 translation.

To define non-IBM hosts in your network topology to use the standard translation method while the IBM hosts use the nonstandard, perform the following task in global configuration mode:

Task	Command
Allow some other devices to use normal LLC2/IEEE 802.3 translation on a per-DSAP basis	**source-bridge sap-80d5** *sap*

Configure NetBIOS Support

NetBIOS is a nonroutable protocol that was originally designed to transmit messages between stations, typically IBM PCs, on a Token Ring network. NetBIOS allows messages to be exchanged between the stations using a name rather than a station address. Each station knows its name and is responsible for knowing the names of other stations on the network.

Note In addition to this type of NetBIOS, which runs over LLC2, we have implemented another type of NetBIOS that runs over IPX. For information on the IPX type of NetBIOS, refer to the “Configuing Novell IPX” chapter of this manual.

NetBIOS name caching allows the router to maintain a cache of NetBIOS names, which avoids the high overhead of transmitting many of the broadcasts used between client and server NetBIOS PCs (IBM PCs or PS/2s) in a source-route bridging environment.

When NetBIOS name caching is enabled, the router performs the following actions:

- Notices when any hosts send a series of duplicated “query” frames and reduces them to one frame per period. The time period is configurable.
- Keeps a cache of mappings between NetBIOS server and client names and their MAC addresses. By watching NAME_QUERY and NAME_RECOGNIZED request and response traffic between clients and servers, the router can forward broadcast requests sent by clients to find servers (and by servers in reply to their clients) directly to their needed destinations, rather than forwarding them for broadcast across the entire bridged network.

The router will time out the entries in the NetBIOS name cache after a specific interval of their initial storage. The timeout value is a user-configurable value. You can configure the timeout value for a particular Token Ring if the NetBIOS name cache is enabled on the interface connecting to that Token Ring. In addition you can configure static name cache entries that never time out for frequently accessed servers whose locations or paths typically do not change. Static RIF entries are also specified for such hosts.

Generally, NetBIOS name caching is most useful when a large amount of NetBIOS broadcast traffic creates bottlenecks on WAN media connecting distant locations, and the WAN media is overwhelmed with this traffic. However, when two high-speed LAN segments are directly interconnected, the packet savings of NetBIOS name caching is probably not worth the router processor overhead associated with it.

Note NetBIOS name caching is not recommended to be turned on in backbone routers, particularly if you have it enabled in all the routers connected to the backbone. NetBIOS caching should be distributed among multiple routers.

Note NetBIOS name caching can be used only between routers that are running Software Release 9.1 or later.

To enable NetBIOS name caching, you must perform the following tasks:

- Ensure that proxy explorers is enabled on the appropriate interface
- Specify timeout and enable NetBIOS name caching

In addition, you can configure NetBIOS name caching as follows:

- Create static entries in the NetBIOS name cache
- Specify dead-time intervals for NetBIOS packets

Ensure that Proxy Explorers Is Enabled on the Appropriate Interface

In order to enable NetBIOS name caching on an interface, the proxy explorers feature must first be enabled on that interface. This feature must either be enabled for response to all explorer packets or for response to NetBIOS packets only.

To enable this feature, perform the following task in EXEC mode:

Task	Command
Ensure that proxy explorers has been enabled	**show configuration**

To determine whether proxy explorers has been configured for response to all explorer packets, look in the router's configuration file for the **source-bridge proxy-explorer** entry for the appropriate interface. For example, if the appropriate interface is Token Ring 0, look for an entry similar to the following:

```
!
interface tokenring 0
source-bridge proxy-explorer
!
```

If that entry does not exist, look for the **source-bridge proxy-netbios-only** entry for the appropriate interface.

If neither entry exists, proxy explorers has not yet been enabled for the appropriate interface. To enable proxy explorers for response to all explorer packets, refer to the section "Configure Proxy Explorers" later in this chapter.

Otherwise, enable proxy explorers only for the NetBIOS name caching function by performing the following task in interface configuration mode:

Task	Command
Enable use of proxy explorers only for the NetBIOS name caching function and not for their general local response to explorers.	**source-bridge proxy-netbios-only**

Specify Timeout and Enable NetBIOS Name Caching

After you have ensured that proxy explorers has been enabled for the appropriate interface, you can proceed to specify a cache timeout and to enable NetBIOS name caching. To do this, perform these tasks in global configuration mode:

Task	Command
Specify the timeout for entries in the router's NetBIOS name cache.	**netbios name-cache timeout** *minutes*
Enable NetBIOS name caching for the appropriate interfaces.	**netbios enable-name-cache**

Create Static Entries in the NetBIOS Name Cache

If the router communicates with one or more NetBIOS stations on a regular basis, adding static entries to the NetBIOS name cache for these stations can reduce network traffic and router overhead. You can define a static NetBIOS name cache entry that associates the server with the NetBIOS name and the *MAC-address*. If the router acts as a NetBIOS server, you can specify that the static NetBIOS name cache is available locally through a particular interface. If a remote router acts as the NetBIOS server, you can specify that the NetBIOS name cache is available remotely. To do this, perform one of the following tasks in global configuration mode:

Task	Command
Define a static NetBIOS name cache entry and specify that it is available locally through a particular interface.	**netbios name-cache** *MAC-address NetBIOS-name interface-name*
Define a static NetBIOS name cache entry and specify that it is available remotely.	**netbios name-cache** *MAC-address NetBIOS-name* **ring-group** *group-number*

If you have defined a NetBIOS name cache entry, you must also define a RIF entry. For an example of how to configure a static NetBIOS entry, see the "Example of NetBIOS Support with a Static NetBIOS Cache Entry" section later in this chapter.

Specify "Dead Time" Intervals for NetBIOS Packets

When NetBIOS name caching is enabled and default parameters are set on the router (as well as the NetBIOS name server and the NetBIOS name client), approximately 20 broadcast packets per logon are kept on the local ring where they are generated. The broadcast packets are of the type ADD_NAME_QUERY, ADD_GROUP_NAME, and STATUS_QUERY.

The router also converts pairs of FIND_NAME and NAME_RECOGNIZED packets received from explorers, which traverse all rings, to specific route frames that are sent only between the two machines that need to see these packets.

The combination of name caching and name pairing often results in significant packet savings on source-route bridged Token Ring networks.

To specify dead time intervals, perform one or both of the following tasks in global configuration mode:

Task	Command
Specify a "dead time" interval during which the router drops any broadcast (NetBIOS ADD_NAME_QUERY, ADD_GROUP_NAME, or STATUS_QUERY) frames if they are duplicate frames sent by the same host.	**netbios name-cache query-timeout** *seconds*
Specify a "dead time" interval during which the router drops FIND_NAME and NAME_RECOGNIZED frames if they are duplicate frames sent by the same host.	**netbios name-cache recognized-timeout** *seconds*

Configure LAN Network Manager Support

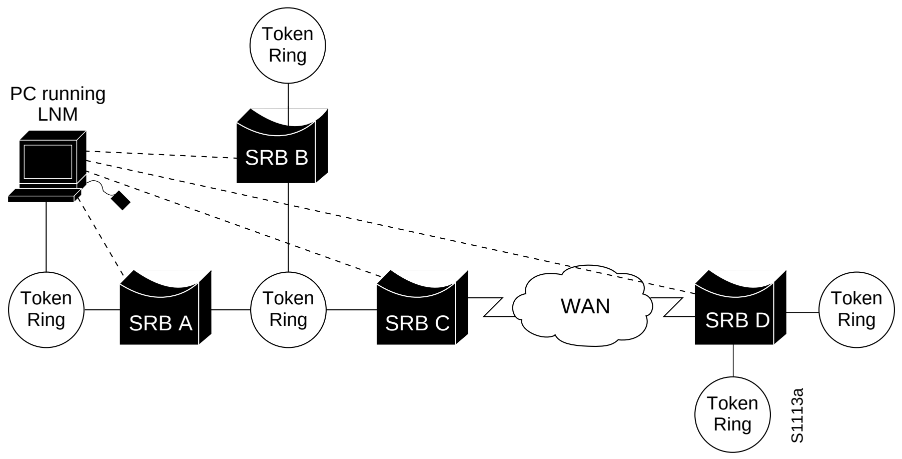
LAN Network Manager (LNM), formerly called LAN Manager, is an IBM product for managing a collection of source-route bridges. Using either a proprietary protocol or SNMP, LNM allows you to monitor the entire collection of Token Rings that comprise your source-route bridged network. You can use LNM to manage the configuration of source-route bridges, monitor Token Ring errors, and gather information from Token Ring parameter servers.

Note LNM is supported on the 4/16-Mb Token Ring cards that can be configured for either 4- or 16-Mb transmission speeds. LNM support is not provided on CSC-R16M cards with SBEMON 2.0.

LNM is not limited to managing locally attached Token Ring networks; it also can manage any other Token Rings in your source-route bridged network that are connected through non-Token Ring media. To accomplish this task, LNM works in conjunction with the IBM Bridge Program. The IBM Bridge Program gathers data about the local Token Ring network and relays it back to LNM. In this manner, the bridge program becomes a proxy for information about its local Token Ring. Without this ability, you would require direct access to a device on every Token Ring in the network. This process would make managing a source-route bridged environment awkward and cumbersome.

Figure 1-10 shows some Token Rings attached through a cloud and one LNM linking to a source-route bridge on each local ring.

Figure 1-10 LNM Linking to a Source-Route Bridge on Each Local Ring



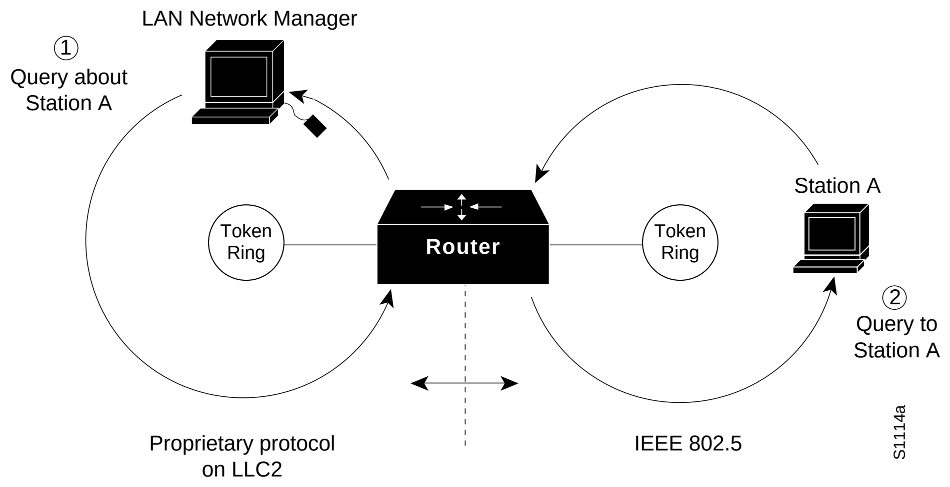
If LNM requires information about a station somewhere on a Token Ring, it uses a proprietary IBM protocol to query to one of the source-route bridges connected to that ring. If the bridge can provide the requested information, it simply responds directly to LNM. If the bridge does not have the necessary information, it queries the station using a protocol published in the IEEE 802.5 specification. In either case, the bridge uses the proprietary protocol to send a valid response back to LNM, using the proprietary protocol.

As an analogy, consider a language translator who sits between a French-speaking diplomat and a German-speaking diplomat. If the French diplomat asks the translator a question in French for the German diplomat and the translator knows the answer, he or she simply responds without translating the original question into German. If the French diplomat asks a question the translator does not know how to answer, he or she must first translate the question to German, wait for the German diplomat to answer, and then translate the answer back to French.

Similarly, if LNM queries a source-route bridge in the proprietary protocol and the bridge knows the answer, it responds directly using the same protocol. If the bridge does not know the answer, it must first translate the question to the IEEE 802.5 protocol, query the station on the ring, and then translate the response back to the proprietary protocol to send to LNM.

Figure 1-11 illustrates requests from the LNM originating in an IBM proprietary protocol, and then translated into IEEE 802.5 MAC-level frames.

Figure 1-11 LAN Network Manager Monitoring and Translating



Notice that the proprietary protocol LNM uses to communicate with the source-route bridge is a nonroutable protocol running over a Logical Link Control (LLC2) connection. Although its protocol cannot be routed, LNM can monitor or manage anything within the source-route bridging network.

How the Router Works with LNM

As of Software Release 9.0, our routers using 4/16-Mbps Token Ring interfaces configured for source-route bridging support the proprietary protocol that LNM uses. These routers provide all functions the IBM Bridge Program currently provides. Thus LNM can communicate with a router as if it were an IBM source-route bridge, such as the IBM 8209, and can manage or monitor any Token Ring connected to the router.

Through IBM Bridge support, LNM provides three basic services for the source-route bridging network:

- The Configuration Report Server (CRS) monitors the current logical configuration of a Token Ring and reports any changes to LNM. CRS also reports various other events, such as the change of an Active Monitor on a Token Ring.
- The Ring Error Monitor (REM) monitors errors reported by any station on the ring. In addition, REM monitors whether the ring is in a functional or a failure state.
- The Ring Parameter Server (RPS) reports to LNM when any new station joins a Token Ring and ensures that all stations on a ring are using a consistent set of reporting parameters.

