Distributed Routing Software

Bridging Configuration Guide

Part Number: AA–QL29B–TE

December 1995

This manual includes information on bridging methods, operational features of bridging, configuration methods and basic configurations, and monitoring of your bridging software.

Revision/Update Information:	This is a revised manual.
Software Version:	Distributed Routing Software V1.0A

This manual was produced by Shared Engineering Services.

Contents

Preface

1 Bridging Basics

Bridging Overview . . . 1–1
Bridges vs. Routers . . . 1–2
  Router Connections . . . 1–3
  Bridge Connections . . . 1–3
Types of Bridges . . . 1–4
  Simple Bridges . . . 1–4
  Complex Bridges . . . 1–5
  Local Bridges . . . 1–5
  Remote Bridges . . . 1–5
Basic Bridge Operation . . . 1–6
  Operation Example 1: Local Bridge Connecting Two LANS . . . 1–6
  Operation Example 2: Remote Bridging Over a Serial Link . . . 1–6
  Data Encapsulation Scheme . . . 1–7
MAC Bridge Frame Formats . . . 1–8
    CSMA/CD (Ethernet) MAC Frames . . . 1–9
    Token ring MAC Frames . . . 1–10

2 Bridging Methods

Transparent Bridging (STB) . . . 2–1

Routers and Transparent Bridges 2–2
Network Requirements 2–2
Transparent Bridge Operation 2–3
Shaping the Spanning Tree 2–4
Spanning Tree Bridges and Ethernet Packet Format Translation 2–7
Transparent Bridge Terminology and Concepts 2–8
Source Routing Bridging (SRB) 2–13
Source Routing Bridge Operation 2–14
Source Routing Frames 2–15
The Spanning Tree Explore Option 2–19
Simulating a Spanning Tree Network 2–20
Balancing Traffic Loads 2–20
Protocol Filtering 2–21
Source Routing Bridge Terminology and Concepts 2–22
Source Routing Transparent Bridge (SRT) 2–24
General Description 2–25
Source Routing Transparent Bridge Operation and Architecture 2–26
Source Routing Transparent Bridge Terminology 2–27
Adaptive Source Routing Transparent Bridge (ASRT)
(SR-TB Conversion) 2–28
General Description 2–28
Source Routing – Transparent Bridge Operation 2–29
Specific Source Routing and Transparent Bridging Operations . . . 2–30
SR-TB Bridging: Four Examples 2–34
Example 1: Frame Sent from End Station A to End Station B . . . 2–35
Example 2: Frame Sent from End Station A to End Station C . . . 2–35
Example 3: Frame Sent from End Station C to End Station D . . . 2–36
Example 4: Frame Sent from End Station C to End Station A . . . 2–36
Source Routing – Transparent Bridge Terminology and Concepts . . . 2–37
ASRT Bridge Overview 2–39
Transparent-Source Routing Compatibility – Issues and Solutions . . 2–39
Elimination of Packet Size Problems 2–40
Bit Ordering in STB (802.3) and SRB (802.5) Bridges 2–40
ASRT Configuration Considerations 2–41

3 Bridging Features

Bridging Tunnel 3–1
Encapsulation and IP Routing Protocols 3–3
TCP/IP Host Services (Bridge-Only Management) 3–3
Bridge-MIB Support 3–4
NETBIOS Filtering 3–4
NETBIOS Filtering Using Host Names 3–5
NETBIOS Filtering Using Bytes 3–6
Building a Filter 3–7
Simple and Complex Filters 3–8
Pseudo Serial Ethernet 3–8
Multiple Spanning Tree Protocol Options 3–9
Background: Problems with Multiple Spanning Tree protocols 3–9
STP/8209 3–10
Logical Link Class 2 Support 3–11
Threading 3–12
IP Threading with ARP 3–12
DECnet (DNA) Threading 3–13
IPX Threading 3–14
AppleTalk 1 and 2 Threading 3–14

4 Configuring Bridging

Accessing the ASRT Configuration Environment 4–1
ASRT Configuration Commands 4–1
Destination Address Filtering Enabled For Entry 4–6
Output Port Map Created For Address Entry Having More Than One Input Port 4–7
Single Output Port Map Created All Incoming Ports Associated With Address Entry 4–8
LNM Configuration Commands 4–38
NETBIOS Filtering Configuration Commands 4–43
Update Byte-Filter-List Command Options 4–50
Update Name-Filter-List Command Options 4–52

Tunnel Configuration Commands 4–55
Tunneling and Multicast Packets 4–56

5 Basic Bridging Configurations

Accessing the ASRT Configuration Environment 5–1
Basic Bridging Configuration Procedures 5–1
Bridging Interfaces 5–2
Enabling the Transparent Bridge 5–3
Enabling the Source Routing Bridge 5–3
Enabling the SR-TB Bridge 5–3
ASRT Configuration Matrix 5–4
Basic Configuration Procedures for NETBIOS Filtering 5–6
Example 1: Creating a Host Name Filter 5–6
Example 2: Creating a Byte Filter 5–10
Basic LNM Configuration 5–14
Obtain the MAC Address Required For Network Manager Software . 5–14
Enable The LNM Agents on the Bridging Router 5–14
Check the Configuration By Displaying Enabled LNM Agents 5–15

6 Monitoring Bridging

Accessing the ASRT Console Environment 6–1
ASRT Console Commands 6–1
NETBIOS Filtering Console Commands 6–25

7 Configuring TCP/IP Host Services

Accessing the TCP/IP Host Configuration Environment 7–1
Basic Configuration Procedures 7–1
Setting the IP Address 7–1
Adding a Default Gateway 7–1
Enabling TCP/IP Host Services 7–2
TCP/IP Host Configuration Commands 7–2

8 Monitoring TCP/IP Host Services

Accessing the TCP/IP Host Console Environment 8–1
TCP/IP Host Console Commands 8–1

Index

Figures

1–1 Simple and Complex Bridging Configuration 1–2
1–2 Two Port Bridge Connecting Two LANs 1–6
1–3 Bridging Over a Point-to-Point Link 1–7
1–4 Data Encapsulation Over a Point-to-Point Link 1–7
1–5 Examples of MAC Frame Formats 1–9
2–1 Networked LANs Before Spanning Tree 2–5
2–2 Spanning Tree Created With Default Values 2–6
2–3 User-Adjusted Spanning Tree 2–7
2–4 Example of Source Routing Bridge Connectivity 2–13
2–5 802.5 Source Address Format 2–16
2–6 802.5 Routing Information Field 2–17
2–7 Example of Parallel Bridges 2–20
2–8 Using Spanning Tree Explore for Load Balancing 2–21
2–9 Bridge Instances Within a Bridge 2–23
2–10 SRT Bridge Operation 2–26
2–11 SR-TB Bridge Connecting Two Domains 2–30
2–12 SR-TB Bridging Examples 2–34
3–1 Example of the Bridge Tunnel Feature 3–2

Tables

2–1 Route/Bridge Decision Table 2–22
2–2 SR-TB Bridge Decision Table 2–32
4–1 ASRT Configuration Command Summary 4–2
4–2 LNM Configuration Commands 4–39

4–3 NETBIOS Filtering Configuration Commands 4–43
4–4 Tunnel Configuration Commands 4–56
5–1 Bridge Configuration Settings Matrix 5–5
6–1 ASRT Console Commands Summary 6–2
6–2 NETBIOS Filtering Console Commands Summary 6–25
7–1 TCP/IP Host Configuration Commands Summary 7–2
8–1 TCP/IP Host Console Commands Summary 8–2

Preface

Objectives

This guide contains the information you need to configure your bridging router. This manual includes information on bridging methods, operational features of bridging, configuration methods and basic configurations, and monitoring of your bridging software.

Audience

This guide is intended for persons who install and operate computer networks. Although experience with computer networking hardware and software is helpful, you do not need programming experience to use the protocol software.

Organization

This manual is organized as follows:

- Chapter 1 provides basic information about bridges and bridging operation.
- Chapter 2 provides conceptual information about the various bridging methods supported by the Adaptive Source Routing Transparent (ASRT) Bridge.
- Chapter 3 describes special features supported by the Adaptive Source Routing Transparent (ASRT) Bridge.
- Chapter 4 describes how to access the ASRT configuration process and how to use the bridging configuration commands.

- Chapter 5 describes how to create basic configurations for the Adaptive Source Routing Transparent (ASRT) Bridge using the ASRT configuration commands.
- Chapter 6 describes how to access the ASRT console process and how to use the bridging console commands.
- Chapter 7 describes how to access the Host Services (HST) configuration process and how to use the HST configuration commands.
- Chapter 8 describes how to access the Host Services (HST) console process and how to use the HST console commands.

Related Documentation

The following documents provide additional information about the router hardware and software:

- *Event Logging System Messages Guide*, AA–QL2AB–TE
- *Network Interface Operations Guide*, AA–QL2BB–TE
- *Routing Protocols Reference Guide*, AA–QL2CB–TE
- *Routing Protocols User's Guide*, AA–QL2DB–TE
- *System Software Guide*, AA–QL2EB–TE

Conventions Used in This Guide

`Special type`	This special type in examples indicates system output or user input.
`Boldface`	Boldface type in examples indicates user input.
lowercase-italics	Lowercase italics in command syntax or examples indicate variables for which either the user or the system supplies a value.
[ ]	Brackets enclose operands or symbols that are either optional or conditional. Specify the operand and value if you want the condition to apply. Do not type the brackets in the line of code.
$\boxed{\text{RET}}$	Indicates that you should press the Return key.
<u>underscore</u>	Characters underscored in a command listing represent the least number of characters you must enter to identify that command to the interpreter.

1

Bridging Basics

This chapter discusses basic information about bridges and bridging operation.

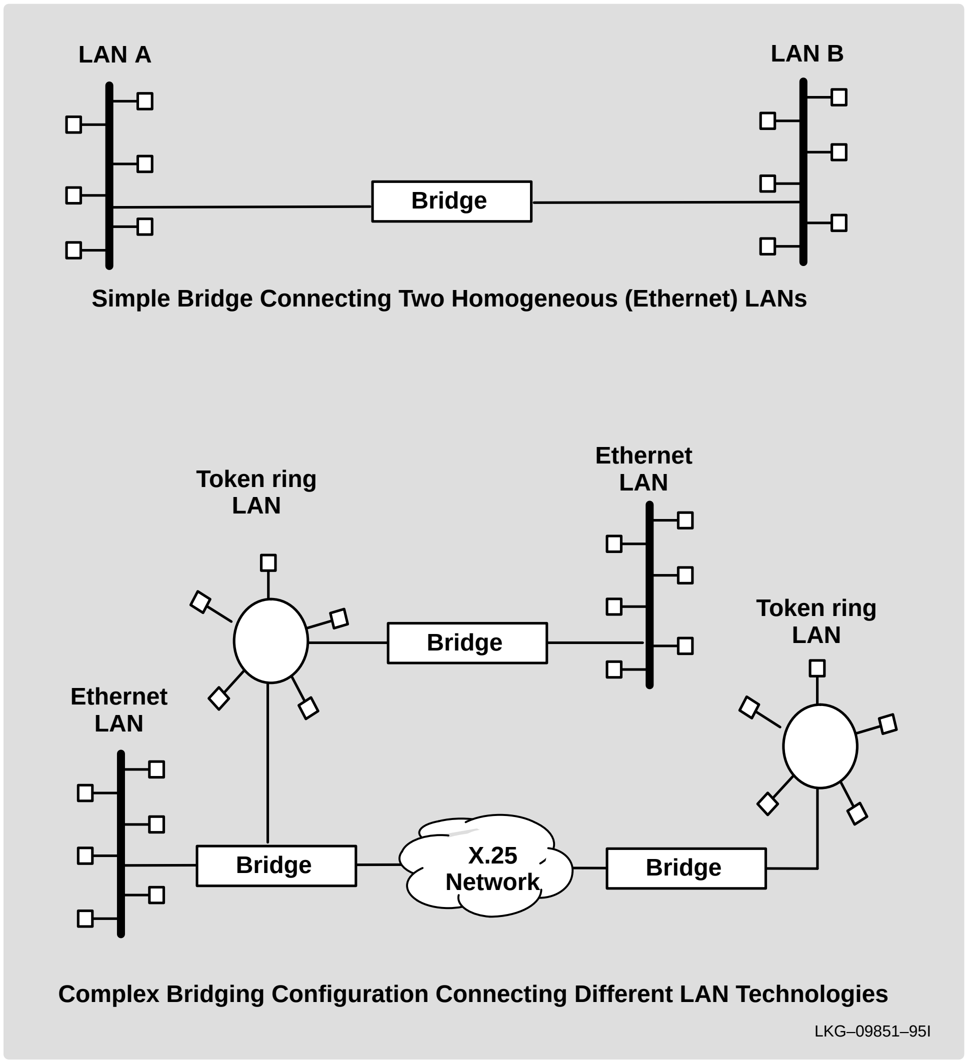
Bridging Overview

A bridge is a device that links two or more local area networks. The bridge accepts data frames from each connected network and then decides whether to forward each frame based on the MAC header contained in the frame. Bridges originally linked two or more homogeneous networks. The term *homogeneous* means that the connected networks use the same bridging method and media types. Examples of these are networks supporting the source routing bridging method *only* or transparent bridging algorithm *only* (these methods are explained later).

Current bridges also allow communication between non-homogeneous networks. *Non-homogeneous* refers to networks that may mix different bridging methods or different media types and may also offer more configuration options.

Figure 1–1 illustrates examples of simple and complex bridging configurations.

Figure 1–1 Simple and Complex Bridging Configuration


LAN A
LAN B
Bridge
Simple Bridge Connecting Two Homogeneous (Ethernet) LANs
Token ring LAN
Ethernet LAN
Bridge
Token ring LAN
Ethernet LAN
Bridge
X.25 Network
Bridge
Complex Bridging Configuration Connecting Different LAN Technologies
LKG–09851–95I

Bridges vs. Routers

Internetworking devices such as bridges and routers connect network segments. However, each device uses a different method to establish and maintain their LAN-to-LAN connections. Routers connect LANs at Layer 3 (Network Layer) of the OSI model while bridges connect LANs at Layer 2 (Link Layer).

Router Connections

Network layer protocols efficiently move information in large and diverse network configurations. Connecting at the Network Layer (Layer 3) with a router allows connectivity and path selection between end stations at large geographical distances. Routing protocols are then used to select the best path for connecting distant and diverse LANs. Because a large variety of network and subnetwork configurations exist in networks of this type, connecting LANs through the Network Layer is usually the preferred method.

Bridge Connections

Connecting a bridge at Layer 2 provides connectivity across a physical link. This connection is essentially "transparent" to the host connected on the network.

Note: Source routing bridges are not considered completely "transparent." See the "Bridging Methods" chapter in this guide for more information on source routing and transparent bridges.

The Link Layer maintains physical addressing schemes, line discipline, topology reporting, error notification, flow control, and ordered delivery of data frames. Isolation from upper layer protocols is one of the advantages of bridging. Because bridges function at the Link Layer, they are not concerned with looking at the protocol information that occurs at the upper layers. This results in lower processing overhead and faster communication of network layer protocol traffic. Because bridges are not concerned with Layer 3 information, they may also forward different types of protocol traffic (such as IP, DECnet, IPX) between two or more networks.

Bridges can also filter frames based on Layer 2 fields. This means that the bridge may be configured to accept and forward frames of a certain type or ones that originate from a particular network. This ability to configure filters is very useful for maintaining effective traffic flow.

Bridges are used to divide large networks into manageable segments. The advantages of bridging are summed up as follows:

- Bridging lets you transparently extend the connectivity between systems attached to LAN segments by forming a single *extended LAN*.

- Bridging lets you form an extended LAN from segments of different media types.
- Bridging increases the effective throughput capacity of the extended LAN by automatically restricting traffic to only those segments that need to see it.
- Configurable filters let you regulate the amount of traffic that is forwarded to a specific segment.
- Bridging partitions the extended LAN into separate collision domains for each segment, effectively increasing the throughput capacity. (A collision domain is a shared physical medium to which systems must contend for access.)
- Bridging allows interconnection between systems using nonroutable protocols such as 3 LAT when those systems are attached to different LAN segments.

Types of Bridges

The following sections describe specific types of bridges and how they can be classified by their hardware and software capabilities.

Simple Bridges

Figure 1–1 illustrates simple bridges consisting of two or more linked network interfaces connecting Local Area Networks (LANs). Bridges interconnect separate Local Area Networks by relaying data frames between the separate MAC (Media Access Control) entities of the bridged LANs.

The main functions of a simple bridge are summarized as follows:

- The bridge receives all frames transmitted on each of LAN A and LAN B.
- It forwards frames from the input LAN to the output LAN if the destination MAC Address of the frame is known to be attached to the output LAN (bridges can automatically learn on which LAN a MAC Address exists).

Complex Bridges

Complex bridges carry out more sophisticated functions than simple bridges. These functions include the bridge maintaining status information on the other bridges. This information includes the communication path cost as well as the number of hops required to reach each connected network. Periodic exchanges of information between bridges update all bridge information. This type of exchange permits dynamic routing between bridges.

Complex bridges can also modify frames and recognize and transmit packets from different LAN technologies (for example, Token ring and Ethernet). In this case the bridge is sometimes referred to as a *translational* bridge.

The Adaptive Source Routing Transparent (ASRT) Bridge is a software collection capable of several of the bridging options just described and more. These functions are explained in greater detail later in this chapter.

Local Bridges

Local bridges provide connections between several LAN segments in the same geographical area. An example of this would be a bridge used to the various LANs located in your company's main headquarters.

Remote Bridges

Remote bridges connect multiple LAN segments in different geographical areas. An example of this is bridges used to the connect LANs located in your company's main headquarters to LANs in other branch offices around the country. Because of the geographical differences, this configuration moves from a Local Area Network configuration to a Wide Area Network (WAN) configuration.

Remote bridges can differ from local bridges in several ways. One major difference is the data transmission speed. WAN connections are generally slower than LAN connections. This speed difference can be significant when running time-sensitive applications. Another difference is the physical connections between remote and local bridges and LANs. In local bridges, the connection is made through local cabling media (for example, Ethernet). Remote bridge connections are made over serial lines.

Basic Bridge Operation

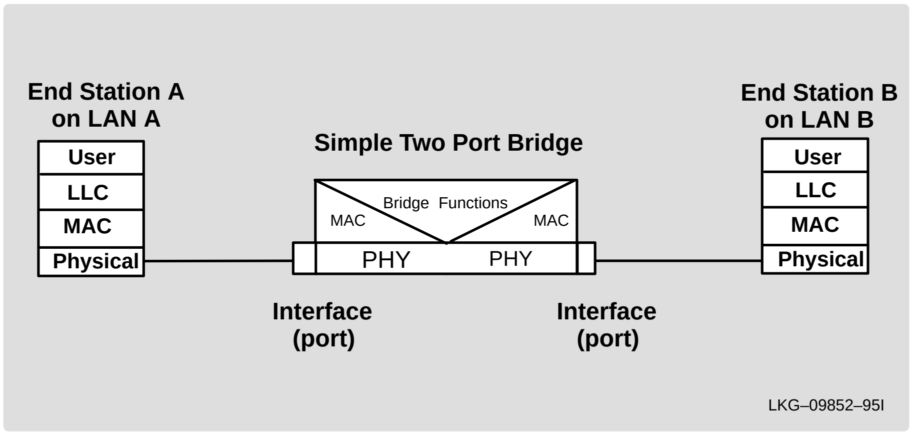
According to the IEEE 802 LAN standard, all station addresses are specified at the MAC level. At the Logical Link Control (LLC) level, only Service Access Point (SAP) addresses are designated. Therefore, the MAC level is the level at which the bridge functions. The following examples explain how bridging functions proceed at this level.

Operation Example 1: Local Bridge Connecting Two LANS

Figure 1–2 shows a two port bridge connecting end stations on two separate LANs. In this example, the local bridge connects LANs with identical LLC and MAC layers (two ethernet LANs). Conceptually, you can think of the bridge as a data link relay that forwards frames between the MAC sublayers and physical channels of the attached LANs, thus providing data link connectivity between them.

The bridge captures MAC frames whose destination addresses are not on the local LAN (the LAN connected to the interface receiving the transmitted frame). It then forwards them to the appropriate destination LAN.

Figure 1–2 Two Port Bridge Connecting Two LANs

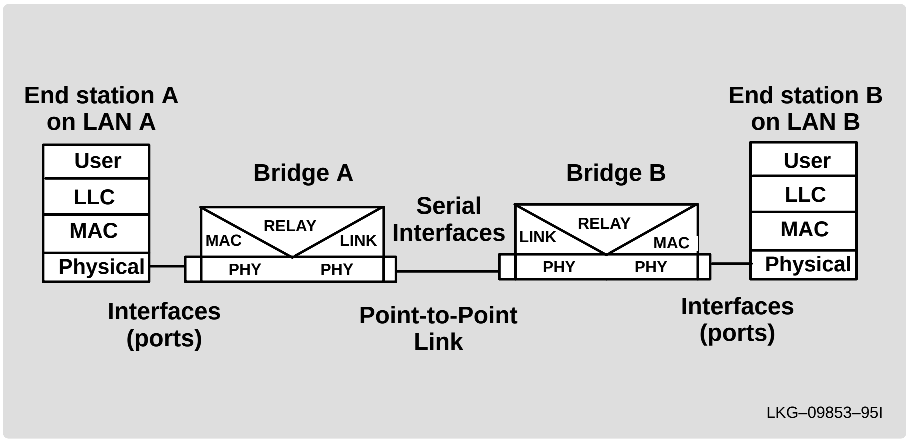


Operation Example 2: Remote Bridging Over a Serial Link

Figure 1–3 shows a pair of bridges connected over a serial link. These remote bridges connect LANs with identical LLC and MAC layers (two ethernet LANs).

The bridge captures a MAC frame whose destination address is not on the local LAN and sends it to the appropriate destination LAN by way of the point-to-point link.

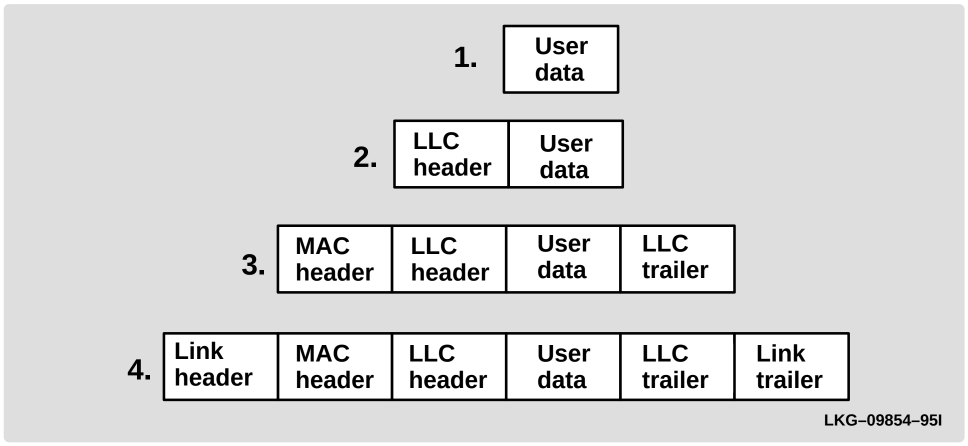
Figure 1–3 Bridging Over a Point-to-Point Link



Data Encapsulation Scheme

Data is encapsulated as the bridges communicate data over the serial link. Figure 1–4 illustrates the encapsulation process.

Figure 1–4 Data Encapsulation Over a Point-to-Point Link



Encapsulation proceeds as follows:

1. End station A provides data to its LLC.
2. LLC appends a header and passes the resulting data unit to the MAC level.
3. MAC then appends a header (3) and trailer to form a MAC frame. Bridge A captures the frame.
4. Bridge A does not strip off the MAC fields because its function is to relay the intact MAC frame to the destination LAN. In the point-to-point configuration, the bridge appends a link layer (for example, PPP and HDLC) header and trailer and transmits the point-to-point frame across the link.

When the data frame reaches its target, Bridge B, the link fields are stripped off and Bridge B transmits the *original, unchanged* MAC frame to its destination, end station B.

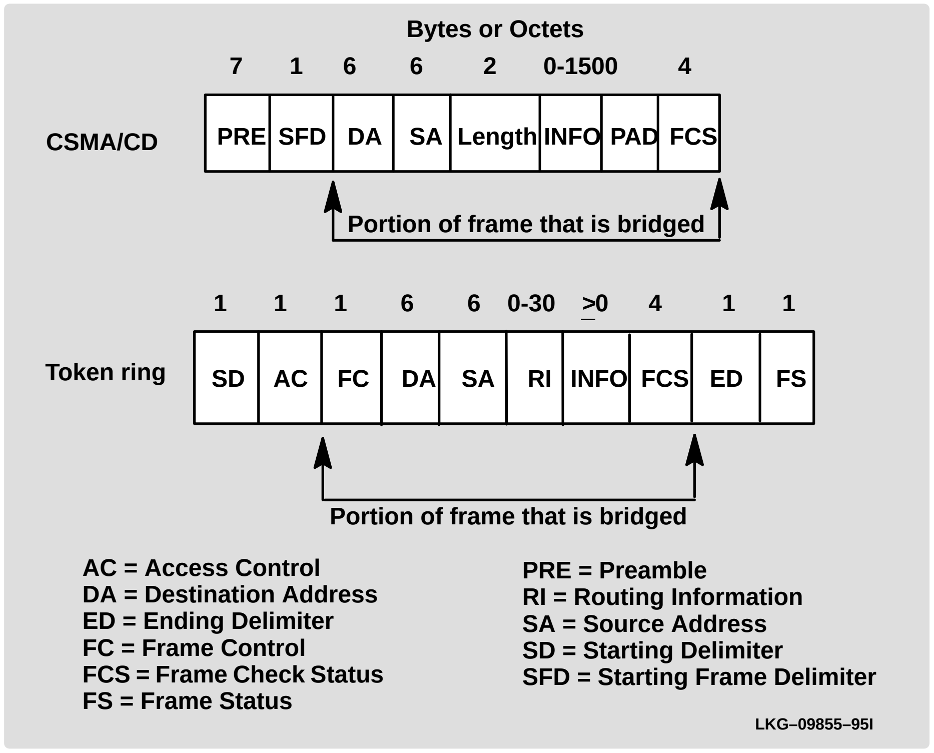
MAC Bridge Frame Formats

Bridges interconnect LANs by relaying data frames, specifically MAC frames, between the separate MAC entities of the bridged LANs. MAC frames provide the necessary “Where?” information for frame forwarding in the form of source and destination addresses.

Figure 1–5 shows the CSMA/CD and Token ring MAC frame formats supported by the bridges. The specific frames are detailed in the following section.

Note: A separate frame format is used at the LLC level. This frame is then embedded in the appropriate MAC frame.

Figure 1–5 Examples of MAC Frame Formats



CSMA/CD (Ethernet) MAC Frames

The following information describes each of the fields found in CSMA/CD (Ethernet) MAC frames:

- **Preamble (PRE)** – 7-byte pattern used by the receiving end station to establish bit synchronization and then locate the first bit of the frame.
- **Start Frame Delimiter (SFD)** – Indicates the start of the frame.

The portion of the frame that is actually bridged is consists of the following fields:

- **Destination Address (DA)** – Specifies the end station for which the frame is intended. This address may be a unique physical address (one destination), a multicast address (a group of end stations as a destination), or a global address (all stations as the destination). The format is either 16- or 48-bit (2 or 6 octets) and must be the same for all stations on that particular LAN.

- **Source Address (SA)** – Specifies the end station that transmitted the frame. The format must be the same as the destination address format.

- **Length** – Specifies the number of LLC bytes that follow.

- **Info (INFO)** – Embedded fields created at the LLC level that contain service access point information, control information, and user data.

- **Pad** – Sequence of bytes that ensures that the frame is long enough for proper collision detection (CD) operation.

- **Frame Check Sequence (FCS)** – 32-bit cyclic redundancy check value. This value is based on all fields, starting with the destination address.

Token ring MAC Frames

The following information describes each of the fields found in Token ring MAC frames:

- **Starting Delimiter (SD)** – Unique 8-bit pattern that indicates the start of the frame.

- **Access Control (AC)** – Field with the format PPPTMRRR where PPP and RRR are 3-bit priority and reservation variables, M is the monitor bit, and T indicates that this is either a token or a data frame. If it is a token frame, the only other field is the ending delimiter (ED).