IBM Bridge support for LNM also allows asynchronous notification of some events that can occur on a Token Ring. Examples of these events include notification of a new station joining the Token Ring or of the ring entering failure mode, known as *beaconing*. Support also is provided for LNM to change the operating parameters in the bridge. For a complete description of LNM, refer to the IBM product manual supplied with the LNM program.

LNM support in our source-route bridges is a powerful tool for managing source-route bridged networks. Through the ability to communicate with LNM and to provide the functionality of the IBM Bridge Program, our device appears as part of the IBM network. You therefore gain from the interconnectivity of our products without having to learn a new management product or interface.

When source-route bridging is enabled on the router, configuring the routers to perform the functions of an IBM Bridge for communication with LNM occurs automatically. Therefore, if source-route bridging has been enabled on the router, you do not need to perform any tasks to enable LNM support. However, the LNM software residing on a management station on a Token Ring on the network should be configured to properly communicate with the router.

There are several options for modifying LNM parameters in the router, but none are required for basic functionality. For example, because users can now modify the operation of the router through the Simple Network Management Protocol (SNMP) as well as through LNM, there is an option to exclude a user from modifying the router configuration through LNM. You also can specify which of the three LNM services (CRS, REM, RPS) the source-route bridge will perform.

To configure LNM support, perform these tasks:

- Configure LNM software on the management stations to communicate with the router
- Prevent LNM stations from modifying router parameters
- Enable other LRMs to change router/bridge parameters
- Apply a password to an LNM reporting link
- Enable LNM servers
- Change reporting thresholds
- Change an LNM reporting interval
- Monitor LNM operation

Configure LNM Software on the Management Stations to Communicate with the Router

Because configuring an LNM station is a fairly simple task and is well covered in the LNM documentation, it will not be covered in depth here. However, it is important to mention that the MAC addresses of the interfaces comprising the ports of the bridges must be entered as adapter addresses. When the router is configured as a multiport bridge, configuring an LNM station is complicated by the virtual ring that is involved. The basic problem extends from the fact that LNM is designed to only understand the concept of a two-port bridge, and the router with a virtual ring is a *multiport* bridge. The solution is to configure a virtual ring into the LNM Manager station as a series of two-port bridges.

Prevent LNM Stations from Modifying Router Parameters

Because there is now more than one way to remotely change parameters in a router (either using SNMP or the proprietary IBM protocol), some way is needed to prevent such changes from detrimentally interacting with each other.You can prevent any LNM station from modifying parameters in the router. It does not affect the ability of LNM to monitor events, only to change parameters in the router.

To prevent the modification of router parameters by LNM station, perform the following task in global configuration mode:

Task	Command
Prevent LNM stations from modifying LNM parameters in the router.	**lnm snmp-only**

Enable Other LRMs to Change Router/Bridge Parameters

LNM has a concept of reporting links and reporting link numbers. A reporting link is simply a connection (or potential connection) between a LAN Reporting Manager (LRM) and a bridge. A reporting link number is a unique number used to identify a reporting link. An IBM bridge allows four simultaneous reporting links numbered 0 through 3. Only the LRM attached on the lowest-numbered connection is allowed to change LNM parameters in the router, and then only when that connection number falls below a certain configurable number. In the default configuration, the LRM connected through link 0 is the only LRM that can change LNM parameters in the router.

To enable other LRMs to change router/bridge parameters, perform the following task in interface configuration mode:

Task	Command
Enable a LRM other than that connected through link 0 to change router/bridge parameters.	**lnm alternate** *number*

Apply a Password to an LNM Reporting Link

Each reporting link has its own password that is used not only to prevent unauthorized access from an LRM to a bridge, but to control access to the different reporting links. This is important because it is possible to change parameters through some reporting links.

To apply a password to an LNM reporting link, perform the following task in interface configuration mode:

Task	Command
Apply a password to an LNM reporting link.	**lnm password** *number string*

Enable LNM Servers

As in an IBM bridge, the router provides several functions that gather information from a local Token Ring. All of these functions are enabled by default, but also can be disabled. The LNM servers are explained in the section "How the Router Works with LNM" earlier in this chapter.

To enable LNM servers, perform one or more of the following tasks in interface configuration mode:

Task	Command
Enable the LNM Configuration Report Server (CRS).	**lnm crs**
Enable the LNM Ring Error Monitor (REM).	**lnm rem**
Enable the LNM Ring Parameter Server (RPS).	**lnm rps**

Change Reporting Thresholds

The router sends a message to all attached LNMs whenever it begins to drop frames. The threshold at which this report is generated is based on a percentage of frames dropped compared with those forwarded. This threshold is configurable, and defaults to a value of 0.10 percent. You can configure the threshold by entering a single number, expressing the percentage loss rate in hundredths of a percent. The valid range is 0 to 9999.

To change reporting thresholds, perform the following task in interface configuration mode:

Task	Command
Change the threshold at which the router reports the frames-lost percentage to LNM.	**lnm loss-threshold** *number*

Change an LNM Reporting Interval

All stations on a Token Ring notify the Ring Error Monitor (REM) when they detect errors on the ring. In order to prevent excessive messages, error reports are not sent immediately, but are accumulated for a short time interval and then reported. A station learns the duration of this interval from a router (configured as a source-route bridge) when it first enters the ring. This value is expressed in tens of milliseconds between error messages. The default is 200, or 2 seconds. The valid range is 0 to 65535.

To change an LNM reporting interval, perform the following task in interface configuration mode:

Task	Command
Set the time interval during which stations report ring errors to the Ring Error Monitor (REM).	**lnm softerr** *number*

Monitor LNM Operation

Once LNM support is enabled, you can monitor LNM operation. To observe the configuration of the LNM bridge and its operating parameters, perform the following tasks in the EXEC mode:

Task	Command
Display all configured bridges and their global parameters	**show lnm bridge**
Display the logical configuration of all bridges configured in the router	**show lnm config**
Display LNM information for an interface or all interfaces of the router	**show lnm interface**
Display LNM information about a Token Ring or all Token Rings on the network	**show lnm ring**
Display LNM information about a station or all stations on the network	**show lnm station**

Secure the SRB Network

This section describes how to configure three features that are used primarily to provide network security: NetBIOS access filters, administrative filters, and access expressions that can be combined with administrative filters. In addition, these features can be used to increase network performance, because they reduce the number of packets that traverse the backbone network.

Configure NetBIOS Access Filters

NetBIOS packets can be filtered when transmitted across a Token Ring bridge. Two types of filters can be configured: one for source and destination station names and one for arbitrary byte patterns in the packet itself.

As you configure NetBIOS access filters, keep the following issues in mind:

- The access lists that apply filters to an interface are scanned in the order they are entered.
- There is no way to put a new access list entry in the middle of an access list. All new additions to existing NetBIOS access lists are placed at the end of the existing list.
- Access list arguments are case-sensitive. The software makes a literal translation, so that a lowercase "a" is different from an uppercase "A." (Most nodes are named in uppercase letters.)
- A host NetBIOS access list and byte NetBIOS access list can each use the same name. The two lists are identified as unique and bear no relationship to each other.
- The station names included in the access lists are compared with the source name field for NetBIOS commands 00 and 01 (ADD_GROUP_NAME_QUERY and ADD_NAME_QUERY), as well as the destination name field for NetBIOS commands 08 and 0A (DATAGRAM and NAME_QUERY).
- If an access list does not contain a particular station name, the default action is to deny the access to that station.

In order to minimize any performance degradation, NetBIOS access filters do not examine all packets. Rather, they examine certain packets that are used to establish and maintain NetBIOS client/server connections, thereby effectively stopping new access and load across the router. However, applying a new access filter does not terminate existing sessions immediately. All new sessions will be filtered, but existing sessions could continue for some time.

There are two ways you can configure NetBIOS access filters:

- Configure NetBIOS access filters using station names
- Configure NetBIOS access filters using a byte offset

Configure NetBIOS Access Filters Using Station Names

To configure access filters using station names, you must:

- Assign the station access list name
- Specify the direction of the message to be filtered on the interface

Assign the Station Access List Name

The NetBIOS station access list contains the station name to match, along with a permit or deny condition. You must assign the name of the access list to a station or set of stations on the network.

To assign a station access list name, perform the following task in global configuration mode:

Task	Command
Assign the name of an access list to a station or set of stations on the network.	**netbios access-list host** *name* {**permit** | **deny**} *pattern*

Specify the Direction of the Message to Be Filtered on the Interface

When filtering by station name, you can choose to filter either incoming or outgoing messages on the interface. To specify the direction, perform the one of the following tasks in interface configuration mode:

Task	Command
Define an access list filter for incoming messages.	**netbios input-access-filter host** *name*
Define an access list filter for outgoing messages.	**netbios output-access-filter host** *name*

Configure Access Filters Using a Byte Offset

To configure access filters using a byte offset, you must:

- Assign a byte offset access list name
- Specify the direction of the message to be filtered on the interface

Configuration Tips and Notes

Keep the following notes in mind while configuring access filters using a byte offset:

- The access lists that apply filters to an interface are scanned in the order they are entered.
- There is no way to put a new access list entry in the middle of an access list. All new additions to existing NetBIOS access lists are placed at the end of the existing list.
- When an access list entry has an offset plus the length of the pattern that is larger than the packet's length, the entry will not make a match for that packet.
- A host NetBIOS access list and byte NetBIOS access list can each use the same name. The two lists are identified as unique and bear no relationship to each other.
- Because these access lists allow arbitrary byte offsets into packets, these access filters can have a significant impact on the amount of packets per second transiting across the bridge. They should be used only when situations absolutely dictate their use.
- If an access list does not contain a particular station name, the default action is to deny the access to that station.

Assign a Byte Offset Access List Name

The NetBIOS byte offset access list contains a series of offsets and hexadecimal patterns with which to match byte offsets in NetBIOS packets. To assign a byte offset access list name, perform the following task in global configuration mode:

Task	Command
Define the byte offsets and patterns within NetBIOS messages to match with access list parameters.	**netbios access-list bytes** *name* {**permit** | **deny**} *offset pattern*

Note Using NetBIOS Byte Offset access filters disables the autonomous or fast switching of source-route bridging frames.

Specify the Direction of the Message to be Filtered on the Interface

When filtering by byte offset, you can filter either incoming or outgoing messages on the interface. To specify the direction, perform one of the following tasks in interface configuration mode:

Task	Command
Specify a byte-based access filter on incoming messages.	**netbios input-access-filter bytes** *name*
Specify a byte-based access filter on outgoing messages.	**netbios output-access-filter bytes** *name*

Configure Administrative Filters for Token Ring Traffic

Source-route bridges normally filter frames according to the routing information contained in the frame. That is, a bridge will not forward a frame back to its originating network segment or any other network segment that the frame has already traversed. This section describes how to configure another type of filter—the administrative filter.

Administrative filters can filter frames based on the following methods:

- Protocol type—IEEE 802 or Subnetwork Access Protocol (SNAP)
- Token Ring vendor code
- Source address
- Destination address

Whereas filtering by Token Ring address or vendor code causes no significant performance penalty, filtering by protocol type significantly affects performance. A list of SNAP (Ethernet) type codes is provided in Appendix B, “Ethernet Type Codes,” in the *Router Products Command Reference* publication.

Filter Frames by Protocol Type

You can filter frames by protocol type by specifying protocol type codes in an access list. You then apply that access list to either IEEE 802.2 encapsulated packets or to SNAP-encapsulated packets on the appropriate interface.

The order in which you specify these elements affects the order in which the access conditions are checked. Each condition is tested in succession. A matching condition is then used to execute a permit or deny decision. If no conditions match, a deny decision is reached.

Note If a *single condition* is to be denied, there must be an **access-list** command that permits *everything* as well, or *all* access is denied.

To filter frames by protocol type, perform the following task in global configuration mode:

Task	Command
Create an access list for filtering frames by protocol type.	**access-list** *list* {**permit** | **deny**} *type-code wild-mask address mask*

You can filter IEEE 802-encapsulated packets on either input or output. The access list you specify is the one you created that includes the protocol type codes.

To enable filtering on input or output, perform one of the following tasks in interface configuration mode:

Task	Command
Enable filtering of IEEE 802-encapsulated packets on input by type code.	**source-bridge input-lsap-list** *list*
Enable filtering of IEEE 802-encapsulated packets on output by type code.	**source-bridge output-lsap-list** *list*

You can filter SNAP-encapsulated packets on either input or output. The access list you specify is the one you created that includes the protocol type codes.

To enable filtering on input or output, perform one of the following tasks in interface configuration mode:

Task	Command
Filter SNAP-encapsulated packets on input by type code.	**source-bridge input-type-list** *list*
Filter SNAP-encapsulated frames on output by type code.	**source-bridge output-type-list** *list*

Filter Frames by Vendor Code

To configure administrative filtering by vendor code or address, define access lists that look for Token Ring addresses or for particular vendor codes for administrative filtering. To do so, perform the following task in global configuration mode:

Task	Command
Configure vendor code access lists.	**access-list** *list* {**permit** | **deny**} *address mask*

Filter Input by Source Addresses

To configure filtering on IEEE 802 source addresses, assign an access list to a particular input interface for filtering the Token Ring or IEEE 802 source addresses. To do so, perform the following task in interface configuration mode:

Task	Command
Enable filtering on IEEE 802 source addresses.	**source-bridge input-address-list** *list*

Filter Output by Source Addresses

To configure output filtering on IEEE 802 source addresses, assign an access list to a particular output interface. To do so, perform the following task in interface configuration mode:

Task	Command
Enable filtering on IEEE 802 destination addresses.	**source-bridge output-address-list** *list*

Configure Access Expressions that Combine Administrative Filters

You can use access expressions to combine access filters to establish complex conditions under which bridged frames can enter or leave an interface. Using access expressions, you can achieve levels of control on the forwarding of frames that otherwise would be impossible when using only simple access filters.