- **Frame Control (FC)** – Indicates whether this is an LLC data frame. If it is not, bits in this field control operation of the Token ring MAC protocol.

The portion of the frame that is actually bridged consists of the following fields:

- **Destination Address (DA)** – Same as CSMA/CD and token bus.
- **Source Address (SA)** – Identifies the specific station that originates the frame. The length of the field may be either a 2- or 6-octet address.

  Both address lengths carry a Routing Information Indicator (RII) bit that indicates whether a Routing Information Field (RIF) is present in the frame. When the RII set to 1, it indicates that a Routing Information Field (RIF) is present after the source address. When the RII set to 0, it indicates that a Routing Information Field (RIF) is not present after the source address. This field is explained in more detail in the "Source Routing Bridge" section of this chapter.
- **Routing Information Field (RIF)** – When the RII in the source address field is set to 1, this field is present after the source address. The RIF is required for the source routing protocol. It consists of a 2-octet routing control field and a series of 2-octet route designator fields. This field is explained in more detail in the "Source Routing Bridge" section of this chapter.
- **Info (INFO***)* – Embedded fields created at the LLC level that contain service access point information, control information, and user data.
- **Frame Check Sequence (FCS)** – A 32-bit cyclic redundancy check value. This value is based on all fields, starting with the destination address.
- **End Delimiter (ED)** – Contains the error detection (E) bit, and the intermediate frame (I) bit. The I bit indicates that this is the frame other than the final one of a multiple frame transmission.
- **Frame Status (FS)** – Contains the address recognized (A) and frame copied (C) bits.

2 Bridging Methods

This chapter describes the methods of bridging supported by the Adaptive Source Routing Transparent (ASRT) Bridge. Each section gives an overview of a specific technology and is followed by a description of the data frames supported by that technology.

Transparent Bridging (STB)

The transparent bridge is also commonly known as a Spanning Tree Bridge (STB). The term *transparent* refers to the fact that the bridge silently forwards non-local traffic to attached LANs in a way that is transparent to or unseen by the originating and receiving systems. End station applications do not know about the presence of the bridge. The bridge learns about the presence of end stations by listening to traffic. From this listening process, it builds a database of addresses of end stations attached to its LANs.

For each frame it receives, the bridge checks the frame’s destination address against the ones in its database. If the frame’s destination is an end station on the same LAN, the frame is not forwarded. If the destination is on another LAN, the frame is forwarded. If the destination address is not present in the database, the frame is forwarded to all the LANs that are connected to the bridge (except the LAN from which it originated).

All transparent bridges use the Spanning Tree protocol and algorithm. The spanning tree algorithm produces and maintains a logical loop-free topology in a bridged network that contains loops in its physical design. In a mesh topology where more than one bridge is connected between two LANs, looping occurs. In such cases, data packets bounce back and forth between two LAN’s parallel bridges. This creates a redundancy in data traffic and produces the phenomenon known as looping.

When looping occurs, you must reconfigure the local or remote LAN, or both, to remove the physical loop. With spanning tree, a self-configuring algorithm allows a bridge to be added anywhere in the LAN without creating loops. Upon adding the new bridge, the spanning tree algorithm transparently configures all bridges on the LAN into a single loop-free spanning tree.

In order to eliminate data loops, the spanning tree only has one active path between two end stations. To create a loop-free topology, the algorithm determines which bridge ports can forward data and which ones are blocked. Spanning tree provides the following features:

- **Loop detection** – Detects and eliminates looping in extended LAN configurations.
- **Automatic backup of data paths** – The bridges connecting to the redundant paths enter backup mode automatically. When a primary path fails, a backup path becomes active.
- **User configurability** – Lets you tailor your network topology. Sometimes the default settings do not produce the desired network topology. You can adjust the bridge priority, port priority, and path cost parameters to determine the shape of the spanning tree.

Routers and Transparent Bridges

A system may be capable of performing both routing and bridging functions concurrently. In this mode, the router is called a *brouter.* During this mode of operation, the following occurs:

- Packets are routed if the packet’s protocol has been enabled for routing.
- Packets are filtered at the MAC layer if you configure specific bridging filters.
- Packets that are not routed or filtered are candidates for bridging, depending on the destination MAC (Media Access Control) address.

Network Requirements

Transparent Bridge implements a spanning tree bridge that conforms to the IEEE 802.1D standard. All Ethernet bridges on the network must be 802.1D spanning tree bridges. This spanning tree protocol is not compatible with bridges implementing Digital’s LANbridge 100 spanning tree protocol.

Transparent Bridge Operation

The spanning tree protocol specifies that all participating bridges in the network exchange HELLO Bridge Protocol Data Units (BPDUs) which provide configuration information about each bridge. BPDUs include information such as the bridge ID, root ID, and root path cost. This information helps the bridges to unanimously determine which bridge is the root bridge and which bridges are the designated bridges for LANs to which they are connected.

Of all the information exchanged in the HELLO messages, the following parameters are the most important for computing the spanning tree:

- **Root Bridge ID** – The root bridge ID is the bridge ID of the bridge. The root bridge is the designated bridge for all the LANs to which it is connected.

- **Root Path Cost** – The sum total of the path costs to the root through this bridge's root port. This information is transmitted by both the root bridge and the designated bridges to update all bridges on path information if the topology changes.

- **Bridge ID** – A unique ID used by the spanning tree algorithm to determine the spanning tree. Each bridge in the network is assigned a unique bridge identifier.

- **Port ID** – The ID of the port from which the current HELLO BPDU message was transmitted.

With this information available, the spanning tree begins to determine its shape and direction and then creates a logical path configuration. This process can be summarized as follows:

1. A root bridge for the network is selected by comparing the bridge IDs of each bridge in the network. The bridge with the highest priority ID (lowest value) wins.

2. The spanning tree algorithm then selects a designated bridge for each LAN. If more than one bridge is connected to the same LAN, the bridge with the smallest path cost to the root is selected as the designated bridge. In the case of equal path costs, the bridge with the lowest bridge ID value is selected as the designated bridge.

3. The non-designated bridges on the LANs put each port that has not been selected as a root port into a BLOCKED state. In the BLOCKED state a bridge still listens to HELLO BPDUs so that it can act on any changes that are made in the network (for example, designated bridge failures) but it does not forward any data packets.

Through this process, the spanning tree algorithm reduces a bridged LAN network of arbitrary topology into a single spanning tree. With the spanning tree there is never more than one active data path between any two end stations, thus eliminating data loops.

This new configuration is bounded by a time factor. If a designated bridge fails or is physically removed, other bridges on the LAN detect the situation when they do not receive HELLO BPDUs within the time period set by the bridge maximum age time. This event triggers a new configuration process where another bridge is selected as the designated bridge. A new configuration is also created if the root bridge fails.

Shaping the Spanning Tree

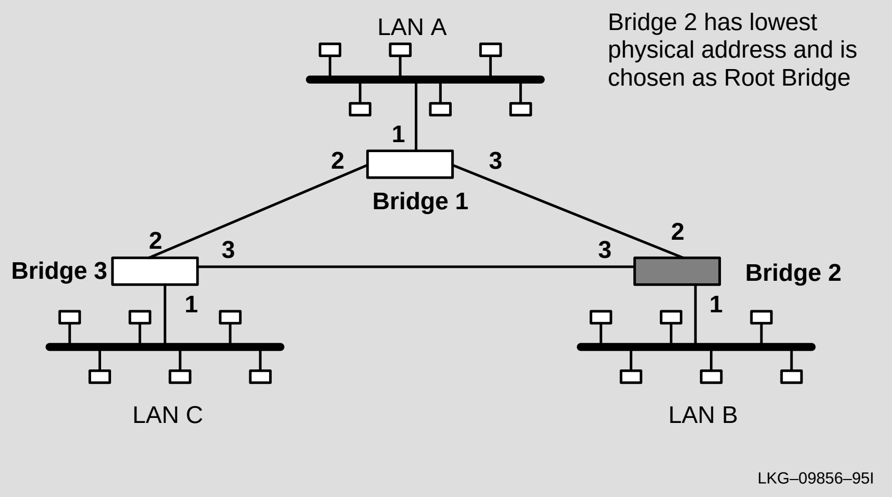
When the spanning tree uses its default settings, the spanning tree algorithm generally provides acceptable results. However, the algorithm may sometimes produce a spanning tree with poor network performance. In this case, you can adjust the bridge priority, port priority, and port cost to shape the spanning tree to meet your network performance expectations. The discussion that follows explains how this is done.

Figure 2–1 shows three LANs networked using three bridges. Each bridge is using default bridge priority settings for its spanning tree configuration. In this case, the bridge with the lowest physical address is chosen as the root bridge since the bridge priority of each bridge is the same. In the figure, this is Bridge 2.

The newly configured spanning tree stays intact due to the repeated transmissions of HELLO BPDUs from the root bridge at a preset interval (bridge hello time). Through this process, designated bridges are updated with all configuration information. The designated bridges then regenerate the information from the HELLO BPDUs and distribute it to the LANs for which they are designated bridges.

Figure 2–1 Networked LANs Before Spanning Tree

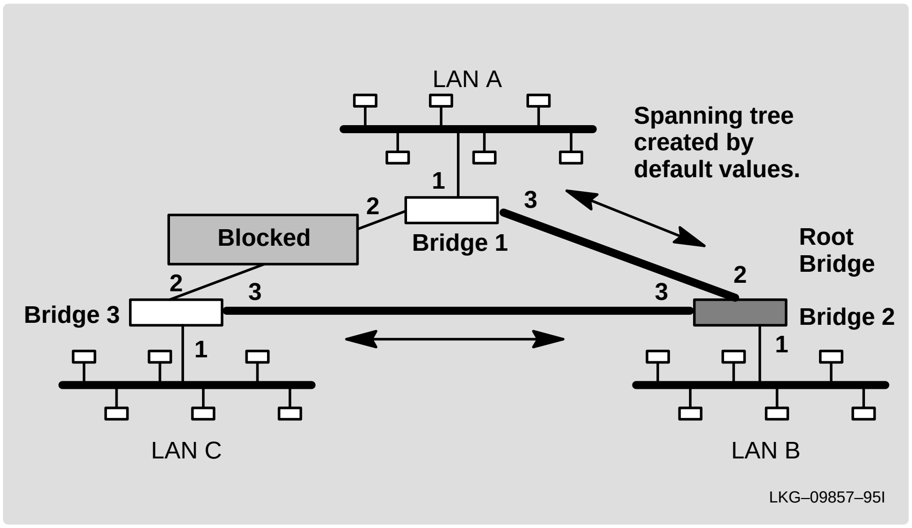
Bridge 1	Bridge 2	Bridge 3
Bridge priority: 32768	Bridge priority: 32768	Bridge priority: 32768
Address:	Address:	Address:
00:00:90:00:00:10	00:00:90:00:00:01	00:00:90:00:00:05
Port 1	Port 1	Port 1
Priority: 128	Priority: 128	Priority: 128
Path cost: 100	Path cost: 100	Path cost: 100
Port 2:	Port 2:	Port 2:
Priority: 128	Priority: 128	Priority: 128
Path cost: 17857	Path cost: 17857	Path cost: 17857
Port 3	Port 3	Port 3
Priority: 128	Priority: 128	Priority: 128
Path cost: 17857	Path cost: 17857	Path cost: 17857


LAN A
Bridge 2 has lowest physical address and is chosen as Root Bridge
1
2
3
Bridge 1
2
3
Bridge 3
1
3
2
Bridge 2
1
LAN C
LAN B
LKG–09856–95I

The spanning tree algorithm designates the port connecting Bridge 1 to Bridge 3 (port 2) as a backup port and blocks it from forwarding frames that would cause a loop condition. The spanning tree created by the algorithm using the default values is shown in the Figure 2–2 as the heavy lines connecting Bridge 1 to Bridge 2, and then Bridge 2 to Bridge 3. The root bridge is Bridge 2.

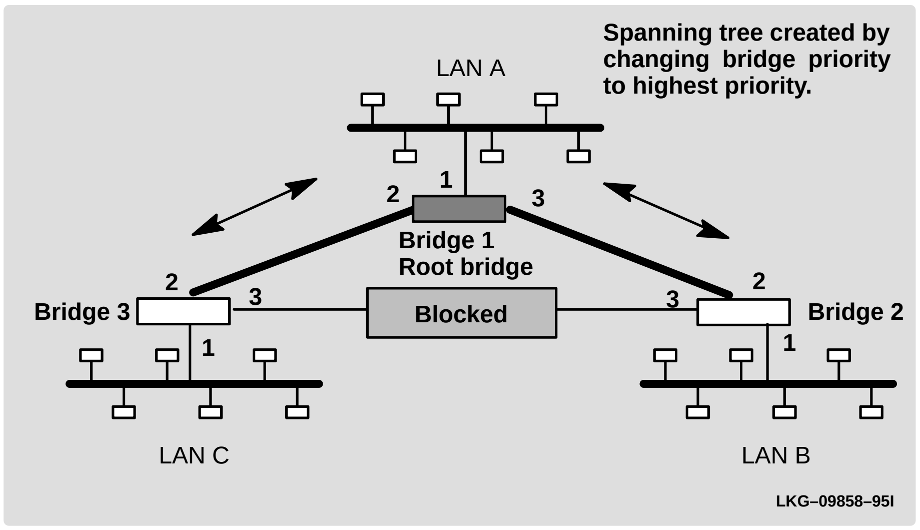
This spanning tree results in poor network performance because the workstations on LAN C can only get to the file server on LAN A indirectly through Bridge 2 rather than using the direct connection between Bridge 1 and Bridge 3.

Figure 2–2 Spanning Tree Created With Default Values



Normally this network seldom uses the port between Bridge 2 and Bridge 3. Therefore, you can improve network performance by making Bridge 1 the root bridge of the spanning tree. You can do this by configuring Bridge 1 with the highest priority of 1000. The spanning tree that results from this modification is shown in Figure 2–3 as the heavy lines connecting Bridge 1 to Bridge 3 and Bridge 1 to Bridge 2. The root bridge is now Bridge 1. The connection between Bridge 2 and Bridge 3 is now blocked and serves as a backup data path.

Figure 2–3 User-Adjusted Spanning Tree



Spanning Tree Bridges and Ethernet Packet Format Translation

An Ethernet /IEEE 802.3 network can simultaneously support the Ethernet data link layer and the IEEE 802.2 data link layer, based on the value of the length/type field in the MAC header. While IEEE 802.3 packets can be sent on FDDI (and vice versa) by only replacing of the MAC header, Ethernet packets require translation of the *data link* header to communicate with and across FDDI. Thus, the bridge must translate to and from Ethernet format to provide transparency across mixed LAN types.

The basic approach consists of translating Ethernet packets to IEEE 802.2 Unnumbered Information (UI) packets using the IEEE 802 SNAP SAP. The SNAP Protocol Identifier has the Organizationally Unique Identifier (OUI) of 00-00-00, with the last two bytes being the Ethernet type value. The reverse mapping happens when going from FDDI to Ethernet.

One special case is AppleTalk 1 AppleARP which has an Ethernet type of 80F3 (hexadecimal) and Protocol Identifier 00-00-F8-80-F3 on FDDI. (AppleTalk 2 AppleARP uses Protocol Identifier 00-00-00-80-F3 on IEEE 802.3 which leads to ambiguities.)

The translation occurs when a frame is sent on a LAN. The original frame format is preserved across serial lines.

Transparent Bridge Terminology and Concepts

This section reviews the terms and concepts commonly used in transparent bridging.

Aging Time

The aging time parameter determines the length of time (age) before a dynamic entry is removed from the filtering database when the port with the entry is in the forwarding state. If dynamic entries are not refreshed before the aging time, they are deleted.

Bridge

A bridge is a protocol-independent device that connects local area networks. These devices operate at the data link layer, storing and forwarding data packets between LANs.

Bridge Address

The bridge address is the least significant 6-octet part of the bridge identifier used by the spanning tree algorithm to identify a bridge on the network. The bridge address is set to the Media Access Control address of the lowest numbered port by default. You can override the default address by using the **set bridge** configuration command.

Bridge Hello Time

The bridge hello time specifies how often a bridge sends out HELLO BPDUs (containing bridge configuration information) when it becomes the root bridge in the spanning tree. This value is useful only for the root bridge since it controls the hello time for all bridges in the spanning tree. Use the **set protocol bridge** command to set the bridge hello time.

Bridge Forwarding Delay

The bridge forward delay specifies how much time a bridge port spends in the listening state as well as the learning state. The forward delay is the amount of time the bridge port listens in order to adjust the spanning tree topology. It is also the amount of time the bridge spends learning the source address of every packet that it receives while the spanning tree is configuring. This value is useful only for the root bridge since it controls the forward delay for all bridges in the spanning tree.

The root bridge conveys this value to all bridges. This time is set with the **set protocol bridge** command. The procedure for setting this parameter is discussed in the next chapter.

Bridge Identifier

The spanning tree algorithm uses the bridge identifier as a unique ID to determine the spanning tree. Each bridge in the network must have a unique bridge identifier.

The bridge identifier consists of two parts: a least significant 6-octet bridge address and a most significant 2-octet bridge priority. By default the bridge address is set to the Media Access Control address of the lowest numbered port. You can override the default address with the **set bridge** configuration command.

Bridge Maximum Age

The bridge maximum age specifies the amount of time that Spanning Tree protocol information is considered valid before the protocol discards the information and a topology change occurs. All the bridges in the spanning tree use this age to time out the received configuration information in their database. This allows a uniform time-out for every bridge in the spanning tree. Use the **set protocol bridge** command to set the bridge maximum age.

Bridge Priority

The bridge priority is the most significant 2-octet part of the bridge identifier set by the **set protocol bridge** command. This value indicates the priority for each bridge to become the root bridge of the network. In setting the bridge priority, the spanning tree algorithm chooses the bridge with the highest priority to be the root bridge of the spanning tree. The bridge with the lowest numerical value has the highest priority.

Designated Bridge

The designated bridge is the bridge that claims to be the closest to the root bridge (or the lowest bridge ID value if two or more bridges have equal cost root paths) on a specific LAN. This closeness is measured according to the accumulated path cost to the root bridge.

Designated Port

The designated port is the port ID of the designated bridge attached to the LAN.

Filtering and Permanent Databases

The bridge's filtering and permanent databases contain information about station addresses that belong to specific ports connected to the LAN.

The filtering database is initialized with entries from the permanent database. These entries are permanent and survive power cycles or system resets. You can add or delete these entries through the bridge configuration commands. Entries in the permanent database are stored as records in non-volatile memory, and the number of entries is limited by the size of non-volatile memory.

Note: You can also add entries (static) by using console commands but these do not survive power cycles and system resets.

The filtering database also accumulates entries learned by the bridge (dynamic entries) that have an aging time associated with them. When entries are not refreshed over a certain time period (age time), they are deleted. Static entries are ageless; dynamic entries cannot overwrite them.

Entries in the filtering and permanent databases contain the following information:

- **Address** – 6-byte MAC address of the entry.
- **Port Map** – Specifies all port numbers associated with that entry.
- **Type of Entry** – Specifies one of the following types:
  - Reserved Entries. Reserved by the IEEE 802.1d committee.

  - Registered Entries. Consist of unicast addresses belonging to communications hardware attached to the box or multicast addresses enabled by protocol forwarders.

  - Permanent Entries. Entered by the user in the configuration process. They survive power on/off and system resets.

  - Static Entries. Entered by the user in the console process. They do not survive power on/off and system resets and are ageless.

  - Dynamic Entries. Dynamically learned by the bridge. They do not survive power on/off and system resets and have an associated age.

  - Free. Locations in the database that are free to be filled by address entries.

- **Address Age *(*dynamic entries only*)*** – Resolution of time period when address entries are ticked down before being discarded. The user can set this value.

Make changes to the permanent database through the bridge configuration commands and make changes to the filtering database through the GWCON console process.

Parallel Bridges

Two or more bridges connecting the same LANs are considered parallel bridges.

Path Cost

Each port interface has an associated path cost which is the relative cost of using this port to reach the root bridge in a bridged network. The spanning tree algorithm uses the path cost to compute a path that minimizes the cost from the root bridge to all other bridges in the network topology. The sum total of all the port costs on the path to the root bridge is called the root path cost.

Port

A port represents the bridge’s connection to each attached LAN or WAN. A bridge must have at least two ports to function as a bridge.

Port ID

The port ID is a 2-octet port identifier. The most significant octet represents the port priority and the least significant octet represents the port number. Both port number and port priority are user-settable. Each port's ID must be unique within the bridge.

Port Number

The port number is a user-assigned 1-octet part of the port ID whose value represents the attachment to the physical medium. A port number of zero is not allowed.

Port Priority

The port priority is the second 1-octet part of the port ID. This value represents the priority of the port that the spanning tree algorithm uses in making comparisons for port selection and blocking decisions.

Resolution

Resolution is the time factor by which dynamic entries are ticked down as they age within the database. The range is 1 to 60 seconds.

Root Bridge

The root bridge is the bridge selected as the root of the spanning tree because it possesses the highest priority bridge ID. This bridge is responsible for keeping the spanning tree intact by regularly emitting HELLO BPDUs (containing bridge configuration information). The root bridge is the designated bridge for all the LANs to which it is connected.

Root Port

The root port is the port ID of a bridge's port that offers the lowest cost path to the root bridge.

Spanning Tree

The spanning tree is a topology of bridges and the ports interconnecting them such that there is one and only one path between any two end stations.

Transparent Bridging

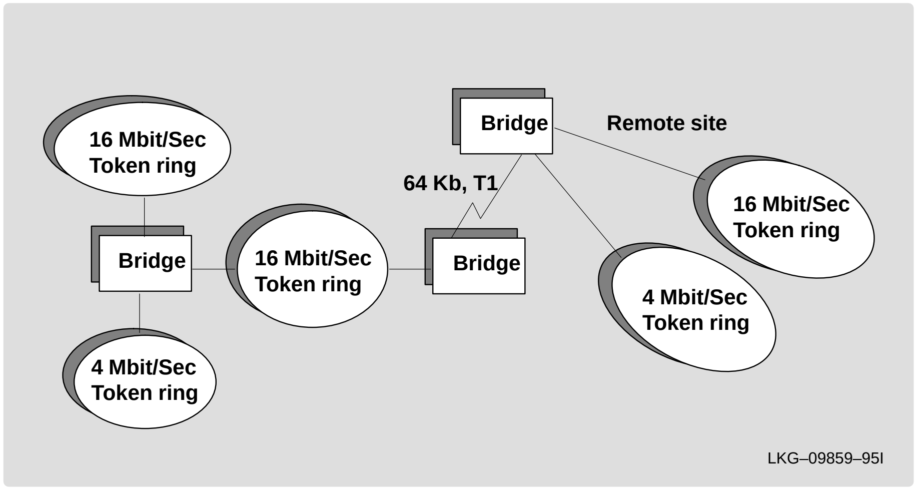
This type of bridging involves a mechanism which is transparent to end stations. Transparent bridging interconnects local area network segments by bridges designated to forward data frames through a spanning tree algorithm.

Source Routing Bridging (SRB)

Source routing is a method of forwarding frames through a bridged network in which the source station identifies the route that the frame follows. In a distributed routing scheme, routing tables at each bridge determine the path that data takes through the network. By contrast, in a source routing scheme, the source station defines the entire route in the transmitted frame.

The Source Routing Bridge provides local bridging over 4- and 16-Mbps token rings, as shown in Figure 2–4. It can also connect remote LANs through a WAN link.

Figure 2–4 Example of Source Routing Bridge Connectivity



Among its features, the source routing bridge provides:

- **IBM compatibility** – The bridge is compatible with the IBM source routing bridge. You can use the bridge to connect IBM PC LANs running systems such as OS/2, PC LAN Manager, and NETBIOS. The bridge can also carry IBM SNA traffic between PC LANs and mainframes.

- **Bridge Tunneling** – By encapsulating source routing packets in IP, the bridging router dynamically routes these packets through internetworks to the desired destination end station without network size restrictions.

  Source routing end stations see this path (the tunnel) as a single hop, regardless of the network complexity. This helps overcome the usual seven-hop distance limit encountered in source routing configurations. This feature also lets you connect source routing end stations across non-source routing media (for example, Ethernet networks).

Source Routing Bridge Operation

The source station defines the entire route in the transmitted frame in a source routing configuration. Both end stations and bridges participate in the route discovery and forwarding process. The following steps describe this process:

1. A source station sends out a frame and finds that the frame's destination is not on its own (local) segment or ring.

2. The source station builds a route discovery broadcast frame and transmits it onto the local segment.

3. All bridges on the local segment capture the route discovery frame and send it over their connected networks.

   As the route discovery frame continues its search for the destination end station, each bridge that forwards it adds its own bridge number and segment number to the routing information field (RIF) in the frame. As the frame continues to pass through the bridged network, the RIF compiles a list of bridge and segment number pairs describing the path to the destination.

   When the broadcast frame finally reaches its destination, it contains the exact sequence of addresses from source to destination.

4. When the destination end station receives the frame, it generates a response frame including the route path for communication. Frames that wander to other parts of the bridged network (accumulating irrelevant routing information in the meantime) never reach the destination end station and are ignored by other end stations.

5. The originating station receives the learned route path. It can then transmit information across this established path.

Source Routing Frames

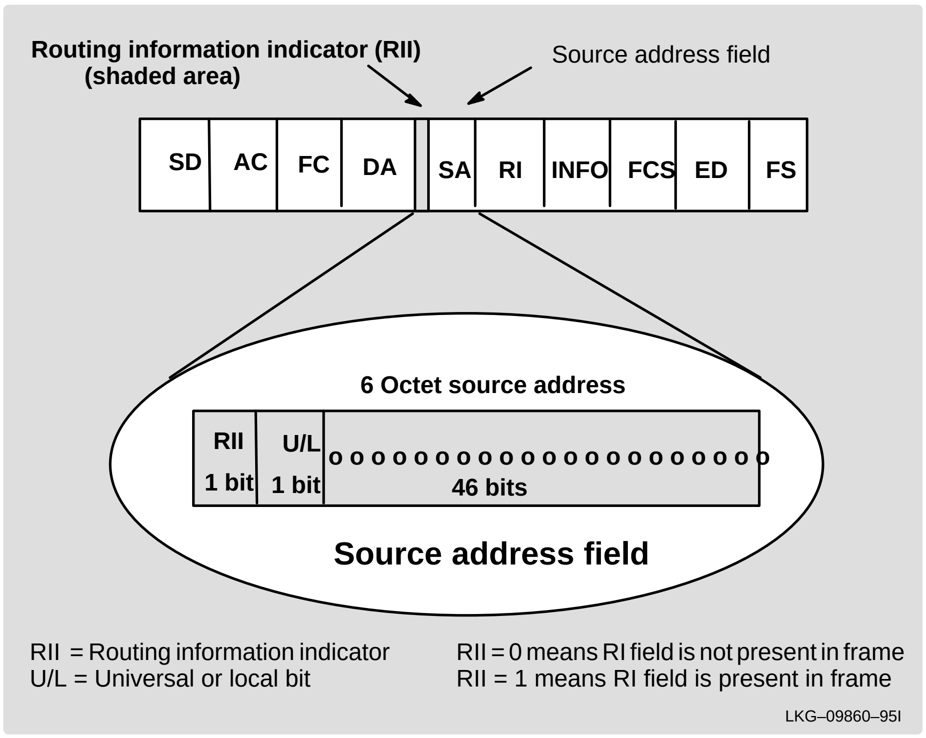
Bridges interconnect LANs by relaying MAC frames between the MAC layers of the bridged LANs. MAC frames provide the necessary “Where?” information in the form of source and destination addresses.

In source routing, the data frame forwarding decision is based on routing information within the frame. The source station that originates the frame designates the route that the frame travels by embedding a description of the route in the routing-information field (RIF) of the transmitted frame. A closer look at the various types of source routing bridge frames helps to further explain how bridges obtain and transmit routing information.

Because source routing MAC frames contain routing information necessary for data communication over multi-ring environments, their formats differ slightly from the typical token ring MAC frames. The presence of a “1” in the Routing Information Indicator (RII) within the source address field indicates that an RIF containing routing information follows the source address.

Figure 2–5 illustrates the format of the source address field of a source routing frame.