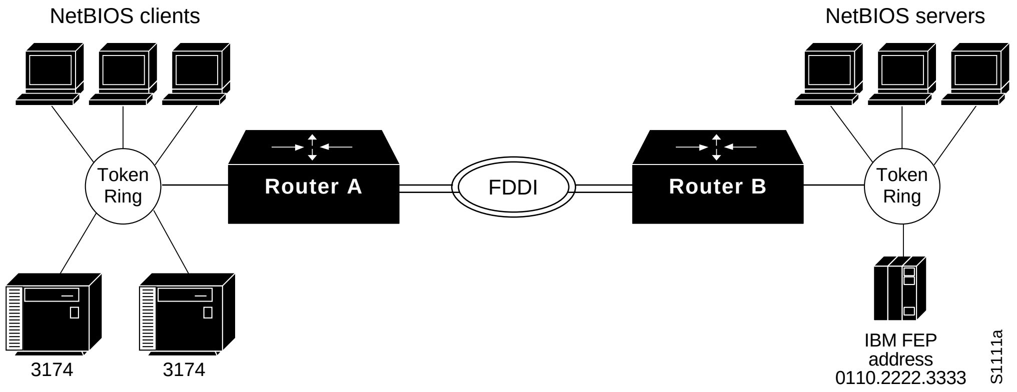
Access expressions are constructed from individual access lists that define administrative filters for the following fields in packets:

- LSAP and SNAP type codes
- MAC addresses
- NetBIOS station names
- NetBIOS arbitrary byte values

Note For any given router interface, an access expression cannot be used if an access list has been defined for a given direction. For example, if an input access list is defined for MAC addresses on an interface, no access expression can be specified for the input side of that interface.

Figure 1-12 shows how access expressions can be useful.

Figure 1-12 Access Expression Example



In Figure 1-12, two routers each connect a Token Ring to an FDDI backbone. On both Token Rings, SNA and NetBIOS bridging support is required. On Token Ring A, NetBIOS clients must communicate with any NetBIOS server off Token Ring B or any other, unpictured router. However, the 3174s off Token Ring A must only communicate with the one FEP off of Token Ring B, located at MAC address 0110.2222.3333.

Without the using access expressions, this scenario cannot be achieved. A filter on Router A that restricted access to only the FEP would have the side result of also restricting access of the NetBIOS clients to the FEP. What is needed is an access *expression* that would state "If it is a NetBIOS frame, pass through, but if it is an SNA frame, allow only access to address 0110.2222.3333."

Note Using access-expressions that combine access filters disables the autonomous or fast switching of source-route bridging frames.

Configure Access Expressions

To configure an access expression perform the following tasks:

- Design the access expression
- Configure the access lists used by the expression
- Configure the access expression into the router

Design the Access Expression

When designing an access expression, you must create some phrase that indicates, in its entirety, all the frames that will *pass* the access expression. This access expression is designed to apply on frames coming from the Token Ring interface on Router A in Figure 1-12:

"Pass the frame if it is a NetBIOS frame or if it is an SNA frame destined to address 0110.2222.3333."

In Boolean form, this phrase can be written as follows:

"Pass if "NetBIOS or (SNA and destined to 0110.2222.3333).""

Configure the Access Lists Used by the Expression

The preceding statement requires three access lists to be configured:

- An access list that passes a frame if it is a NetBIOS frame (SAP = 0xF0F0)
- An access list that passes a frame if it is an SNA frame (SAP = 0x0404)
- An access list that passes a MAC address of 0110.2222.3333

The following configuration allows for all these conditions:

```
! Access list 201 passes NetBIOS frames (command or response)
access-list 201 permit 0xF0F0 0x0001
!
access-list 202 permit 0x0404 0x0001 ! Permits SNA frames (command or response)
access-list 202 permit 0x0004 0x0001 ! Permits SNA Explorers with NULL DSAP
!
! Access list 701 will permit the FEP MAC address
! of 0110.2222.3333
access-list 701 permit 0110.2222.3333
!
```

Configure the Access Expression into the Router

Apply the access expression to the appropriate interface by performing the following task in interface configuration mode:

Task	Command
Define a per-interface access expression	**access-expression** {**in** | **out**} *expression*

Optimize Access Expressions

It is possible to achieve the same result with more than one access expression. Suppose you wanted to transmit SNA traffic through to a single address, but allow other traffic through the router without restriction. The phrase could be written as follows:

“Allow access if the frame is not an SNA frame, or if it is going to host 0110.2222.3333.”

More tersely this would be:

“Not SNA or destined to 0110.2222.3333.”

Using the access lists defined above, the resulting configuration would be as follows:

```
!
interface tokenring 0
access-expression in ~lsap(202) | dmac(701)
!
access-list 202 permit 0x0404 0x0001 ! Permits SNA frames (command or response)
access-list 202 permit 0x0004 0x0001 ! Permits SNA Explorers with NULL DSAP
!
! Access list 701 will permit the FEP MAC address
! of 0110.2222.3333
access-list 701 permit 0110.2222.3333
```

This is a better and simpler access list than the one originally introduced and will probably result in better run-time execution as a result. Therefore, it is best to simplify your access expressions as much as possible before configuring them into the router.

Alter Access Lists Used in Access Expressions

Because access expressions are composed of access lists, special care must be taken when deleting and adding access lists that are referenced in these access expressions.

If an access list that is referenced in an access expression is deleted, the access expression merely ignores the deleted access list. However, if you want to redefine an access list you can create a new access list with the appropriate definition and use the same name as the old access list. The newly defined access list replaces the old one of the same name.

For example, if you want to redefine the NetBIOS access list named MIS that was used in the preceding example, you would enter the following sequence of configuration commands:

```
!  Replace the NetBIOS access list
interface tokenring 0
access-expression in (smac(701) & netbios-host(accept))
no netbios access-list host accept permit CISCO*
```

Tune the SRB Network

This section describes how to configure features that enhance network performance by reducing the number of packets that traverse the backbone network.

You can tune the source-route bridging network as follows:

- Prioritize traffic based on SNA local LU addresses
- Enable class of service
- Assign a priority group to an input interface
- Enable or disable the source-route fast-switching cache
- Enable or disable the source-route autonomous-switching cache
- Configure proxy explorers
- Configure the largest frame size

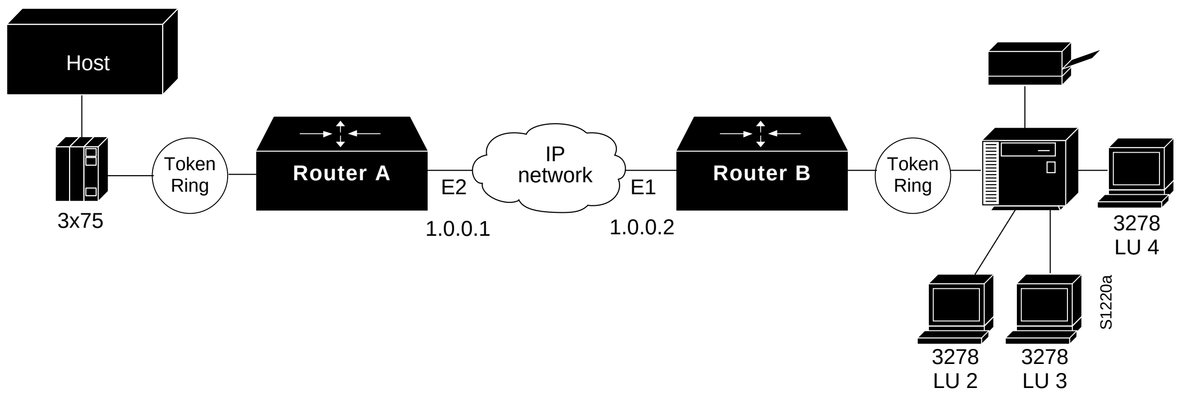
Note In some situations, you may discover that default settings for LLC2 configurations are not acceptable. In such a case, you can configure LLC2 for optimal use. The “Configuring LLC2 and SDLC Parameters” chapter of this manual describes how you can use them to optimize your network performance.

Prioritize Traffic Based on SNA Local LU Addresses

You can prioritize SNA traffic on an interface configured for either Serial Tunnel (STUN) or remote source-route bridging communication. The SNA Local LU address prioritization feature allows SNA traffic to be prioritized according to the address of the Logical Units (LU) on the FID2 transmission headers. Currently, only dependent LUs are supported. The prioritization takes place on LU-LU traffic between an SNA Node type 5 or Node type 4, and Node type 2.

Figure 1-13 shows how SNA Local Address Prioritization can be used.

Figure 1-13 SNA Local Address Prioritization



In Figure 1-13, the IBM mainframe is channel-attached to a 3x75 FEP, which is connected to a cluster controller via remote source-route bridging. Multiple 3270 terminals and printers, each with a unique local LU address, are then attached to the cluster controller. By applying SNA Local LU Address Prioritization, each LU associated with a terminal or printer can be assigned a priority; that is, certain users can have terminals that have better response time than others, and printers can have lowest priority.

Note Both Local Acknowledgment and TCP priority features for STUN or remote source-route bridging must be turned on for SNA local address prioritization to take effect.

With the SNA local LU address prioritization feature, you can establish queuing priorities based on the address of the logical unit. To prioritize traffic, perform the following tasks in global configuration mode:

Task	Command
Map LUs to TCP port numbers.	**locaddr-priority-list** *list address-number queue-keyword*
Set the priority of TCP port numbers	**priority-list** *list* **protocol** *protocol priority* **tcp** *port-number*

Enable Class of Service

To prioritize SNA traffic across the SNA backbone network, you can enable the class of service feature. This feature is useful only between FEP-to-FEP (PU4-to-PU4) communication across the non-SNA backbone. It allows important FEP traffic to flow on high-priority queues.

In order to enable class of service, IP encapsulation over a TCP connection and LLC2 Local Acknowledgment must be enabled.

To enable class of service, perform the following task in interface configuration mode:

Task	Command
Enable class-of-service.	**source-bridge cos-enable**

Assign a Priority Group to an Input Interface

You can assign a priority group to an input interface. To do so perform the following task in interface configuration mode:

Task	Command
Assign a priority group to an input interface.	**locaddr-priority** *list*

Enable or Disable the Source-Route Fast-Switching Cache

Rather than processing packets at the process level, the fast-switching feature enables the router to process packets at the interrupt level. Each packet is transferred from the input interface to the output interface without copying the entire packet to main system memory. Fast switching allows for faster implementations of local source-route bridging between 4/16-Mb Token Ring cards in the same router/bridge, or between two router/bridges using the 4/16-Mb Token Ring cards and direct encapsulation.

By default, fast-switching software is enabled when source-route bridging is enabled. To enable or disable source-route fast-switching, perform one of the following tasks in interface configuration mode:

Task	Command
Enable fast-switching.	**source-bridge route-cache**
Disable fast-switching.	**no source-bridge route-cache**

Note Using either NetBIOS Byte Offset access filters or access expressions that combine access filters disables the fast switching of source-route bridging frames.

Enable or Disable the Source-Route Autonomous-Switching Cache

Autonomous switching is a feature that enables the router to transmit packets from the input ciscoBus card to the output ciscoBus card without any involvement on the part of the router processor.

Autonomous switching is available for local source-route bridging between ciscoBus Token Ring (CTR) cards in the same router/bridge. Autonomous switching provides higher switching rates than does fast-switching between 4/16-MbToken Ring cards. Autonomous switching works for both two-port bridges and multiport bridges that use ciscoBus Token Ring cards.

In a virtual ring that includes both ciscoBus Token Ring and 4/16-MbToken Ring interfaces, frames that flow from one CTR interface to another are autonomously switched, and the remainder of the frames are fast switched. The switching that occurs on the CTR interface takes advantage of the high-speed ciscoBus controller processor.

To enable or disable source-route autonomous switching, perform one of the following tasks in interface configuration mode:

Task	Command
Enable autonomous switching.	**source-bridge route-cache cbus**
Disable autonomous switching.	**no source-bridge route-cache cbus**

Note Using either NetBIOS Byte Offset access filters or access-expressions that combine access filters disables the autonomous switching of source-route bridging frames.

Configure Proxy Explorers

You can use the proxy explorers feature to limit the amount of explorer traffic propagating through the source-bridge network.

Briefly, this feature operates as follows. When proxy explorers is enabled and the router receives an explorer packet, the router compares the destination MAC address of the explorer packet against those in a cache of destination MAC addresses and routes. If a match is found, the router adds the route to that destination address to the explorer packet and returns that packet to its source address, thus reducing the explorer packet's journey on the network. If, however, the destination address is not found in the cache, the router allows the explorer packet to traverse the network to find its destination address, and when the router encounters this explorer packet on its return to its source address, the router caches the route to this destination address for future reference.

Typically, proxy explorers reduces explorer traffic on a bridged network by half. In general, the larger the bridged network, the more useful the proxy explorers feature is. It is particularly useful when communicating across low-bandwidth serial lines or when there are multiple connections to a single node.