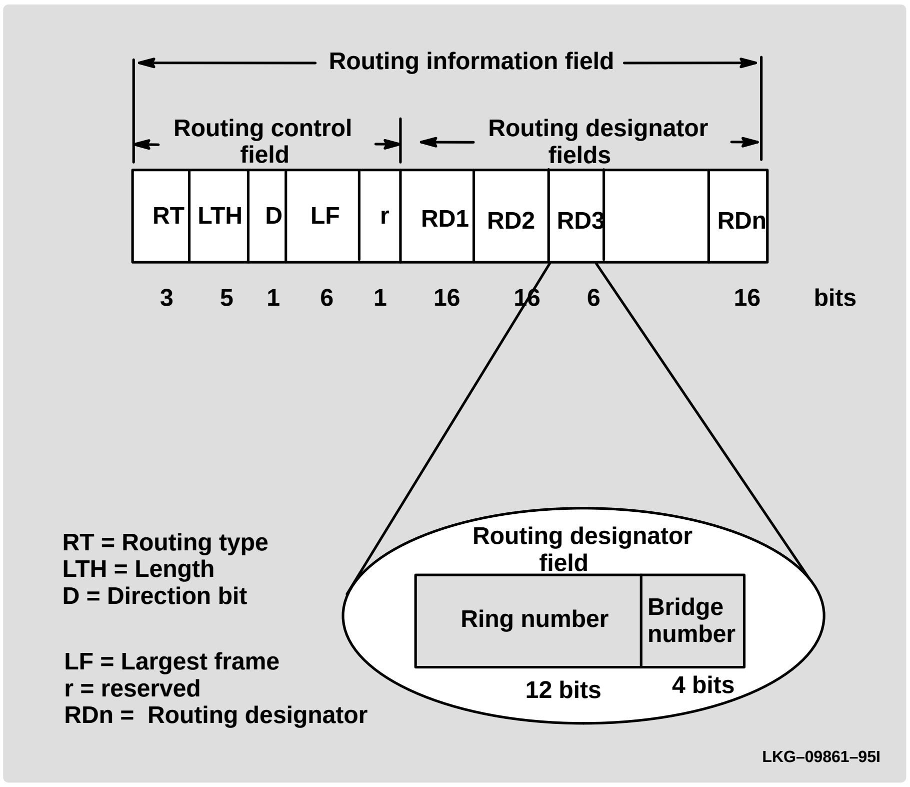
Figure 2–5 802.5 Source Address Format


Routing information indicator (RII)
(shaded area)
Source address field
SD
AC
FC
DA
SA
RI
INFO
FCS
ED
FS
6 Octet source address
RII
1 bit
U/L
1 bit
o o o o o o o o o o o o o o o o o o o o o o
46 bits
Source address field
RII = Routing information indicator
U/L = Universal or local bit
RII = 0 means RI field is not present in frame
RII = 1 means RI field is present in frame
LKG–09860–95I

When the RII in the source address field is set to 1, an RIF is present after the source address. The RIF is required because it provides route information during source routing. It consists of a 2-octet routing control (RC) field and a series of 2-octet route designator (RD) fields.

Figure 2–6 illustrates the format of the routing information field.

Figure 2–6 802.5 Routing Information Field



The following information describes each specific field found in the RIF:

- **Routing Type (RT)** – Indicates by bit settings if the frame is to be forwarded through the network along a specific route or along a route (or routes) that reaches all interconnected LANs. Depending on the bit settings in this field the source routing frame can be identified as one of the following types:
  - All-paths explorer frame (explorer frame)
  - Spanning-tree explorer frame (explorer frame)
  - Specifically-routed frame (routing frame)
  - Spanning-tree routed frame (routing frame)

All-paths explorer frames have RT bits set to 100. These frames are generated and routed along every non-repeating route in the network (from source to destination). This process results in as many frames arriving at the destination end station as there are different routes from the source end station. Each bridge on the path adds routing designators to the frame.

A **S**panning-Tree Explorer frame has RT bits set to 110. Only spanning tree bridges relay the frame from one network to another. This means that the frame appears only once on every ring in the network and only once at the destination end station. A station initiating the route discovery process can use this frame type. The bridge adds routing designator fields to the frame. It can also be used for frames sent to stations using a group address which is discussed more fully in the next section.

Specifically-routed frames have the first RT bit set to 0. In this case, the Route Designator (RD) fields contains specific routing information that guides the frame through the network to the destination address.

A spanning tree routed frame has the RT bits set to 111. Only spanning tree bridges relay the frame one network to another. This means that the frame appears only once on every ring in the network and therefore only once at the destination end station. The bridges do not add any routing designators to the frame. If the frame has been forwarded from network A and is being transmitted on network B, when the frame is stripped it is not forwarded back to network A. This type of frame is the default type that a station uses to send frames to stations using a group address.

- **Length bits (LTH)** – Indicates the length (in octets) of the RI field.

- **Direction bit (D)** – Indicates the direction the frame takes to traverse the connected networks. If this bit is set to 0, the frame travels the connected networks in the order in which they are specified in the routing information field (for example, RD1 to RD2 to.... to RDn). If the direction bit is set to 1, the frame travels the networks in the reverse order.

- **Largest Frame Bits (LF)** – Indicates the largest frame size of the INFO field that may be transmitted between two communicating end stations on a specific route. The LF bits are meaningful only for STE and ARE frames. In Specifically Routed Frames (SRF), the bridge ignores the LF bits and cannot alter them. A station originating an explorer frame sets the LF bits to the maximum frame size it can handle. Forwarding bridges set the LF bits to the largest value that does not exceed the minimum of

  - The indicated value of the received LF bits
  - The largest MSDU size supported by the bridge
  - The largest MSDU size supported by the port from which the frame was received
  - The largest MSDU size supported by the port on which the frame is to be transmitted

  The destination station may further reduce the LF value to indicate its maximum frame capacity.

  LF bit encodings are made up of a 3-bit base encoding and a 3-bit extended encoding (6 bits total). The SRT bridge (explained in a later section) contains an LF mode indicator so the bridge can select either base or extended LF bits. When the LF mode indicator is set to the base mode, the bridge sets the LF bits in explorer frames with the largest frame base values. When the LF mode indicator is set to extended mode, the bridge sets the LF bits in explorer frames in with the largest frame extended values.

- **Route Designator fields (RDn)** – Indicates the specific route through the network according to the sequence of the RD fields. Each RD field contains a unique network 12-bit ring number and 4-bit bridge number that differentiates between two or more bridges when they connect the same two rings (parallel bridges). The last bridge number in the routing information field has a null value (all zeros).

The Spanning Tree Explore Option

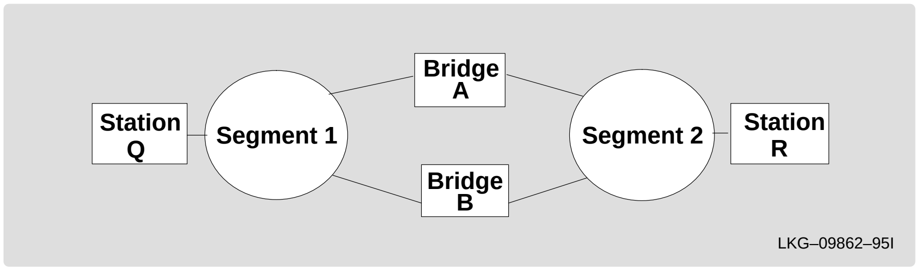
The spanning tree explore option lets you select a single route to a destination when your network has two or more bridges connecting the same LANs. With this feature enabled, only the bridges you select receive Spanning-Tree Explorer (STE) frames. This option allows you to simulate a spanning tree network and balance traffic loads.

Simulating a Spanning Tree Network

A spanning tree network contains a single data route between any two end stations. If your network uses two or more parallel bridges, such as those in Figure 2–7, you can manually configure a spanning tree in a network by preventing duplication of discovery frames onto the network. Without spanning tree explore enabled, if Station Q transmits a discovery frame to a Station R, both Bridge A and Bridge B retransmit that frame. Segment 2 then receives two copies of the same frame.

With spanning tree explore enabled, each LAN segment on the network receives only one copy of the transmitted frame. Only the bridges you select can receive STE frames, reducing the creation of redundant frames, lowering network overhead.

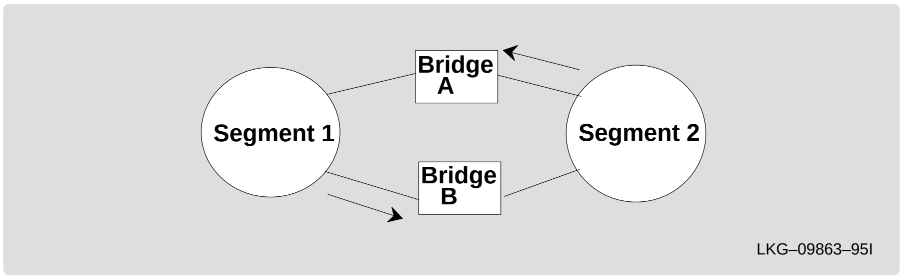
Figure 2–7 Example of Parallel Bridges



Balancing Traffic Loads

You can also use the spanning tree explore option for load balancing. For example, in Figure 2–8, Bridge A is configured to accept STE frames over the interface connecting Segment 2. Bridge B is configured to accept STE frames over the interface connecting Segment 1. Traffic travels in the direction of the arrows. This configuration allows parallel bridges to share the traffic load.

Figure 2–8 Using Spanning Tree Explore for Load Balancing



Note: For source routing to work, some end-node applications such as the IBM PC LAN program require you to enable spanning tree explore on attached interfaces. For parallel bridge configuration, the spanning tree explore option should be enabled only on one of the parallel interfaces. No serious harm (other than some extra traffic) results from having too many interfaces enabled for the spanning tree.

If you use the spanning tree explore option and any bridge on the single-route path goes down, source routing traffic cannot reach its destination. You must manually reconfigure an alternate path.

Protocol Filtering

A brouter can perform both bridging and routing. Protocol filtering is the process that determines whether the incoming data is routed or bridged. This decision is based on the contents of the destination address field and protocol type of incoming frames.

Table 2–1 shows how the "Bridge or Route?" question is answered, based on the destination address and protocol type contents.

Table 2–1 Route/Bridge Decision Table

If the Destination MAC Address in the Received Frame Contains:	The Bridge uses this Logic:
The Bridge’s MAC Address	The bridge passes the frame to the configured layer 3 protocol that either routes the frame or passes it to the application residing in the brouter.
Multicast or Broadcast Address	If the protocol type is configured for routing, then the frame is passed to the layer 3 protocol that either routes the frame or passes it to the application residing in the brouter. If the protocol type is not configured for routing, then the frame is bridged.
Unicast (not the Bridge’s MAC Address)	The frame is bridged if its protocol type is not configured for routing, otherwise it is dropped.

Source Routing Bridge Terminology and Concepts

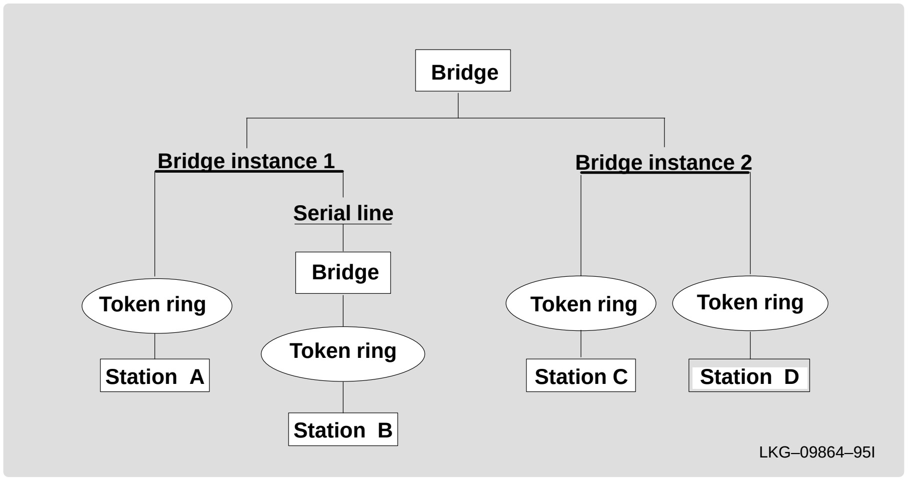
This section reviews the terms and concepts commonly used in source routing bridging.

Bridge Instance

The bridge instance identifies a logical bridge defined within the system. For example, in a bridge with two configured bridge instances, the bridge instance identifiers are 1 and 2.

Bridge instances within a single system are independent and do not communicate. For example, in Figure 2–9, Station A cannot pass data to either station on Bridge Instance 2. It can only pass frames to Station B. In effect, the bridge instance allows you to create two separate networks. These networks do not communicate unless they physically interconnect at some other point.

Figure 2–9 Bridge Instances Within a Bridge



Bridge Number

The bridge number identifies the source routing bridge specific to a segment or ring. This is a 4-bit integer value which must be unique on a given segment or ring.

Explorer Frames

The source routing bridge adds routing information to an explorer frame as it forwards the frame through the network to its destination end station. The explorer frame is used to discover routes. There are two types of explorer frames: All-Routes Explorer (ARE) frames and Spanning-Tree Explorer (STE) frames. ARE frames are forwarded by all ports while STE frames are forwarded only by ports assigned to forward them by the Spanning Tree protocol.

Interface Number

The interface number identifies a device (network interface card) within the bridge and links it to other configuration information. When you configure the router software, the router/bridge sequentially numbers the interfaces. To use the source routing bridge, you must use the numbers assigned in that process to identify the interface connecting each network segment.

Route

A path through a series of LAN segments and bridges (SRB bridges, for example).

Route Discovery

The process by which a route is learned to a destination end station.

Segment Number

The segment number identifies each individual LAN, such as a single token ring or serial line.

Source Routing

Source routing is a bridging mechanism that routes frames through a multi-LAN network by specifying in the frame the sequence of bridges and ports that the frame travels to reach its destination.

Source Routing Transparent Bridge (SRT)

Having worked hard to adopt standardized technologies (Ethernet and token ring are both defined by IEEE), you may actually be forced back into the proprietary arena when trying to connect them. This is because bridges function differently in Token-ring and Ethernet networks.