To configure proxy explorers, perform the following task in interface configuration mode:

Task	Command
Enable the interface to respond to any explorer packets that meet certain conditions necessary for a proxy response to occur.	**source-bridge proxy-explorer**

Configure the Largest Frame Size

You can configure the largest frame size that is used to communicate with any peers in the ring group.

Generally, the router and the LLC2 device with which it communicates should support the same maximum SDLC I-frame size. The larger this value, the more efficiently the line is used, thus increasing performance.

Faster screen updates to 3278-style terminals often result by configuring the Token Ring FEP to send as large an I-frame as possible and then allowing the router to segment the frame into multiple SDLC I-frames.

After the Token Ring FEP has been configured to send the largest possible I-frame, it is best to configure the router to support the same maximum I-frame size. The default is 516 bytes. The maximum value the router can support is 8144 bytes.

To configure the largest frame size, perform the following task in interface configuration mode:

Task	Command
Specify the largest frame size used to communicate with any peers in the ring group.	**source-bridge largest-frame** *ring-group size*

Establish SRB Interoperability with Specific Token Ring Implementations

This section describes how you can establish interoperability between router/bridges and specific Token Ring implementations. It includes the following tasks:

- Establish SRB interoperability with IBM PC/3270 emulation software.
- Establish SRB interoperability with TI MAC firmware.

Establish SRB Interoperability with IBM PC/3270 Emulation Software

You can establish interoperability with the IBM PC/3270 emulation program Version 3.0, even though it does not properly send packets over a source-route bridge.

Our implementation rewrites the RIF headers of the explorer packets that the PC/3270 emulation program sends to go beyond the local ring, thus confusing the IBM implementation into not looking beyond the local ring for the remote host.

To rewrite RIF headers, perform the following task in interface configuration mode:

Task	Command
Rewrite the RIF headers of explorer packets send by the PC/3270 emulation program to go beyond the local ring.	**source-bridge old-sna**

Establish SRB Interoperability with TI MAC Firmware

You can use a workaround to establish interoperability with Texas Instruments (TI)MAC firmware.

There is a known defect in earlier versions of the TI Token Ring MAC firmware. This implementation is used by Proteon, Apollo, and IBM RTs. A host using a MAC address whose first two bytes are zeros (such as a Cisco router/bridge) will not properly communicate with hosts using that version of TI firmware.

There are two solutions. The first involves installing a static RIF entry for every faulty node with which the router communicates. If there are many such nodes on the ring, this may not be practical.

You also can set the MAC address of our Token Ring to a value that works around the problem. Resetting the MAC address forces the use of a different MAC address on the specified interface, thereby avoiding the TI MAC firmware problem. However, you must ensure that no other host on the network is using that MAC address.

To reset the MAC address, perform the following task in interface configuration mode:

Task	Command
Reset the MAC address of the Token Ring interface to a value that provides a workaround to a problem in TI Token Ring MAC firmware.	**mac-address** *ieee-address*

Reporting Spurious Frame-Copied Errors

An IBM 3174 controller can be configured to report frame-copied errors to IBM LAN Network Manager software. These errors indicate that another host is responding to the MAC address of the 3174 controller. Both the 3174 and the IBM LAN Network Manager software can be configured to ignore frame-copied errors.

Monitor and Maintain the SRB Network

You can display a variety of information about the source-route bridging network. To display the information you require, perform one or more of the following tasks in EXEC mode:

Task	Command
Display internal state information about the Token Ring interfaces in the system.	**show controllers token**
Provide high-level statistics about the state of source bridging for a particular interface.	**show interfaces**
Display all currently configured bridges and all parameters that are related to the bridge as a whole and not to one of its interfaces.	**show lnm bridge**
Display the logical (multiport bridge) configuration of the router.	**show lnm config**
Display all LNM-relevant information about a specific interface.	**show lnm interface** *interface*

Task	Command
Display all LNM-relevant information about a specific router ring number.	**show lnm ring** *number*
Display all LNM-relevant information about a specific station or about all known stations on the ring.	**show lnm station** *address*
Show the current state of any current Local Acknowledgment for both LLC2 and SDLLC connections.	**show local-ack**
Display the contents of the NetBIOS cache.	**show netbios-cache**
Display the contents of the RIF cache.	**show netbios-cache**
Display the current source bridge configuration and miscellaneous statistics.	**show source-bridge**

To maintain the source-route bridged network, you can perform the following tasks:

- Clear the entries of all dynamically learned NetBIOS names
- Clear the entire RIF cache
- Limit the size of the backup queue

To maintain the source-routed bridged network, perform one of the following tasks in EXEC mode:

Task	Command
Clear the entries of all dynamically learned NetBIOS names.	**clear netbios-cache**
Clear the entire RIF cache.	**clear rif-cache**
Clear the source-bridge statistical counters	**clear source-bridge**

In addition to the EXEC mode tasks to maintain the source-routed bridged network, you can perform the following task in interface configuration mode:

Task	Command
Limit the size of the backup queue for remote source-route bridging to control the number of packets that can wait for transmission to a remote ring before they start being thrown away.	**source-bridge tcp-queue-max** *number*

SRB Configuration Examples

This section provides example configurations that you can use as a guide to configuring your source-route bridging environment. It includes examples of the following configuration types:

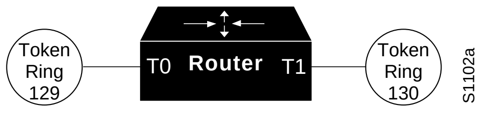
- Example of basic SRB with spanning explorers (page 21-48)
- Example of SRB only (page 21-48)
- Example of SRB and routing certain protocols (page 21-49)
- Example of multiport SRB (page 21-49)
- Example of SRB with multiple virtual ring groups (page 21-50)
- Example of RSRB using IP encapsulation over a TCP connection (page 21-51)
- Example of RSRB using IP encapsulation over an FST connection (page 21-13)
- Example of RSRB using all types of transport methods (page 21-54)
- Example of RSRB with local acknowledgment (page 21-55)
- Example of RSRB with local acknowledgment and passthrough (page 21-59)
- Example of local acknowledgment for LLC2 (page 21-62)
- Example of IP for load sharing over RSRB (page 21-64)
- Example of adding a static RIF cache entry (page 21-66)
- Example of adding a static RIF cache for a two-hop path (page 21-66)
- Example of SR/TLB for a simple network (page 21-67)
- Example of SR/TLB with access filtering (page 21-69)
- Example of NetBIOS support with a static NetBIOS cache entry (page 21-70)
- Example of LNM for a simple network (page 21-70)
- Example of LNM for a more complex network (page 21-71)
- Example of NetBIOS access filters (page 21-72)
- Example of filtering bridged Token Ring packets for IBM machines (page 21-73)
- Example of administrative access filters filtering SNAP frames on output (page 21-74)
- Example of creating access expressions (page 21-76)
- Example of access expressions (page 21-77)
- Example of configuring priority for locally terminated Token Ring interfaces in RSRB (page 21-77)
- Example of fast switching (page 21-79)
- Example of autonomous switching (page 21-80)
- Example of SNA traffic by LU address (page 21-80)

Example of Basic SRB with Spanning Explorers

Figure 1-14 illustrates a simple two-port bridge configuration. The example that follows routes IP, but source-route bridges all other protocols using spanning explorers.

```
!
interface tokenring 0
ip address 131.108.129.2 255.255.255.0
source-bridge 129 1 130
source-bridge spanning
multiring all
!
interface tokenring 1
ip address 131.108.130.2 255.255.255.0
source-bridge 130 1 129
source-bridge spanning
! use RIFs, as necessary, with IP routing software
multiring all
 !
```

Figure 1-14 Dual Port Source-Route Bridge Configuration



Token Rings 129 and 130 are connected through the router/bridge.

Example of SRB Only

This example, including part of a configuration file, shows that all protocols are bridged, including IP. Because IP is being bridged, the system has only one IP address.

```
!
no ip routing
!
interface TokenRing 0
ip address 131.108.129.2 255.255.255.0
source-bridge 129 1 130
source-bridge spanning
!
interface TokenRing 1
ip address 131.108.129.2 255.255.255.0
source-bridge 130 1 129
source-bridge spanning
!
interface Ethernet 0
ip address 131.108.129.2 255.255.255.0
!
```

Example of SRB and Routing Certain Protocols

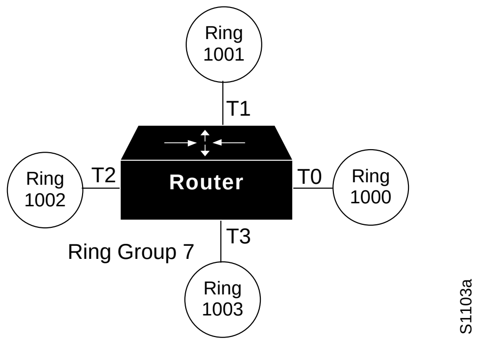
In this configuration, IP, XNS, and IPX are being routed, while all other protocols will be bridged between rings. While not strictly necessary, the Novell and XNS network numbers are set consistently with the IP subnetwork numbers. This makes the network easier to maintain.

```
!
xns routing 0000.0C00.02C3
!
novell routing 0000.0C00.02C3
!
interface TokenRing 0
ip address 131.108.129.2 255.255.255.0
xns network 129
novell network 129
source-bridge 129 1 130
source-bridge spanning
multiring all
!
interface TokenRing 1
ip address 131.108.130.2 255.255.255.0
xns network 130
novell network 130
source-bridge 130 1 129
source-bridge spanning
multiring all
!
interface Ethernet 0
ip address 131.108.2.68 255.255.255.0
xns network 2
novell network 2
!
```

Example of Multiport SRB

Figure 1-15 shows an example configuration of a four-port Token Ring source-route bridge.

Figure 1-15 Four-Port Source-Route Bridge



Rings 1000, 1001, 1002, and 1003 are all source-route bridged to each other across ring group 7.

```
 !
source-bridge ring-group 7
!
interface tokenring 0
source-bridge 1000 1 7
source-bridge spanning
!
interface tokenring 1
source-bridge 1001 1 7
source-bridge spanning
!
interface tokenring 2
source-bridge 1002 1 7
source-bridge spanning
!
interface tokenring 3
source-bridge 1003 1 7
source-bridge spanning
!
```

Example of Source-Route Bridging with Multiple Virtual Ring Groups

Two virtual ring groups can only be connected through an actual Token Ring. Figure 1-16 shows Virtual Rings 100 and 200 connected through Token Ring 3.

Figure 1-16 Two Virtual Rings Connected by an Actual Token Ring

Configuration for Router A

```
source-bridge ring-group 100
!
interface tokenring 0
source-bridge 3 1 100
source-bridge spanning
interface tokenring 1
source-bridge 1 1 100
source-bridge spanning
!
```

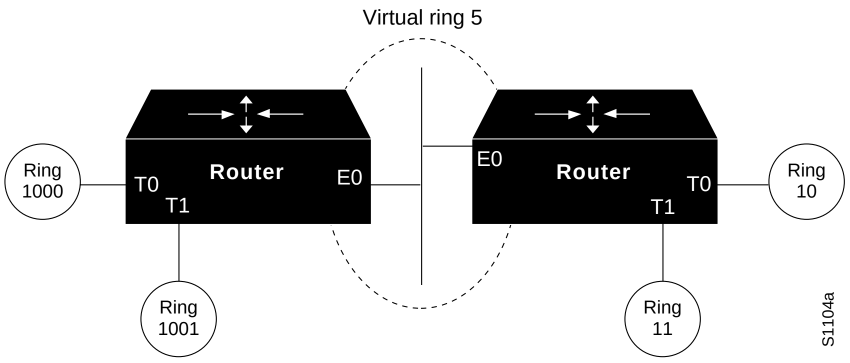
Configuration for Router B

```
!
source-bridge ring-group 200
!
interface tokenring0
source-bridge 3 1 200
source-bridge spanning
interface tokenring 2
source-bridge 2 1 200
source-bridge spanning
!
```

Example of RSRB Using IP Encapsulation over a TCP Connection

Figure 1-17 illustrates a configuration of two router/bridges configured for remote source-route bridging using TCP as a transport. Each router has two Token Rings. They are connected by an Ethernet segment over which the source-route bridged traffic will pass. The first router configuration is a source-route bridge at address 131.108.2.29. These peers are both running Release 9.0, so the **version** keyword has been specified on the **remote-peer** statements.