Aside from the differences such as bit-ordering, packet size, and acknowledgement bits, differences in bridging methods are another obstacle. Ethernet bridges use the transparent bridging method in which the bridges determine the route of the traffic through the network. Token-ring networks use transparent bridging only in some instances so they generally depend on source routing as the primary bridging method.

Source routing cannot operate in a transparent environment because transparent packets contain no routing information. In this case, the bridge has no way of knowing whether to forward the packet. While transparent bridging can operate in a source routing environment, it does so without any routing information being passed to an end station. Significant information (for example, packet sizing) is missing and can potentially create problems.

Source Routing Transparent (SRT) bridging is defined in IEEE 802.1D (ISO 10038) Appendix C. SRT is a bridging technology that attempts to resolve a large part of the incompatibility issues inherent in bridging token ring and Ethernet. It saves you the cost of installing multiple bridges and separate links to support the two types of traffic by adding a parallel bridging architecture (rather than an alternative) to the transparent bridging standard.

The following section describes SRT Bridging in more detail and includes the following:

- General Description
- Source Routing Transparent Bridge Operation and Architecture
- Source Routing Transparent Bridge Terminology and Concepts

General Description

A Source Routing Transparent (SRT) bridge is a MAC bridge that performs source routing when source routing frames with routing information are received and that performs transparent bridging when frames are received without routing information. In SRT, all the bridges between Ethernets and token rings are transparent. The bridges operate at the MAC sub-layer of the data link layer and are completely invisible to the end stations.

The SRT bridge distinguishes between frames that must be bridged transparently and those that must be bridged by source routing by checking the value in the RII field of the frame (see the section “Source Routing Bridge” for more information).   An RII value of 1 indicates that the frame is carrying routing information while a value of 0 in the RII indicates that no routing information is present. With this method, the SRT bridge forwards transparent bridging frames without any conversions to the outgoing media (including token ring). Source routing frames are restricted to a source routing bridging domain.

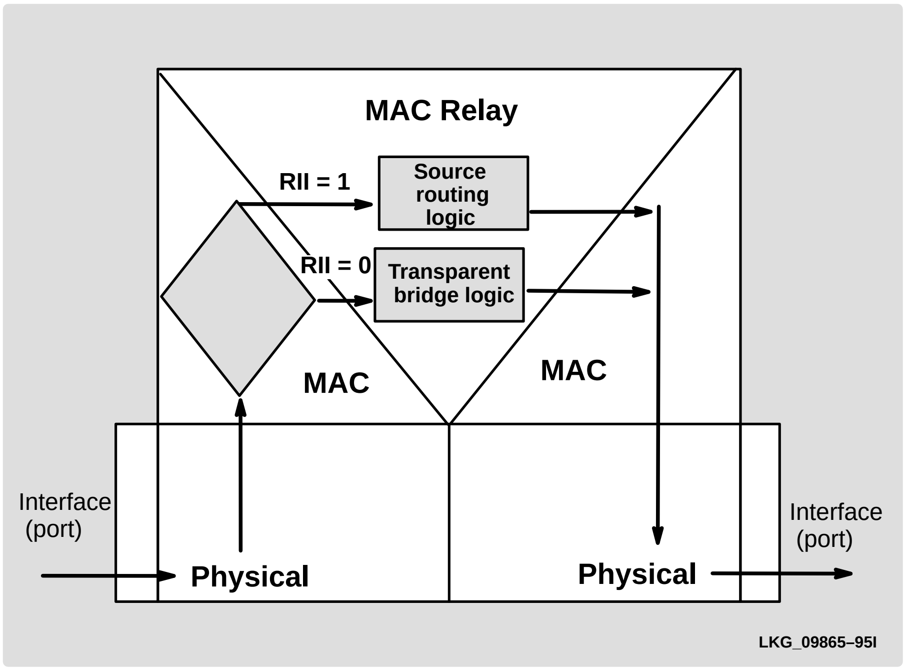
The Spanning Tree protocol and algorithm forms a single tree involving all the networks connected by SRT bridges. The SRT bridged network offers a larger domain of transparent bridging with a sub-domain of source routing. Thus, transparent frames are capable of reaching to the farthest side of the SRT and TB bridged LAN while source routed frames are limited to only the SRT and SRB bridged LAN. In the SRT bridging model, source routing and transparent bridging parts use the same spanning tree. In the SRT bridged domain, end stations decide whether to use Source Routing or Transparent Bridging when sending a packet.

Source Routing Transparent Bridge Operation and Architecture

With an SRT bridge, each bridge port receives and transmits frames to and from the attached local area networks using the MAC services provided by the individual MAC entity associated with that port. The MAC relay entity takes care of the MAC-independent task of relaying frames between bridge ports. If the received frame is not source routed (RII = 0), then the bridge frame is forwarded or discarded using the transparent bridging logic. If the received frame is source routed (RII = 1), then the frame is handled according to the source routing logic.

This process is illustrated in Figure 2–10. The arrows represent the data path.

Figure 2–10 SRT Bridge Operation



SRT differentiates between source-routed and non-source-routed traffic on a frame-by-frame basis.

Source Routing Transparent Bridge Terminology

This section reviews the terms and concepts commonly used in SRT bridging.

Explorer Frames

The source routing bridge adds routing information to an explorer frame as it forwards the frame through the network to its destination end station. The explorer frame discovers routes. There are two types of explorer frames: All-Routes Explorer (ARE) frames and Spanning-Tree Explorer (STE) frames. ARE frames are intended to be forwarded by all ports while STE frames are forwarded only by ports assigned to forward them by the Spanning Tree protocol.

Routing Information Field (RIF)

In source routing, the data frame forwarding decision is based on routing information within the frame. Before forwarding the frame end stations obtain the route to the destination station by the route discovery process. The station that originates the frame (the source station) designates the route that the frame travels by embedding a description of the route in the Routing Information Field (RIF) of the transmitted frame.

Routing Information Indicator (RII)

Since source routing MAC frames contain routing information necessary for data communication over multi-ring environments, their format differs slightly from the typical token ring MAC frames. The presence of a 1 in the source address field called the Routing Information Indicator indicates that a Routing Information Field containing routing information follows the source address. The SRT bridge distinguishes between source-routed and non-source-routed frames by checking for a 1 or 0 value in the RII field.

Source Routing

Source routing is a bridging mechanism that routes frames through a multi-token ring LAN network by specifying in the frame the route it travels.

Spanning Tree

The spanning tree is a topology of bridges in which there is only one data route between any two end stations.

Transparent Bridging

This type of bridging involves a mechanism that is transparent to end stations. Transparent bridging interconnects local area network segments by bridges designated to forward data frames through the operation of the spanning tree algorithm.

Adaptive Source Routing Transparent Bridge (ASRT) (SR-TB Conversion)

While source routing is still available in the SRT model, it is only available between adjacent source routing token rings. Source routing-only bridges cannot coexist with SRT bridges that link Ethernet and token ring LANs. Because a token ring end node needs to communicate with an Ethernet node, it must be configured to omit RIFs. Also, if the end node is configured to omit RIFs, it cannot communicate through ordinary source routing bridges that require that RIF.

The following section describes the ASRT bridge in detail and includes the following sections:

- General Description
- Source Routing – Transparent Bridge Operation
- Source Routing – Transparent Bridge Terminology and Concepts

General Description

The Source Routing – Transparent Bridge (SR-TB) option interconnects networks using source routing bridging (source routing domain) and transparent bridging (transparent bridging domain). It transparently joins both domains. During operation, stations in both domains are not aware of the existence of each other or of the SR-TB bridge. From a station’s point of view, any station on the combined network appears to be in its own domain.

The bridge achieves this functionality by converting frames from the transparent bridging domain to source routing frames before forwarding them to the source routing domain (and vice versa). This is accomplished by the bridge maintaining a database of end station addresses each with its Routing Information Field in the source routing domain. The bridge also conducts route discovery on behalf of the end stations present in the transparent bridging domain. The route discovery process is used to find the route to the destination station in the source routing domain. Frames sent to an unknown destination are sent in the Spanning-Tree Explorer (STE) format.

The SR-TB bridge anticipates three types of spanning trees:

- A spanning tree formed by transparent bridge domain
- A spanning tree formed by source routing bridge domain
- A special spanning tree of all SR-TB bridges

The next sections discuss the operation of the SR-TB bridge in more detail.

Source Routing – Transparent Bridge Operation

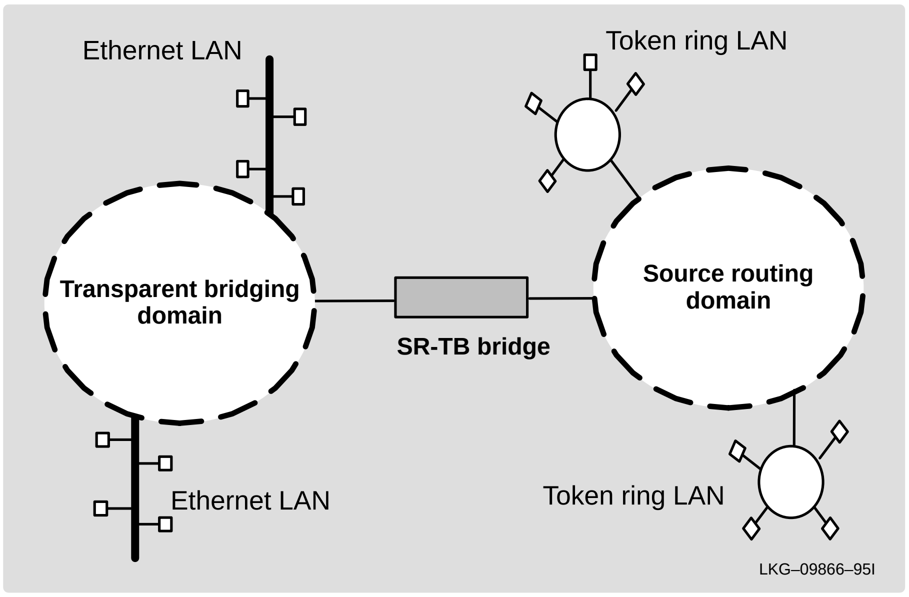
During SR-TB bridging, a network is partitioned into a series of two or more separate domains. Each domain is made up of a collection of LAN segments interconnected by bridges all operating under a common bridging method. This allows networks comprised of two types of domains (depending on the bridging method):

- Source routing domains
- Transparent bridging domains

With separate domains, each source routing domain has a single-route broadcast topology set up for its bridges. Only bridges belonging to that source routing *spanning tree* are designated to forward single-route broadcast frames. In this case, frames that carry the single-route broadcast indicator are routed to every segment of the source routing domain. Only one copy of the frame reaches each segment, since the source routing spanning tree does not allow multiple paths between any two stations in the domain.

Figure 2–11 shows an example of these domains.

Figure 2–11 SR-TB Bridge Connecting Two Domains



Specific Source Routing and Transparent Bridging Operations

The SR-TB bridge is a two-port device with a MAC interface assigned to the LAN segment on the source routing side and another assigned to the LAN segment on the transparent bridging side. Each end station reads the appropriate MAC layer for its LAN segment. This means that bridging functions can be divided into two types of operations:

- Transparent bridging operations
- Source routing bridging operations

On the transparent bridging side, the SR-TB bridge operates the same as any other transparent bridge. The bridge keeps a table of addresses for stations it knows are transparent bridging stations. The SR-TB bridge observes the inter-bridge protocols necessary to create and maintain the network spanning tree since more than one SR-TB bridge joins different domains.

The SR-TB bridge forwards the frames received from its transparent bridging station to the source routing side of the bridge only if the destination address carried in the frame is not found in the bridge’s transparent bridging side address table.

On the source routing bridging side, the SR-TB bridge combines the functions of a source routing bridge and a source routing end station in a specific way. As a source routing end station, the bridge maintains an association of destination addresses and routing information on the source routing side. It communicates either as an end station for applications in the bridge itself (for example, network management) or as an intermediary for stations on the transparent bridging side.

The SR-TB bridge forwards the frames received from its transparent bridging station to the source routing side of the bridge only if the destination address carried in the frame is not found in the bridge’s transparent bridging side address table. Frames transmitted by the bridge’s source routing station carry the routing information associated with the bridge, if such information is known and held by the bridge.

As a source routing bridge, the SR-TB bridge participates in the route discovery process and in the routing of frames already carrying routing information. The route designator unique to the SR-TB bridge consists of the LAN number of the individual LAN on its source routing side and the bridge’s individual bridge number.

The bridge also maintains a single LAN number representing all of the LANs on the transparent bridging side. The SR-TB bridge treats each case of received and forwarded frames differently as described in Table 2–2.

Table 2–2 SR-TB Bridge Decision Table

Type of Frame Received	Action Taken by SR-TB Bridge
Non-routed frame received by the Source Routing station.	Does not copy or forward frame carrying routing information.
All-routes broadcast frame received by the source routing station.	Copies frame and sets A and C bits of the broadcast indicator in the repeated frame. If destination address is in the transparent bridging table, bridge forwards the frame without routing information on the transparent bridging network. Otherwise, frame is not forwarded.
Single-route broadcast frame received by the Source Routing station. Bridge *is not* designated as single-route broadcast bridge.	Does not copy or forward the frame.
Single-route broadcast frame received by the Source Routing station. Bridge *is* designated as single-route broadcast bridge.	Copies frame, sets A and C bits in the broadcast indicator, removes the routing information from the frame, and forwards modified frame to transparent bridging side. Adds its bridge number to saved routing information field and the LAN number for transparent bridging side. Changes broadcast indicator to non-broadcast, complements D-bit, and stores this routing information for the source address of the frame.

(continued on next page)

Table 2–2 (Cont.) SR–TB Bridge Decision Table

Type of Frame Received	Action Taken by SR-TB Bridge
Non-broadcast frame received by the source routing station.	If frame carries specific route, bridge examines the routing information. If SR-TB bridge is part of the route and appears between the LAN number for the source routing side and LAN number for transparent bridge side, bridge copies frame and sets A and C bits in the repeated frame. Forwards frame to the transparent bridging side without routing information. If bridge does not already have a permanent route for the source address, it saves a copy of the routing information, complements D-bit, and stores saved routing information for the source address of the frame.
Frame received from the Transparent bridging side.	To forward frame to the source routing side, bridge first determines if it has routing information associated with the destination address carried in the frame. If yes, bridge adds routing information to the frame, sets the RII to 1, and queues the frame for transmission on the source routing side. If no, bridge adds a routing control field to the frame containing an indicator for single-route broadcast and two route designators containing the first two LAN numbers and its own individual bridge number.

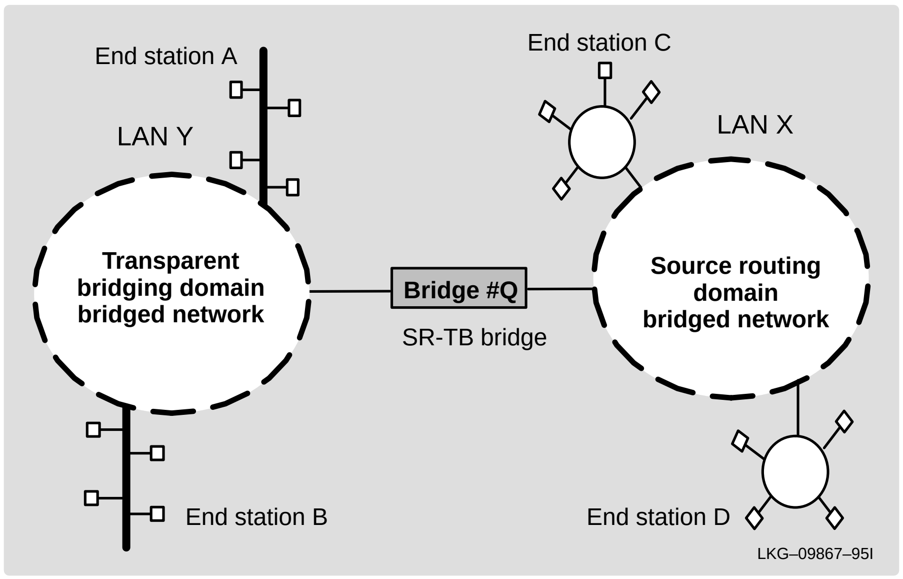
SR-TB Bridging: Four Examples

The SR-TB bridge interconnects source routing domains with transparent bridging domains by transparently joining the domains. During operation, stations in both domains are not aware of the existence of each other or of the SR-TB bridge. From the end station's point of view, any station on the combined network appears to be in its own domain.

The following sections provide specific examples of frame forwarding during SR-TB bridging. These examples assume that the SR-TB bridge is designated as a single-route broadcast bridge. Figure 2–12 provides the following information to accompany the situations described in each section:

- Q is the bridge's own bridge number
- X is the LAN number for the LAN on the source routing side
- Y is the LAN number for the LAN on the transparent bridging side
- A, B, C, and D represent end stations

Figure 2–12   SR-TB Bridging Examples



Example 1: Frame Sent from End Station A to End Station B

When the SR-TB bridge receives a frame with a source address of end station A and a destination address of end station B, it enters end station A’s address into its transparent bridging side address table. This table contains the addresses of stations known to be on the transparent bridging side of the bridge, which is the normal process for transparent bridging.

If end station B’s address is in the transparent bridging side’s address table, the SR-TB bridge does not forward the frame. If end station B’s address is not in the transparent bridging side’s address table and not in the source routing side’s address table, its location is not known to the SR-TB bridge. In this case, the frame is forwarded on the source routing side as a single-route broadcast with no request for route explorer return. Any frame sent by end station B (regardless of its destination) causes its address to be added to the transparent bridging address table. This prevents future forwarding of frames addressed to end station B to the source routing side.

Example 2: Frame Sent from End Station A to End Station C

In this example, end station A’s address is treated the same as in the previous example. Since end station C’s address is definitely not in the transparent bridge address table, the SR-TB bridge forwards the frame on the source routing side.

The bridge then looks for end station C’s address in its source routing address table. This table contains all known addresses with related routing information for stations known to be on the source routing side of the bridge. If C’s address is in the source routing table, the bridge forwards the frame using the routing information in the address table.   If C’s address is not in the source routing table (or if it appears but has null routing information), the bridge forwards the frame on the source routing side as a single-route broadcast with no request for route explorer return.

When end station C receives this frame, it enters end station A’s address in its source routing table together with the reverse direction of the route built from the SR-TB bridge and marks it as a temporary entry. When end station C later tries to send a frame to end station A, it uses this specific route, and because the route is marked as temporary, the frame is sent as a non-broadcast route *with* a request for route explorer return.

When the returning frame arrives at the SR-TB bridge, it is forwarded on the transparent bridge side without routing information but causes the route to end station C to be entered in the source routing table as a temporary route. This further causes the network management entity (SMT) to send a route-explorer frame with an all-routes broadcast setting back to end station C. This lets end station C select the optimal routing for frames addressed to end station A to be entered as a permanent route in the SR-TB bridge's source routing table.

Example 3: Frame Sent from End Station C to End Station D

If the frame is sent as a non-broadcast and crosses over the segment to which the SR-TB bridge is attached, the bridge scans the RII filed for the routing sequence (Lan X to Bridge Q to LAN Y). It cannot find the sequence and does not forward the frame.

If the frame is sent as a single-route broadcast, the bridge discards the frame if end station D is already known to be on the source routing side. If end station D is not known to be on the source routing side, the bridge forwards the frame to the transparent bridging side (minus the routing information), and adds "Q to Y" to the routing information. Finally, it saves the routing information for end station C as a temporary route in the source routing table with a non-broadcast indicator and the direction bit complemented.

If the frame is sent as an all-routes broadcast, the SR-TB bridge discards the frame (because end station D's address is not present in the transparent bridging address table) and makes sure that end station C's address is in the source routing table.

Example 4: Frame Sent from End Station C to End Station A

If the frame is sent non-broadcast, the bridge scans the RII field for the routing sequence (X to Q to Y). When it finds it, it forwards the frame to the transparent bridging side. It also stores the routing information for end station C.

If the frame is sent as a single-route broadcast, the bridge forwards the frame (minus the routing information) to the transparent bridging side and adds "Q to Y" to the routing information. It also sets the non-broadcast indicator, complements the direction bit, and enters the routing information for C's address in its source routing table.

If a temporary entry for end station C already exists in the source routing table, the SR-TB bridge updates the routing information. If the frame is sent as an all-routes broadcast, the bridge discards the frame but makes sure that end station C’s address is in the source routing table.

Source Routing – Transparent Bridge Terminology and Concepts

This section reviews the terms and concepts used in SR-TB bridging.

All Routes Broadcast

The process of sending a frame through every non-repeating route in the bridged LAN.

All Stations Broadcast

The process of addressing a frame (placing all ones in the destination address) so that every station on the ring the frame appears on copies the frame.

Bridge

A bridge is a protocol-independent device that connects local area networks (LAN). Bridges operate at the data link layer, storing and forwarding data packets between LANs.

Bridge Number

The unique number identifying a bridge. It distinguishes between multiple bridges connecting the same two rings.

Explorer Frames

The source routing bridge adds routing information to an explorer frame as it forwards the frame through the network to its destination end station. The explorer frame discovers routes. There are two types of explorer frames: All-Routes Explorer (ARE) frames and Spanning-Tree Explorer (STE) frames. ARE frames are forwarded by all ports while STE frames are forwarded only by ports assigned to forward them by the Spanning Tree protocol.

Ring Number

The unique number identifying a ring in a bridged network.

Route

A path through a series of LANs and bridges (for example, SRB bridges).

Route Designator

A ring number and a bridge number in the Routing Information Field used to build a route through the network.

Route Discovery

The process of learning a route to a destination end station.

Segment Number

The segment number identifies each individual LAN, such as a single token ring or serial line. A segment connects to the bridge, but can also operate independently.

Single Route Broadcasting

The process of sending a frame through a network such that exactly one copy of the frame appears on each ring in the network.

Source Routing Bridging

Source routing is a bridging mechanism that routes frames through a multi-LAN network by specifying in the frame the route it travels.

Spanning Tree

The spanning tree is a topology of bridges such that there is only one data route between any two end stations.

Transparent Bridging

This type of bridging involves a mechanism that is *transparent* to end stations. Transparent bridging interconnects local area network segments by bridges designated to forward data frames in a spanning tree algorithm.

ASRT Bridge Overview

The Adaptive Source Routing Transparent (ASRT) Bridge is a software collection of several bridging options. The ASRT bridge software combines transparent bridging functionality and source routing functionality so that they function separately or can be combined as a single ASRT bridge. This extended functionality allows communication between a strict source routing end station and a transparent end station through an ASRT bridge. Depending on the set of configuration commands used, the ASRT bridge provides the following bridging options:

- Transparent Bridge (STB)
- Source Routing Bridge (SRB)
- Source Routing Transparent Bridge (SRT)
- Source Routing – Transparent Bridge (SR-TB)

Modifications have been built into the ASRT bridge that provide users with extended functionality exceeding compliance with the SRT standard. The ASRT bridge allows compatibility to the installed base of source routing bridges, while still enabling them to link Ethernet and token ring LANs. ASRT also enhances basic SRT functionality in some additional, critical ways described below.

Transparent-Source Routing Compatibility – Issues and Solutions

First, the ASRT bridge provides transparent bridge compatibility with ordinary source routing bridges through source routing bridge conversion (SR-TB). SR-TB was originally a proposed part of the 802.5 specification. This implementation is similar to and can interoperate with IBM’s 8209 conversion bridge.

SR-TB converts transparent bridging frames to source routing frames and vice versa. In other words, instead of just checking to see whether an RIF is present in a packet and forwarding it to a like destination, the ASRT bridge can translate the packet into either format. This means it functions as either a transparent bridge or a source routing bridge by inserting or removing an RIF as necessary. With this functionality, packets can move between Ethernet and SRT token ring LANs and still be compatible with an installed base of source routing token ring LANs.

Elimination of Packet Size Problems

SR-TB also eliminates packet sizing problems in token rings being bridged together across an Ethernet domain. In this configuration, end stations use the source routing protocol which allows them to dynamically determine that there is a network with a 1518-byte maximum frame size between them. The end station automatically honors this limit without a manual reconfiguration. In the reverse situation, bridging Ethernets across a token ring domain, packet size is not an issue since the token ring packet size allowance is much larger.

Bit Ordering in STB (802.3) and SRB (802.5) Bridges

As bridges are continually being built to connect LANs with different MAC address types, bit ordering during data transmission effects the interoperability of these technologies.

In administering MAC addresses, IEEE assigns addresses known as 48-bit IEEE globally assigned unique MAC addresses. These addresses are supported by 802.3, 802.4, 802.5, and FDDI LANs. Two different standards have arisen:

- 802.3 (Ethernet) and 802.4 LANs transmit source and destination addresses with the group bit first and LLC data fields transmitted Least Significant Bit (LSB) first.
- 802.5 (token ring) and FDDI LANs transmit source and destination addresses with the group bit first and LLC data fields transmitted Most Significant Bit (MSB) first.

Note: 802.3 and 802.4 bridges and LANs are now referred to as LSB bridges and LANs. 802.5 and FDDI bridges and LANs are referred to as MSB bridges and LANs.

The difference in the bit transmission standard means that a bridge from LSB to MSB LANs must reverse the bit order of the destination and source MAC addresses at the start of the MAC frame. This is because the different LAN types use the same bit order for the MAC address (group bit first) and yet use a different bit order for the user data either LSB or MSB first.

The misinterpretation of addresses due to reversed bit ordering is compounded by the fact that some of the higher level communications protocols misinterpret MAC addresses altogether. Protocols such as IP and Novell IPX interpret bridging addresses incorrectly because at the time of their initial development, there was no standard representation of MAC addresses.

The bit order differential is best resolved by combining bridging technology (data link layer technology) with routing technology (network layer technology). Rather than ask the user to “reverse engineer” today’s communications protocols and configure each bridge to “flip” or reverse addresses on a case-by-case basis, the problem is more easily solved by routing these protocols.

Routing eliminates the bit order and protocol addressing problems by accessing the detailed packet addresses running at the higher layer. Routing alone is not a complete solution, since other protocols such as IBM Frames and NetBIOS cannot be routed, and SNA routing is limited. Therefore, it is important to implement SRT in a device where bridging and routing work hand-in-hand.

ASRT Configuration Considerations

The ASRT bridge uses the Spanning Tree protocol and algorithm described in the IEEE 802.1D bridge standard over all interfaces. It is possible that more than one spanning tree forms in an environment where different types of bridges exist. For example a spanning tree of all bridges practicing IEEE802.1D protocol (such as STB and SRT) existing with another tree of IBM 8209 bridges. The loops forming from this configuration require you to correct the situation.

TCP/IP Host Services support SDLC relay. Tunneling of bridge traffic over IP is not supported by TCP/IP Host Services. When running as a pure bridge, and not as an IP router, functions usually associated with an IP router are not available. For example, there is no BootP forwarder functionality or any ARP subnet routing capabilities.

3

Bridging Features

This chapter describes bridging features that are available with the Adaptive Source Routing Transparent (ASRT) Bridge.

Bridging Tunnel

The bridge tunnel (encapsulation) is another feature of the ASRT bridge software. By encapsulating packets in industry-standard TCP/IP packets, the bridging router can dynamically route these packets through large IP internetworks to the destination end stations.

End stations see the IP path (the tunnel) as a single hop, regardless of the network complexity. This helps overcome the usual 7-hop bridging distance limit encountered in source routing configurations. It also lets you connect source routing end stations across non-source routing media, such as Ethernet networks.