Figure 1-17 Remote Source-Route Bridging Using TCP as a Transport



Using TCP as the transport, the configuration for the source-route bridge at address 131.108.2.29 as depicted in Figure 1-17 is as follows:

```
!
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 131.108.2.29 version 2
source-bridge remote-peer 5 tcp 131.108.1.27 version 2
!
interface ethernet 0
ip address 131.108.4.4 255.255.255.0
!
interface tokenring 0
ip address 131.108.2.29 255.255.255.0
source-bridge 1000 1 5
source-bridge spanning
!
interface tokenring 1
ip address 131.108.128.1 255.255.255.0
source-bridge 1001 1 5
source-bridge spanning
!
```

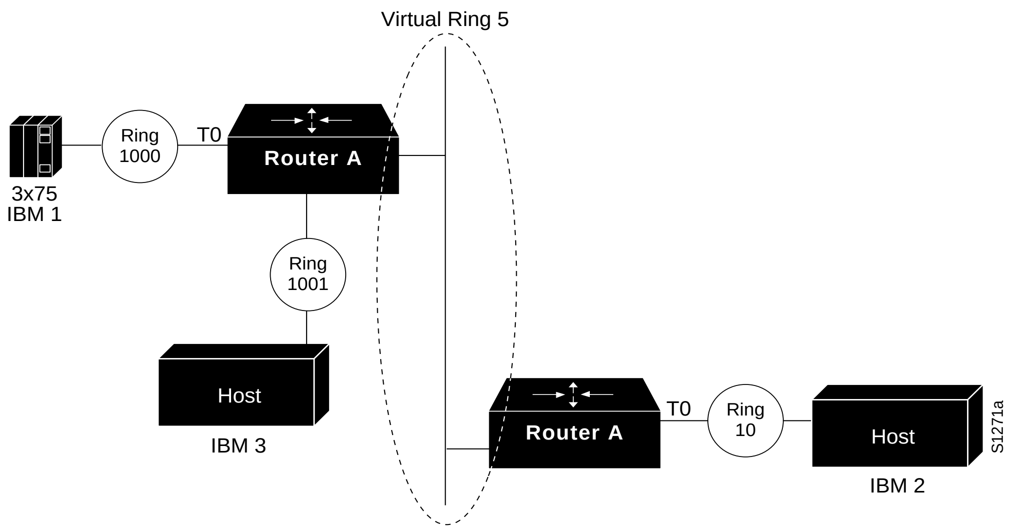
The configuration of the source-route bridge at 131.108.1.27 is as follows:

```
!
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 131.108.2.29
source-bridge remote-peer 5 tcp 131.108.1.27
!
interface ethernet 0
ip address 131.108.4.5 255.255.255.0
!
interface tokenring 0
ip address 131.108.1.27 255.255.255.0
source-bridge 10 1 5
source-bridge spanning
!
interface tokenring 1
ip address 131.108.131.1 255.255.255.0
source-bridge 11 1 5
source-bridge spanning
!
```

Example of RSRB Using IP Encapsulation Over an FST Connection

Figure 1-18 shows two routers connecting IBM hosts on Token Rings through an Ethernet backbone.

Figure 1-18 Remote Source-Route Bridging Using FST as a Transport



This example configuration enables IP encapsulation over an FST connection. In this configuration, the **source-bridge fst-peername** global configuration command is used to provide an IP address for the local router, the **source-bridge ring-group** global configuration command is used to define a ring group, and the **source-bridge remote peer** command with the **fst** option is used to associate the remote peer's IP address with the router's ring group and specify the remote peer's remote source-route bridging protocol version number. Because all FST peers support version 2 RSRB, the **version** keyword is always specified.

The configuration of the source-route bridge at 131.108.2.29 is as follows:

```
!
source-bridge fst-peername 131.108.2.29
source-bridge ring-group 5
source-bridge remote-peer 5 fst 131.108.1.27 version 2
!
interface ethernet 0
ip address 131.108.4.4 255.255.255.0
!
interface tokenring 0
ip address 131.108.2.29 255.255.255.0
source-bridge 1000 1 5
source-bridge spanning
!
interface tokenring 1
ip address 131.108.128.1 255.255.255.0
source-bridge 1001 1 5
source-bridge spanning
!
```

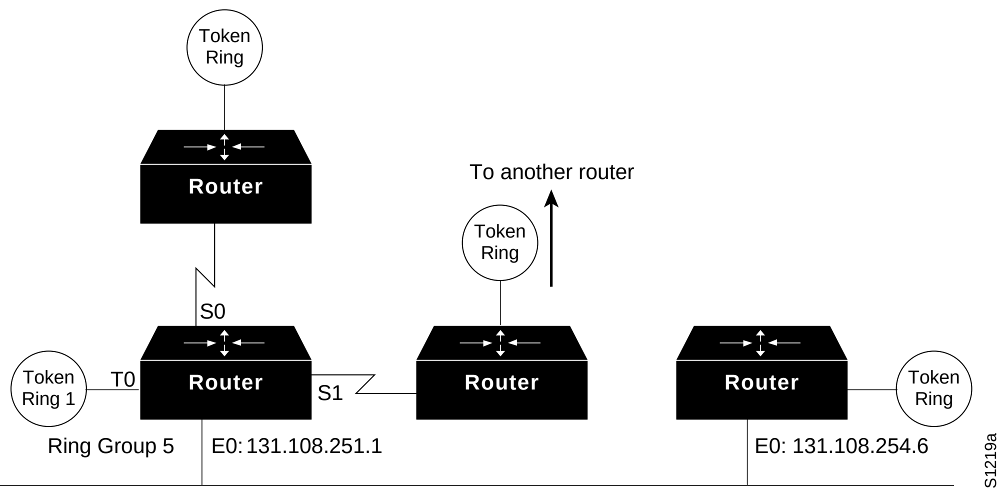
The configuration of the source-route bridge at 131.108.1.27 is as follows:

```
!
source-bridge fst-peername 131.108.1.27
source-bridge ring-group 5
source-bridge remote-peer 5 fst 131.108.2.29 version 2
!
interface ethernet 0
ip address 131.108.4.5 255.255.255.0
!
interface tokenring 0
ip address 131.108.1.27 255.255.255.0
source-bridge 10 1 5
source-bridge spanning
!
interface tokenring 1
ip address 131.108.131.1 255.255.255.0
source-bridge 11 1 5
source-bridge spanning
!
```

Example of RSRB Using All Types of Transport Methods

Figure 1-19 shows a router/bridge configured for remote source-route bridging using all types of transport methods.

Figure 1-19 Remote Source-Route Bridge Using All Types of Transport Methods



The configuration for the network in Figure 1-19 would be as follows:

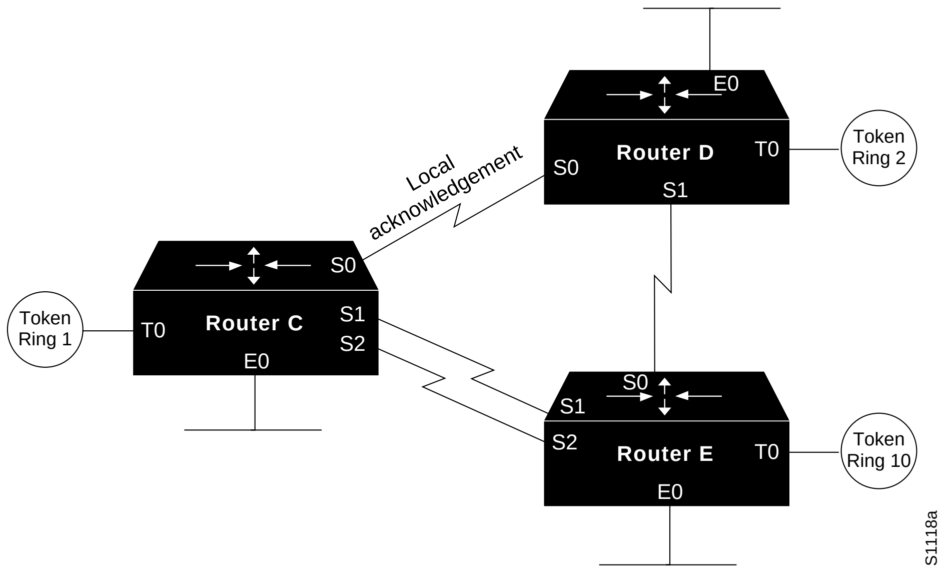
```
!
source-bridge fst-peername 131.108.251.1
source-bridge ring-group 5
source-bridge remote-peer 5 interface serial0
source-bridge remote-peer 5 interface serial1
source-bridge remote-peer 5 interface Ethernet0 0000.0c00.1234
source-bridge remote-peer 5 tcp 131.108.251.1
source-bridge remote-peer 5 fst 131.108.252.4
source-bridge remote-peer 5 tcp 131.108.253.5
!
interface tokenring 0
source-bridge 1 1 5
source-bridge spanning
!
interface ethernet 0
ip address 131.108.251.1 255.255.255.0
!
```

Note The two peers using the serial transport method will only function correctly if there are router/bridges at the other end of the serial line that have been configured to use the serial transport. The peers also must belong to the same ring group.

Example of RSRB with Local Acknowledgment

In Figure 1-20, a triangular configuration is used to provide the maximum reliability with minimal cost. In addition, one of the links is doubled to gain better bandwidth. In addition to IP and source-route bridging traffic, AppleTalk is also being routed between all the sites. Note that in this configuration, all the sessions between Router C and Router D are locally acknowledged. All the sessions between Router C and Router E are not locally acknowledged and are configured for normal remote source-route bridging. This example shows that not every peer must be locally acknowledged, but rather, that Local Acknowledgment can be turned on or off at the customer's discretion.

Figure 1-20 RSRB with Local Acknowledgment—Less Complex Configuration



The configuration files that enable this configuration follow.

Configuration for Router C

```
!
appletalk routing
!
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 132.21.1.1
source-bridge remote-peer 5 tcp 132.21.2.6 local-ack
source-bridge remote-peer 5 tcp 132.21.10.200
!
interface TokenRing 0
ip address 132.21.1.1 255.255.255.0
source-bridge 1 1 5
source-bridge spanning
multiring all
!
interface Ethernet 0
ip address 132.21.4.25 255.255.255.0
appletalk address 4.25
appletalk zone Twilight
!
interface Serial 0
ip address 132.21.16.1 255.255.255.0
appletalk address 16.1
appletalk zone Twilight
!
interface Serial 1
ip address 132.21.17.1 255.255.255.0
appletalk address 17.1
appletalk zone Twilight
!
interface Serial 2
ip address 132.21.18.1 255.255.255.0
appletalk address 18.1
appletalk zone Twilight
!
router igrp 109
network 132.21.0.0
!
hostname RouterC
!
```

Configuration for Router D

```
appletalk routing
!
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 132.21.1.1 local-ack
source-bridge remote-peer 5 tcp 132.21.2.6
source-bridge remote-peer 5 tcp 132.21.10.200
!
interface TokenRing 0
ip address 132.21.2.6 255.255.255.0
source-bridge 2 1 5
source-bridge spanning
multiring all
!
interface Ethernet 0
ip address 132.21.5.1 255.255.255.0
appletalk address 5.1
appletalk zone Twilight
!
interface Serial 0
ip address 132.21.16.2 255.255.255.0
appletalk address 16.2
appletalk zone Twilight
!
interface Serial 1
ip address 132.21.19.1 255.255.255.0
appletalk address 19.1
appletalk zone Twilight
!
router igrp 109
network 132.21.0.0
!
hostname RouterD
!
```

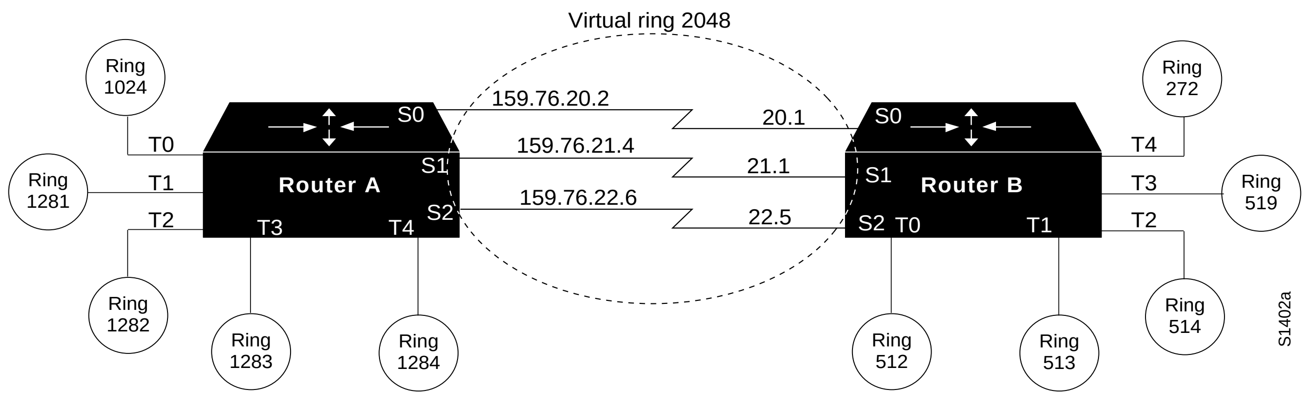
Configuration for Router E

```
!
appletalk routing
!
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 132.21.1.1
source-bridge remote-peer 5 tcp 132.21.2.6
source-bridge remote-peer 5 tcp 132.21.10.200
!
interface TokenRing 0
ip address 132.21.10.200 255.255.255.0
source-bridge 10 1 5
source-bridge spanning
multiring all
!
interface Ethernet 0
ip address 132.21.7.1 255.255.255.0
appletalk address 7.1
appletalk zone Twilight
!
interface Serial 0
ip address 132.21.19.2 255.255.255.0
appletalk address 19.2
appletalk zone Twilight
!
interface Serial 1
ip address 132.21.17.2 255.255.255.0
appletalk address 17.2
appletalk zone Twilight
!
interface Serial 2
ip address 132.21.18.2 255.255.255.0
appletalk address 18.2
appletalk zone Twilight
!
router igrp 109
network 132.21.0.0
!
hostname RouterE
!
```

Example of RSRB with Local Acknowledgment and Passthrough

Figure 1-21 shows two routers configured for remote source-route bridging over the three serial lines that connect these routers. In turn, five Token Rings connect to each of these routers.

Figure 1-21 Network Topology for RSRB with Local Acknowledgment and Passthrough



Configuration for Router A

```
!
source-bridge ring-group 2048
source-bridge remote-peer 2048 tcp 159.76.1.250 local-ack version 2
source-bridge remote-peer 2048 tcp 159.76.7.250 version 2
source-bridge passthrough 1281
source-bridge passthrough 1282
source-bridge passthrough 1283
source-bridge passthrough 1284
!
interface tokenring 0
ip address 159.76.7.250 255.255.255.0
llc2 ack-max 1
llc2 t1-time 1800
llc2 idle-time 29000
llc2 ack-delay-time 5
source-bridge 1024 1 2048
source-bridge spanning
early-token-release
multiring all
!
interface tokenring 1
ip address 159.76.8.250 255.255.255.0
clns-speed 4
clns mtu 464
source-bridge 1281 1 2048
source-bridge spanning
multiring all
!
interface tokenring 2
ip address 159.76.8.250255.255.255.0
shutdown
ring-speed 4
clns mtu 4464
source-bridge 1262 1 2046
source-bridge spanning
multiring all
```

```
!
interface tokenring 3
ip address 159.76.10.250 255.255.255.0
shutdown
ring speed 4
clns mtu 4464
source-bridge 1283 1 2048
source-bridge spanning
multiring all
!
interface tokenring 4
ip address 159.78.11.250 255.255.255.0
shutdown
ring speed 4
clns mtu 4464
source-bridge 1284 1 2048
source-bridge spanning
multiring all
!
interface serial 0
ip address 159.76.6.2 255.255.255.0
!
interface serial 1
ip address 159.76.6.4 255.255.255.0
!
interface serial 2
ip address 159.76.6.6 255.255.255.0
shutdown
!
interface serial 3
no ip address
shutdown
!
```

Configuration for Router B

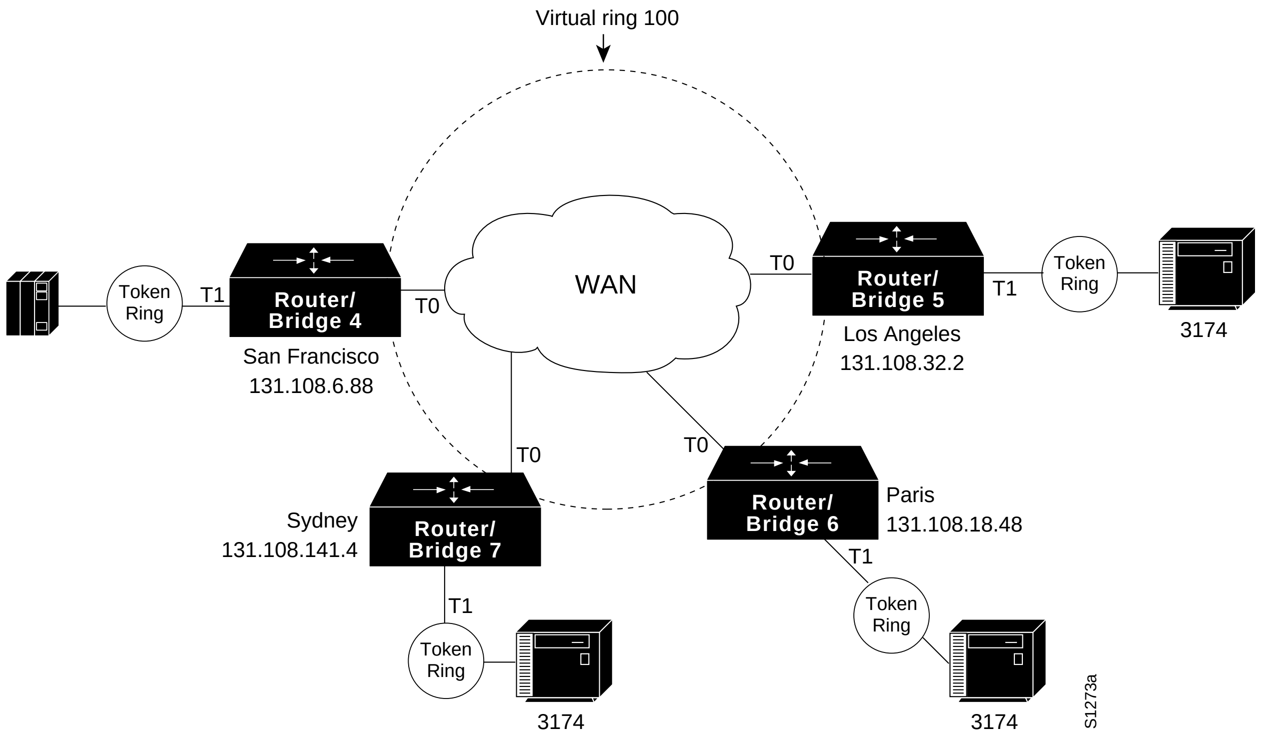
```
!
source-bridge ring-group 2048
source-bridge remote-peer 2048 tcp 159.76.1.250 version 2
source-bridge remote-peer 2048 tcp 159.76.7.250 local-ack version 2
!
interface tokenring 0
ip address 159.76.1.250 255.255.255.0
llc2 ack-max 2
llc2 t1-time 1900
llc2 idle-time 29000
llc2 ack-delay-time 5
source-bridge 512 1 2048
source-bridge spanning
early-token-release
multiring all
!
interface tokenring 1
ip address 159.76.2.250 255.255.255.0
ring-speed 16
clns mtu 8136
!
source-bridge 513 1 2048
source-bridge spanning
early-token-release
multiring all
```

```
!
interface tokenring 2
ip address 159.76.3.250 255.255.255.0
ring speed 16
clns mtu 8136
source-bridge 514 1 2048
source-bridge spanning
early-token-release
multiring all
!
interface tokenring 3
ip address 159.76.5.250 255.255.255.0
ring-speed 4
clns mtu 4464
source-bridge 519 2 2043
source-bridge spanning
multiring all
!
interface tokenring 4
ip address 159.76.107.250 255.255.255.0
ring-speed 4
clns mtu 4464
source-bridge 772 2 2048
source-bridge spanning
multiring all
!
interface tokenring
!
interface serial 0
ip address 159.76.6.1 255.255.255.0
!
interface serial 1
ip address 159.76.6.3 255.255.255.0
!
interface serial 2
ip address 159.76.6.5 255.255.255.0
!
interface serial 3
no ip address
shutdown
!
```

Example of Local Acknowledgment for LLC2

Figure 1-22 shows an IBM FEP located in San Francisco communicating with 3174 hosts in Sydney, Paris, and Los Angeles.

Figure 1-22 Remote Source-Route Bridging Using Local Acknowledgment—More Complex Example



In Figure 1-22, the session between the FEP and the 3174 system in Los Angeles is not locally terminated, because the distance is great enough to cause timeouts on the line. However, the sessions to Paris and Sydney are locally terminated.

Configuration for Router/Bridge 4 in San Francisco:

```
 source-bridge ring-group 100
!use direct encapsulation across serial link to Los Angeles
 source-bridge remote-peer 100 direct 131.108.32.2
 ! use fast sequenced transport with local termination to Paris
 source-bridge remote-peer 100 fst 131.108.18.48 local-ack
 ! use tcp encapsulation with local termination to Sydney
 source-bridge remote-peer 100 tcp 131.108.141.4 local-ack
 !
 interface tokenring 0
  ! source ring 1, bridge 4, destination ring 100
 source-bridge 1 4 100
 ! receive up to seven frames before sending an acknowledgment
 llc2 ack-max 7
 ! allow a 30 msec delay before I-frames must be acknowledged
 llc2 ack-delay-time 30
 !
 interface tokenring 1
 ! source ring 100, bridge 4, destination ring 1
 source-bridge 100 4 1
```

Configuration for Router/Bridge 7 in Sydney:

```
source-bridge ring-group 100
! use tcp encapsulation with local termination from Sydney
source-bridge remote-peer 100 tcp 131.108.6.88 local-ack
interface tokenring 0
! source ring 1, bridge 7, destination ring 100
source-bridge 1 7 100
! receive up to seven frames before sending an acknowledgment
llc2 ack-max 7
! allow a 30 msec delay before I-frames must be acknowledged
llc2 ack-delay-time 30
!
interface tokenring 1
! source ring 100, bridge 7, destination ring 1
source-bridge 100 7 1
```

Configuration for Router/Bridge 6 in Paris:

```
source-bridge ring-group 100
! use fast sequenced transport with local termination from Paris
source-bridge remote-peer 100 fst 131.108.6.88 local-ack
interface tokenring 0
! source ring 1, bridge 6, destination ring 100
source-bridge 1 6 100
! receive up to seven frames before sending an acknowledgment
llc2 ack-max 7
! allow a 30 msec delay before I-frames must be acknowledged
llc2 ack-delay-time 30
!
interface tokenring 1
! source ring 100, bridge 6, destination ring 1
source-bridge 100 6 1
```

Configuration for Router/Bridge 5 in Los Angeles:

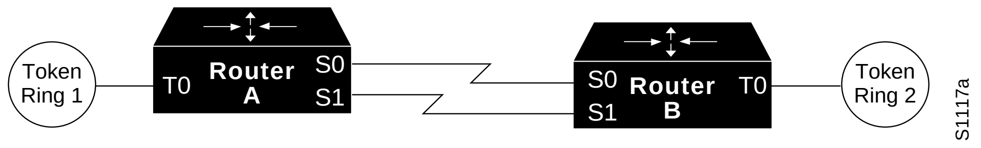
```
source-bridge ring-group 100
! use direct encapsulation across serial link from Los Angeles
source-bridge remote-peer 100 direct 131.108.6.88

interface tokenring 0
! source ring 1, bridge 5, destination ring 100
source-bridge 1 5 100
! receive up to seven frames before sending an acknowledgment
llc2 ack-max 7
! allow a 30 msec delay before I-frames must be acknowledged
llc2 ack-delay-time 30
!
interface tokenring 1
! source ring 100, bridge 5, destination ring 1
source-bridge 100 5 1
```

Note Both peers need to be configured for LLC2 Local Acknowledgment. If only one is so configured, unpredictable (and undesirable) results will occur.

Example of IP for Load Sharing Over RSRB

As Figure 1-23 shows, two routers are connected by two serial lines. Each has been configured as a basic remote two-port bridge, but extended to include both reliability and IP load sharing. When both serial lines are up, traffic is split between them, effectively combining the bandwidth of the connections. If either one of the serial lines goes down, all traffic is routed to the remaining line with no disruption. This happens transparently with respect to the end connections, unlike other source-route bridges that would abort those connections.

Figure 1-23 RSRB—Simple Reliability



The sample configuration files that enable this configuration follow.

Configuration for Router/Bridge A:

```
!
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 204.31.7.1
source-bridge remote-peer 5 tcp 204.31.8.1
!
interface TokenRing 0
ip address 204.31.7.1 255.255.255.0
source-bridge 1 1 5
source-bridge spanning
multiring all
!
interface Serial 0
ip address 204.31.9.1 255.255.255.0
!
interface Serial 1
ip address 204.31.10.1 255.255.255.0
!
router igrp 109
network 204.31.7.0
network 204.31.9.0
network 204.31.10.0
!
hostname RouterA
!
```

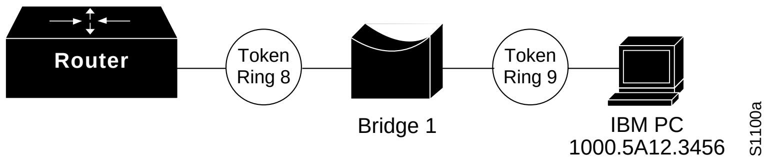
Configuration for Router/Bridge B:

```
!
source-bridge ring-group 5
source-bridge remote-peer 5 tcp 204.31.7.1
source-bridge remote-peer 5 tcp 204.31.8.1
!
interface TokenRing 0
ip address 204.31.8.1 255.255.255.0
source-bridge 2 1 5
source-bridge spanning
multiring all
!
interface Serial 0
ip address 204.31.9.2 255.255.255.0
!
interface Serial 1
ip address 204.31.10.2 255.255.255.0
!
router igrp 109
network 204.31.8.0
network 204.31.9.0
network 204.31.10.0
!
hostname RouterB
!
```

Example of Adding a Static RIF Cache Entry

In this example configuration, the path between rings 8 and 9 connected via source-route bridge 1 is described by the route descriptor 0081.0090. A full RIF, including the route control field, would be 0630.0081.0090. The static RIF entry would be submitted to the leftmost router as shown in Figure 1-24.

Figure 1-24 Assigning a RIF to a Source-Route Bridge

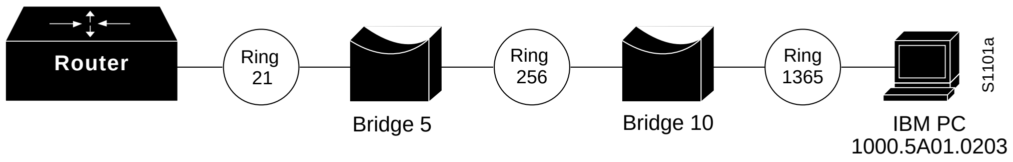


```
rif 1000.5A12.3456 0630.0081.0090
!
```

Example of Adding a Static RIF Cache Entry for a Two-Hop Path

In Figure 1-25, assume that a datagram was sent from a router/bridge on ring 21 (15 hexadecimal), across bridge 5 to ring 256 (100 hexadecimal), and then across bridge 10 (A hexadecimal) to ring 1365 (555 hexadecimal) for delivery to a destination host on that ring.

Figure 1-25 Assigning a RIF to a Two-Hop Path



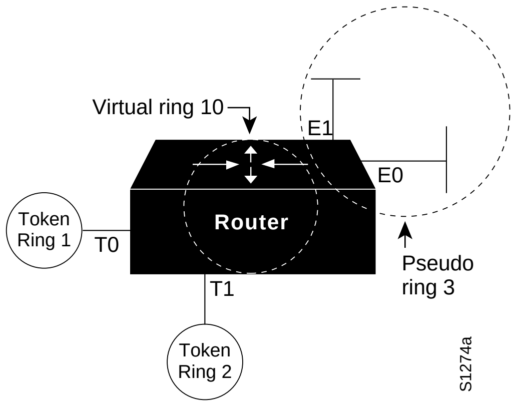
The RIF in the leftmost router describing this two-hop path is 0830.0155.100a.5550 and is entered as follows:

```
!
rif 1000.5A01.0203 0830.0155.100a.5550
!
```

Example of SR/TLB for a Simple Network

In the simple example illustrated in Figure 1-26, a four-port router with two Ethernets and two Token Rings is used to connect transparent bridging on the Ethernets to source-route bridging on the Token Rings.

Figure 1-26 Example of a Simple SR/TLB Configuration



Assume that the following configuration for source-route bridging and transparent bridging existed before you wanted to enable SR/TLB:

```
!
interface tokenring 0
source-bridge 1 1 2
!
interface tokenring 1
source-bridge 2 1 1
!
interface Ethernet 0
bridge-group 1
!
interface Ethernet 0
bridge-group 1
!
bridge 1 protocol dec
```

In order to enable SR/TLB, one aspect of this configuration must change immediately—a third ring must be configured. Before SR/TLB, the two Token Ring interfaces were communicating with two-port local source-route bridging; after SR/TLB, these two interfaces must be reconfigured to communicate through a virtual ring, as follows:

```
!
source-bridge ring-group 10
!
interface tokenring 0
source-bridge 1 1 10
!
interface tokenring 1
source-bridge 2 1 10
!
interface ethernet 0
bridge-group 1
!
interface ethernet 1
bridge-group 1
!
bridge 1 protocol dec
```

Now you are ready to determine two things:

- A ring number for the pseudo-ring that is unique throughout the source-route bridged network. For the preceding example configuration, use a 3.
- A bridge number for the path to the pseudo-ring. For the preceding example configuration, use a 1.

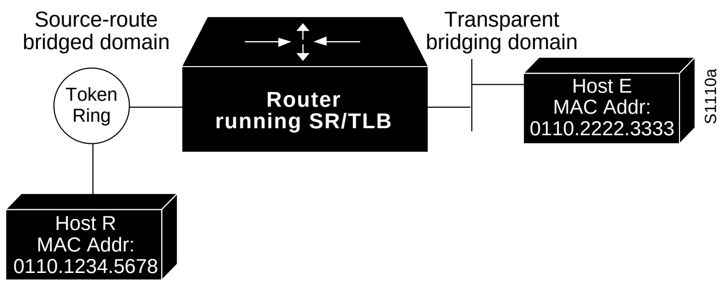
Once you have determined the ring number and the bridge number, you can add the **source-bridge transparent** command to the file, including these two values as parameters for the command. The partial configuration below includes this **source-bridge transparent** entry:

```
!
source-bridge ring-group 10
source-bridge transparent 10 3 1 1
!
interface tokenring 0
source-bridge 1 1 10
!
interface tokenring 1
source-bridge 2 1 10
!
interface ethernet 0
bridge-group 1
!
interface ethernet 1
bridge-group 1
!
bridge 1 protocol dec
```

Example of SR/TLB with Access Filtering

In the example shown in Figure 1-27, you want to connect only a single machine, Host E, on an Ethernet to a single machine, Host R, on the Token Ring.

Figure 1-27 Example of a Bit-Swapped Address


Source-route bridged domain
Transparent bridging domain
Token Ring
Router running SR/TLB
Host E MAC Addr: 0110.2222.3333
Host R MAC Addr: 0110.1234.5678
S1110a

You want to allow only these two machines to communicate across the router. Therefore, you might create the following configuration to restrict the access. However, this configuration will not work, for the last reason stated in the notes above.

Note For the sake of readability, the commands to control the bridging are not shown here, just the commands to control the filtering.

```
interface tokenring 0
access-expression output smac(701)
!
interface ethernet 0
bridge-group 1 input-address-list 701
!
access-list 701 permit 0110.2222.3333
!
```

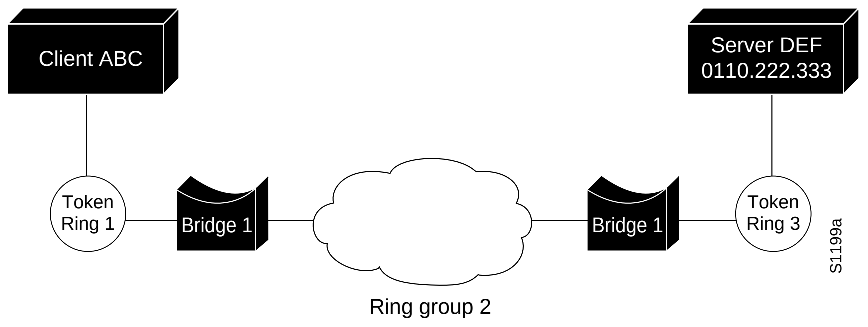
The command for the Token Ring interface specifies that the access-list 701 be applied on the source address of frames going out to the Token Ring, and the command for the Ethernet interface specifies that this access list be applied on the source address frames entering the interface from Ethernet. This would work if both interfaces used the same bit ordering, but Token Rings and Ethernets use opposite (swapped) bit orderings in their addresses in relationship to each other. Therefore, the address of Host E on the Token Ring is not 0110.2222.3333, but rather 8008.4444.cccc, resulting in the following configuration. The following configuration is better. This example shows that access lists for Token Ring and Ethernet should be kept completely separate from each other.

```
!
interface tokenring 0
source-bridge input-address-list 702
!
interface ethernet 0
bridge-group 1 input-address-list 701
!
access-list 701 permit 0110.2222.3333
!
access-list 702 permit 0110.1234.5678
!
```

Example of NetBIOS Support with a Static NetBIOS Cache Entry

Figure 1-28 shows a NetBIOS client on a Token Ring connected through a cloud to a NetBIOS server on another Token Ring.

Figure 1-28 Specifying a Static Entry



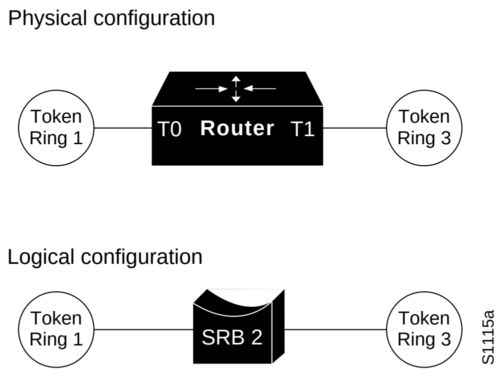
In Figure 1-28, a static entry is created in the router attached to ring 1 on the client side of the ring group. The static entry is to the server DEF, which is reached through the router attached to ring 3. If server DEF has the MAC address 0110.2222.3333, the configuration for the static entry on the client side is as follows:

```
rif 0110.2222.3333 0630.0021.0030 ring-group 2
netbios name-cache 0110.2222.3333 DEF ring-group 2
```

Example of LNM for a Simple Network

Figure 1-29 shows a router with two Token Rings configured as a local source-route bridge.

Figure 1-29 Router with Two Token Rings Configured as a Local Source-Route Bridge



The associated configuration file follows:

```
!
interface TokenRing 0
source-bridge 1 2 3
!
interface TokenRing 1
source-bridge 3 2 1
!
```

The **show lnm config** command displays the logical configuration of this bridge, including the LNM configuration information that needs to be entered at the LNM Station. A sample **show lnm config** display follows.

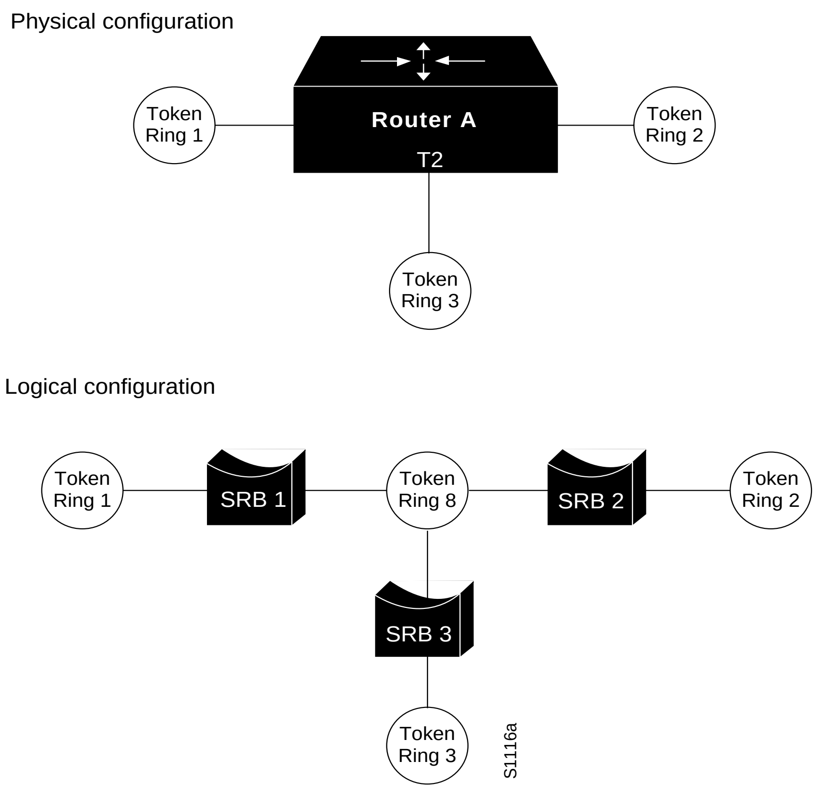
```
Doomgiver# show lnm config

Bridge(s) currently configured:
From     ring 001, address 0000.3000.abc4
Across bridge 002
To       ring 003, address 0000.3000.5735
```

In this example, the MAC addresses 0000.3000.abc4 and 000.3000.5735 must be configured as Adapter Addresses at the LNM Station.

Example of LNM for a More Complex Network

Figure 1-30 shows a router with three Token Rings configured as a multiport bridge, thus employing the concept of the virtual ring.

Figure 1-30 Router with Three Token Rings Configured as a Multiport Bridge



The associated configuration file follows.

```
 !
 source-bridge ring-group 8
!
interface TokenRing 0
source-bridge 1 1 8
!
interface TokenRing 1
source-bridge 2 2 8
!
interface TokenRing 2
source-bridge 3 3 8
!
```

The **show lnm config** command displays the logical configuration of this bridge, including all the pertinent information for configuring this router into LNM.

```
Doomgiver# show lnm config
Bridge(s) currently configured:

            From     ring 001, address 0000.0028.abcd
            Across bridge 001
            To       ring 008, address 4000.0028.abcd

            From     ring 002, address 0000.3000.abc4
            Across bridge 002
            To       ring 008, address 4000.3000.abc4

            From     ring 003, address 0000.3000.5735
            Across bridge 003
            To       ring 008, address 4000.3000.5735
```

In this example, six station definitions must be entered at the LNM Station, one for each of the MAC addresses listed in this sample **show lnm config** display.

Example of NetBIOS Access Filters

The following command permits packets that include the Station name ABCD to pass through the router, but denies passage to packets that do not include the Station name ABCD.

```
!
netbios access-list host marketing permit ABCD
!
```

This command specifies a prefix where the pattern matches any name beginning with the characters DEFG. Note that the string DEFG itself is included in this condition.

```
!
netbios access-list host marketing deny DEFG*
!
```

This command permits any station name with the letter W as the first character and the letter Y as the third character in the name. The second and fourth letters in the name can be any character. This example would allow stations named WXYZ and WAYB; however, stations named WY and WXY would not be included in this statement, because the ? must match some specific character in the name.

```
!
netbios access-list host marketing permit W?Y?
!
```

This example illustrates how to combine wildcard characters:

```
!
netbios access-list host marketing deny AC?*
!
```

The command specifies that the marketing list deny any name beginning with AC that is at least three characters in length (the ? would match any third character). The string ACBD and ACB would match, but the string AC would not.

This command removes the entire marketing NetBIOS access list.

```
!
no netbios access-list host marketing
!
```

To remove single entries from the list, use a command such as the following:

```
!
no netbios access-list host marketing deny AC?*
!
```

This example removes only the list that filters station names with the letters AC at the beginning of the name.

Keep in mind that the access lists are scanned in order. In this example, the first list denies all entries beginning with the letters ABC, including one named ABCD. This voids the second command, because the entry permitting a name with ABCD comes after the entry denying it.

```
!
netbios access-list host marketing deny ABC*
netbios access-list host marketing permit ABCD
!
```

Example of Filtering Bridged Token Ring Packets to IBM Machines

The example in Figure 1-31 disallows the bridging of Token Ring packets to all IBM workstations on Token Ring 1.

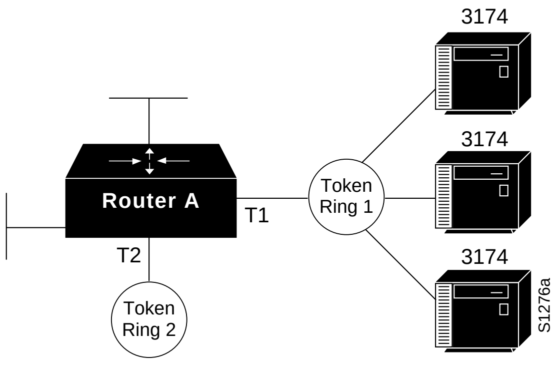


Figure 1-31 Router Filtering Bridged Token Ring Packets to IBM Machines

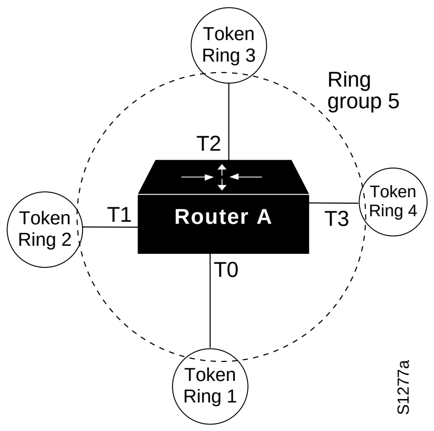
This example assumes that all hosts on Token Ring 1 have Token Ring addresses with the vendor code 1000.5A00.0000. The first line of the access list denies access to all IBM workstations, while the second line permits everything else. Then, the access list is assigned to the input side of Token Ring 1.

```
! deny access to all IBM workstations
access-list 700 deny 1000.5A00.0000   8000.00FF.FFFF
! permit all other traffic
access-list 700 permit 0000.0000.0000   FFFF.FFFF.FFFF
!
interface token ring 1
! apply access list 700 to the input side of Token Ring 1
source-bridge input-address-list 700
!
```

Example of Administrative Access Filters—Filtering SNAP Frames on Output

Figure 1-32 shows a router connecting four Token Rings.

Figure 1-32 Router Filtering SNAP Frames on Output



The following example allows only AppleTalk Phase 2 packets to be source-route bridged between Token Rings 0 and 1, and allows Novell packets only to be source-route bridged between Token Rings 2 and 3.

```
source-bridge ring-group 5
!
interface tokenring 0
ip address 131.108.1.1 255.255.255.0
source-bridge 1000 1 5
source-bridge spanning
source-bridge input-type-list 202
!
interface tokenring 1
ip address 131.108.11.1 255.255.255.0
source-bridge 1001 1 5
source-bridge spanning
source-bridge input-type-list 202
!
interface tokenring 2
ip address 131.108.101.1 255.255.255.0
source-bridge 1002 1 5
source-bridge spanning
source-bridge input-lsap-list 203
!
interface tokenring 3
ip address 131.108.111.1 255.255.255.0
source-bridge 1003 1 5
source-bridge spanning
source-bridge input-lsap-list 203
!
! SNAP type code filtering
! permit ATp2 data (0x809B)
! permit ATp2 AARP (0x80F3)
access-list 202 permit 0x809B 0x0000
access-list 202 permit 0x80F3 0x0000
access-list 202 deny 0x0000 0xFFFF
!
! LSAP filtering
! permit IPX (0xE0E0)
access-list 203 permit 0xE0E0 0x0101
access-list 203 deny 0x0000 0xFFFF
```

Note that it is not necessary to check for an LSAP of 0xAAAA when filtering SNAP-encapsulated AppleTalk packets, because for source-route bridging, the use of type filters implies SNAP encapsulation.

Example of Creating Access Expressions

In math, you have the following:

3 * 4 + 2 = 14 but 3 * (4 + 2) = 18

Similarly, the following access expressions would return TRUE if lsap(201) and dmac(701) returned TRUE or if smac(702) returned TRUE:

lsap(201) & dmac(701) | smac(702)

However, the following access expression would return TRUE only if lsap(201) returned TRUE and either of dmac(701) or smac(702) returned TRUE:

lsap(201) & (dmac(701) | smac(702))

Referring to the earlier example, “An Example Using NetBIOS Access Filters,” we had the phrase:

“Pass the frame if it is NetBIOS, or if it is an SNA frame destined to address 0110.2222.3333.”

This phrase was converted to the simpler form of:

Pass if “NetBIOS or (SNA and destined to 0110.2222.3333).”

So, for the following configuration:

```
! Access list 201 passes NetBIOS frames (command or response)
access-list 201 permit 0xF0F0 0x0001
!
access-list 202 permit 0x0404 0x0001 ! Permits SNA frames (command or response)
access-list 202 permit 0x0004 0x0001 ! Permits SNA Explorers with NULL DSAP
!
! Access list 701 will permit the FEP MAC address
! of 0110.2222.3333
access-list 701 permit 0110.2222.3333
!
```

The following access expression would result:

```
access-expression in lsap(201) | (lsap(202) & dmac(701))
```

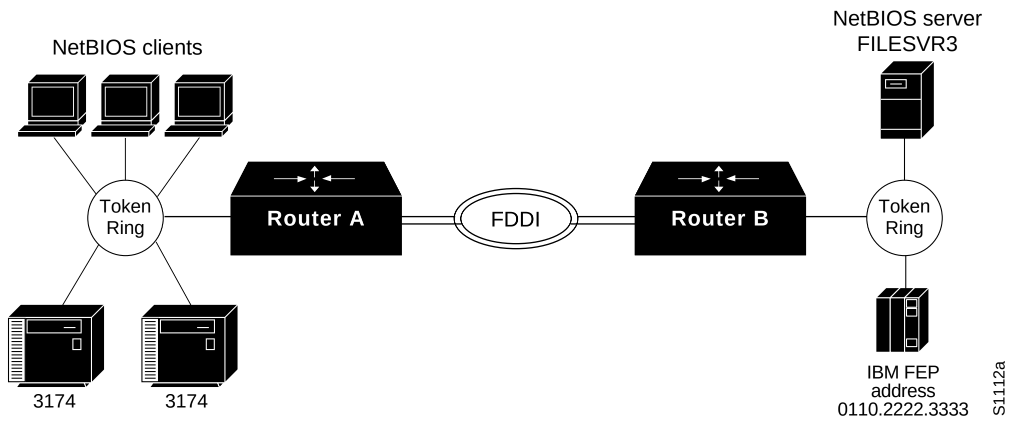
Therefore, the full configuration example is as follows:

```
!
interface tokenring 0
access-expression in lsap(201 | (lsap(202) & dmac(701))
!
! Access list 201 passes NetBIOS frames (command or response
access-list 201 permit 0xF0F0 0x0001
!
access-list 202 permit 0x0404 0x0001 ! Permits SNA frames (command or response)
access-list 202 permit 0x0004 0x0001 ! Permits NSA Explorers with NULL DSAP
!
! Access list 701 will permit the FEP MAC address
! of 0110.2222.3333
access-list 701 permit 0110.2222.3333
!
```

Example of Access Expressions

Figure 1-33 shows two routers connecting two Token Rings to an FDDI backbone.

Figure 1-33 Network Configuration Using NetBIOS Access Filters



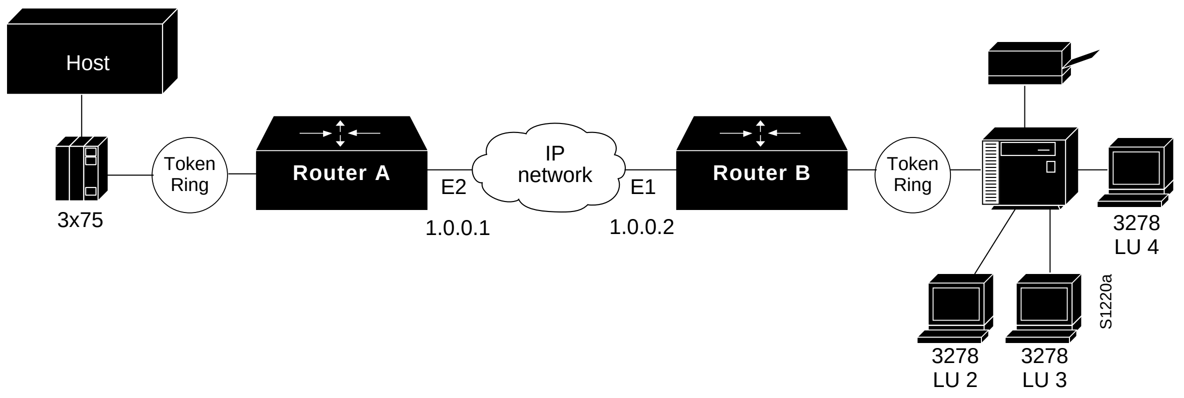
Suppose you want to permit the IBM 3174s to access the FEP at address 0110.2222.3333, and also want the NetBIOS clients to access the NetBIOS server named FILESVR3. The following set of router configuration commands would meet this need:

```
netbios access-list host MIS permit FILESVR3
netbios access-list host MIS deny *
!
access-list 202 permit 0x0404 0x0001 ! Permits SNA frames (command or response)
access-list 202 permit 0x0004 0x0001 ! Permits SNA Explorers with NULL DSAP
!
access-list 701 permit 0110.2222.3333
!
interface tokenring 0
access-expression in (lsap(202) & dmac(701)) | netbios-host(MIS)
```

Example of Configuring Priority for Locally Terminated Token Ring Interfaces in RSRB

Figure 1-34 shows a network that uses remote source-route bridging to bridge Token Ring traffic.

Figure 1-34 RSRB Configuration Example



The configuration for the network shown in Figure 1-34 follows.

Configuration for Router/Bridge A

```
source-bridge ring-group 2624
source-bridge remote-peer 2624 tcp 1.0.0.1
source-bridge remote-peer 2624 tcp 1.0.0.2 local-ack priority
!
interface TokenRing 0
source-bridge 2576 8 2624
source-bridge spanning
multiring all
locaddr-priority 1
priority-group 1
!
interface Ethernet 0
ip address 1.0.0.1 255.255.255.0
!
locaddr-priority-list 1 02 high
locaddr-priority-list 1 03 high
locaddr-priority-list 1 04 medium
locaddr-priority-list 1 05 low
!
priority-list protocol ip high tcp 1996
priority-list protocol ip medium tcp 1987
priority-list protocol ip normal tcp 1988
priority-list protocol ip low tcp 1989
```

Configuration for Router/Bridge B

```
source-bridge ring-group 2624
source-bridge remote-peer 2624 tcp 1.0.0.2
source-bridge remote-peer 2624 tcp 1.0.0.1 local-ack priority
!
interface TokenRing 0
source-bridge 2626 8 2624
source-bridge spanning
multiring all
locaddr-priority 1
priority-group 1
!
interface Ethernet 0
ip address 1.0.0.2 255.255.255.0
!
locaddr-priority-list 1 02 high
locaddr-priority-list 1 03 high
locaddr-priority-list 1 04 medium
locaddr-priority-list 1 05 low
!
priority-list protocol ip high tcp 1996
priority-list protocol ip medium tcp 1987
priority-list protocol ip normal tcp 1988
priority-list protocol ip low tcp 1989
```

Example of Fast Switching

This example disables fast switching between two Token Ring interfaces in the same router/bridge:

```
! global command establishing the ring group for the interface configuration commands
source-bridge ring-group 2
!
! commands that follow apply to interface token 0
interface token 0
! enable srb between local ring 1, bridge 1, and target ring 2
source-bridge 1 1 2
!disable source-route fast-switching cache on interface token 0
no source-bridge route-cache
!
interface token 1
! enable srb between local ring 2, bridge 1, and target ring 1
source-bridge 2 1 1
no source-bridge route-cache
```

Frames entering interface token 0 or token 1 will not be fast-switched to the other interface.

Example of Autonomous Switching

The following example enables use of autonomous switching between two ciscoBus Token Ring interfaces in the same router/bridge:

```
! global command to apply interface configuration commands to the ring group
source-bridge ring-group 2
!
! commands that follow apply to interface token 0
interface token 0
! enable srb between local ring 1, bridge 1, and target ring 2
source-bridge 1 1 2
! enable autonomous switching for interface token 0
source-bridge route-cache cbus
!
interface token 1
! enable srb between local ring 2, bridge 1, and target ring 1
source-bridge 2 1 1
source-bridge route-cache cbus
```

Frames entering interface token 0 or token 1 will be autonomously switched to the other interface.

Example of SNA Traffic Prioritization by LU Address

The following example enables SNA traffic prioritization by LU address:

```
locaddr-priority-list 1 01 medium
locaddr-priority-list 1 02 normal
locaddr-priority-list 1 03 low
locaddr-priority-list 1 04 high
!
priority-list 2 protocol ip low tcp 1996
priority-list 2 protocol ip high tcp 1987
priority-list 2 protocol ip medium tcp 1988
priority-list 2 protocol ip normal tcp 1989
!
interface tokenring 0
source-bridge 123
locaddr-priority 1

interface serial 0
priority-group 2
```