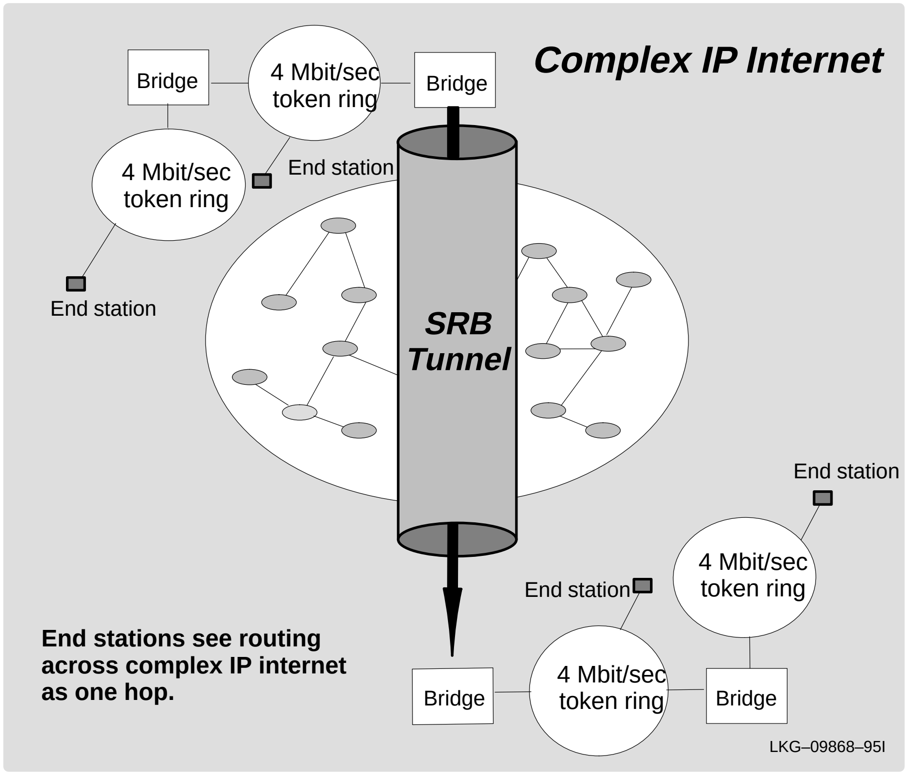
The bridging tunnel overcomes several limitations of regular source routing that includes the following:

- Distance limitations of seven hops
- Large amounts of overhead that source routing causes in wide area networks (WANs)
- Source Routing’s sensitivity to WAN faults and failures (if a path fails, all systems must restart their transmissions)

With the bridge tunnel feature enabled, the software encapsulates packets in IP packets. To the router, the packet looks like a IP packet. Once a frame is encapsulated in an IP envelope, the IP forwarder is responsible for selecting the appropriate network interface based on the destination IP address. This packet can be routed dynamically through large internetworks without degradation or network size restrictions.

Figure 3–1 shows an example of an IP internetwork using the tunnel feature in its configuration.

Figure 3–1 Example of the Bridge Tunnel Feature



The tunnel is transparent to the end stations. The bridging routers participating in tunneling treat the IP internet as one of the bridge segments. When the packet reaches the destination interface, the IP headers are automatically removed and the inner packet proceeds as a standard source routing packet.

Encapsulation and IP Routing Protocols

A major benefit of the encapsulation feature is the addition of a dynamic routing protocol to the routing process. Dynamic routing protocols such as OSPF and Integrated IS-IS offer the following benefits when used with encapsulation:

- **Least-Cost Routing** – Dynamic routing protocols access the shortest path (tunnel), allowing network administrators to distribute traffic over the least expensive route.
- **Dynamic Routing** – Dynamic routing protocols find the least-cost path, detect failures, and reroute traffic with low overhead.

With dynamic routing, tunnels automatically manage paths inside the internetwork. If a line or bridge fails along the path, the tunnel bridge automatically reroutes traffic along a new path. If a path is restored, the tunnel automatically updates to the best path. This rerouting is completely transparent to the end stations.

TCP/IP Host Services (Bridge-Only Management)

The Bridging Router also supports TCP/IP Host services that let you configure and monitor a bridge when routing functions are disabled. This option gives you the following capabilities:

- Management through SNMP
- Telnet server functionality
- Downloading and uploading of configuration through the TFTP protocol
- TFTP neighbor boot functionality
- IP diagnostic tools of ping and traceroute
- Control of the device through SNMP sets and the telnet client.

When viewed from the bridge’s console interface, TCP/IP Host Services is handled as a new protocol having its own configuration and monitoring consoles. These prompts are accessed through the **protocol** command in the `Config>` and `+` (GWCON) consoles.

Bridge-only management functionality is activated by assigning an IP address to the bridge and enabling TCP/IP Host Services. This IP address is associated with the bridge as a whole, instead of being associated with a single interface. When booting over the network, the bridge’s IP address and a default gateway can be learned automatically. Default gateway assignments may also be user-configured.

TCP/IP host services is available whenever bridging is an option in the router software load. These services coexist with the IP routing functionality but do not require IP routing to be present.

Bridge-MIB Support

Bridge Management through SNMP supports the Bridge-MIB as specified by RFC 1286. The entire Bridge-MIB is implemented except for the following:

- The “forwarding database table” for transparent bridges (dot1dTpFdbTable).
- The “static (Destination-Address Filtering) database table” (dot1dStaticTable).
- The “newRoot” and “topologyChange” traps.

The entire Bridge-MIB is read-only.

NETBIOS Filtering

ASRT Bridging performance can be enhanced by an included feature called NETBIOS filtering. NETBIOS filtering lets you configure specific filters through the router configuration process. These filters are sets of rules applied to NETBIOS packets to determine whether the packets are bridged (forwarded) or filtered (dropped). These filters can be applied to the following aspects of the NETBIOS packets:

- Host name fields in the packets
- Arbitrary fields (bytes) in the packets

NETBIOS Filtering using host-names lets you select packets with specific NETBIOS host-names to be bridged or filtered. Another NETBIOS filtering mechanism, byte filtering, specifies certain NETBIOS packets to be bridged or filtered based on arbitrary fields (bytes) in the NETBIOS packets.

Filtering is useful because NETBIOS traffic can contain a high proportion of broadcast packets. Unfiltered NETBIOS traffic can take up a large percentage of network bandwidth (particularly on low-speed WAN interfaces) and greatly reduce the overall performance of WAN interfaces and the network. Configuring NETBIOS filters helps to correct this problem.

A NETBIOS filter (host-name or byte) is made up of three parts:

- The actual filter
- Filter lists
- Filter items

Each filter is made up of one or more filter lists. Each filter list is made up of one or more filter items. Each filter item in the filter list of a filter is evaluated against a packet in the order in which the filter items were specified. When a match between a filter item and a packet is found, the filter list containing the filter item evaluates to the configured value of the matching filter item (Inclusive or Exclusive). This evaluation determines whether the packet matching the filter item is bridged or filtered.

NETBIOS Filtering Using Host Names

NETBIOS Filtering using host-names lets you select packets with specific NETBIOS host-names to be bridged or filtered. When you specify that packets with a particular NETBIOS host-name (or set of NETBIOS host-names) are to be bridged or filtered, the source name or destination name field of the following NETBIOS packet types is examined:

- ADD_GROUP_NAME_QUERY (source NETBIOS name field is examined)
- ADD_NAME_QUERY (source NETBIOS name field is examined)

- DATAGRAM (destination NETBIOS name field is examined)
- NAME_QUERY (destination NETBIOS name field is examined)

Host-name filter lists specify NETBIOS names that are compared with source or destination name fields in the four different types of NETBIOS packets just described.

Note: The result of applying a host-name filter list to a NETBIOS packet which is not one of those four types is Inclusive.

When configuring NETBIOS Filtering using host-names, you specify which ports the filter is applied to and whether it is applied to input or output packets on those ports. Only NETBIOS Unnumbered Information (UI) packets are considered for filtering. Filtering is applied to NETBIOS packets that arrive at the router for either Source Route Bridging (all RIF types) or Transparent Bridging.

When specifying a NETBIOS host-name in a filter, you can indicate the 16th (last) character of the name, as a separate argument, in its hexadecimal form. If this is done, the first 15 bytes of the name are taken as specified and the 16th byte (if any is specified) is determined by the final argument. If fewer than 16 characters are specified (and no 16th byte is specified), then the name is padded with ASCII blank characters up to the 15th character, and the 16th character is treated as a wildcard.

When a specific NETBIOS host-name is evaluated, that name is compared with only certain fields of certain NETBIOS packets. NETBIOS host-names in filter items may include a “?” wildcard character at any point in the NETBIOS host-name, or an “*” as the final character of a NETBIOS host-name. The “?” matches any single character of a host-name. The “*” matches one or more characters at the end of a host-name.

NETBIOS Filtering Using Bytes

Another filtering mechanism, byte filtering, is also available to let you specify which NETBIOS packets are bridged or filtered. With byte filtering, you specify certain NETBIOS packets to be bridged or filtered based on arbitrary fields in the NETBIOS packets. In this case, all NETBIOS packets are examined to determine if they match the configured filtering criteria.

The following are filter items that you can be specify to be evaluated in a byte filter:

- An offset from the beginning of the NETBIOS header
- A byte pattern to match on
- An optional mask to apply to the selected fields of the NETBIOS header

The mask, if present, must be of equal length as the byte pattern, and specifies bytes that are to be logically ANDed with the bytes in the NETBIOS header before the header bytes are compared to the hex pattern for equality. If no mask is specified, it is assumed to be all 1's. The maximum length for the hex pattern (and hence the mask) is 16 bytes (32 hexadecimal digits).

When configuring NETBIOS Filtering using specific bytes, you also specify which ports the filter is applied to and whether it is applied to input or output packets on those ports.

Building a Filter

Each filter is made up of one or more filter lists. Each filter list is made up of one or more filter items. Each filter item within the filter is compared to a packet in the order in which the filter items were specified. When a match between a filter item and a packet is found, the filter list containing the filter item is evaluated for a configured indicator (Inclusive or Exclusive) to determine whether the packet matching the filter item is bridged or filtered.

If no filter items in the filter list produce a match, the filter list is evaluated for its default indicator value (Inclusive or Exclusive). If the filter contains multiple filter lists, each filter list is evaluated. After all filter lists are evaluated, the filter as a whole is given an Inclusive or Exclusive indication. The packet is then bridged or filtered based on that indication.

A filter item is a single rule applied to a particular field of a NETBIOS packet. The result of the application of the rule is either an Inclusive (bridge) or an Exclusive (filter) indication. The following lists the filter items that can be configured with NETBIOS Filtering (the first two items are host-name filters, the last two items are byte filters):

- Include <NETBIOS host-name> <optional 16th character (hex)>
- Exclude <NETBIOS host-name> <optional 16th character (hex)>
- Include <decimal byte offset into NETBIOS hdr>
  <hex pattern starting at that offset><hex mask>
- Exclude <decimal byte offset into NETBIOS hdr>
  <hex pattern starting at that offset><hex mask>

Part of the specification of a filter list indicates whether packets that do not match any of the filter items in the filter list are bridged (Included) or filtered (Excluded). The default action for a filter list is initially set to Include, but this setting can be changed by the user.

Simple and Complex Filters

A simple filter is constructed by combining one filter list with a router port number and an input/output designation. This indicates that the filter list is applied to all NETBIOS packets being input or output on the given port. If the filter list evaluates to Inclusive, then the packet being considered is bridged. Otherwise, the packet is filtered.

A complex filter can be constructed by specifying a port number, an input/output designation, and multiple filter lists separated by one of the logical operators AND or OR. The filter lists in a complex filter are evaluated strictly left to right, and each filter list in the complex filter is evaluated. Each Inclusive filter list result is treated as a TRUE and each Exclusive filter list result is treated as a FALSE. The result of applying all the filter lists and their operators to a packet is a TRUE or FALSE, indicating that the packet is bridged or filtered. Each combination of input/port or output/port can have at most one filter associated with it.

Pseudo Serial Ethernet

Pseudo Serial Ethernet is an optional mode of operation that provides for the encapsulation of any routed protocol on a bridging router proprietary serial line, to be forwarded within an Ethernet encapsulated frame over the same serial line. This allows the protocol to communicate with a pure bridge on the opposite end of the serial line.

When enabled, this mode makes the serial lines appear as an Ethernet interface to the configured routing protocols. The handler uses Ethernet (or IEEE 802.3, as appropriate) encapsulations, thus limiting the protocols to the maximum Ethernet frame size. These Ethernet frames are then sent and received as *bridged* Ethernet frames on the serial line. Any frames arriving on the routed protocol code points from the serial lines are ignored, and bridged Ethernet frames are passed to the bridging or routing forwarders as appropriate.

This encapsulation is normally not necessary with the bridging routers at both ends of the serial line, since both can be configured to route the same set of protocols over the same serial line.

Multiple Spanning Tree Protocol Options

The ASRT Bridge lets you extend Spanning Tree protocol options to cover as many configuration options as possible. The next sections provide information on these features.

Background: Problems with Multiple Spanning Tree protocols

Bridging technology employs different versions of spanning tree algorithms to support different bridging methods. The common purpose of each algorithm is to produce a loop–free topology.

In the spanning tree algorithm used by Transparent Bridges (TB), Hello BPDUs and Topology Change Notification (TCN) BPDUs are sent in a transparent frame to well known group addresses of all participating media (Token ring, Ethernet, FDDI, and so forth). Tables are built from this exchanged information and a loop free topology is calculated.

Source Routing Bridges (SRB) transmit spanning tree Explorer (STE) frames across SRB bridges to determine a loop-free topology. The algorithm sends Hello BPDUs in a transparent frame to well-known functional addresses. Since TCN BDPUs are not used by SRB bridges, the port state setting created as a result of this spanning tree algorithm does not affect All Route Explorer (ARE) Frame and Specifically Routed Frame (SRF) traffic.

In bridging configurations using IBM 8209 Bridges, a different spanning tree method is used to detect parallel 8209 bridges. This algorithm uses Hello BPDUs sent as STE frames to IEEE 802.1d group addresses on the token ring. On the Ethernet, Hello BPDUs are sent as transparent frames to the same group address. This method allows 8209s to build spanning trees with Transparent Bridges and other IBM 8209 bridges. It does not participate in the SRB Spanning Tree protocol and Hello BPDUs sent by SRBs are filtered. There is no way to prevent the 8209 from becoming the root bridge. If the 8209 bridge is selected as the root, then traffic between two Transparent Bridge domains may have to pass through token ring /SRB domains.

As you can see, running multiple Spanning Tree protocols can cause compatibility problems with the way each algorithm creates its own loop-free topology.

STP/8209

The STB/8209 bridging feature is available to allow you to further extend the Spanning Tree protocol. Previously, SRB bridges allowed only manual configuration of a loop–free tree over the token ring. This was the only mechanism to prevent loops in the case of parallel SR-TB bridges. With the addition of the STP/8209 feature the following spanning tree algorithm combinations are possible:

- **Pure Transparent Bridge (TB)** – IEEE 802.1d Spanning Tree protocol is used.
- **Pure Source Routing Bridge (SRB)** – SRB Spanning Tree protocol is used.
- **Transparent and Source Routing Bridges as separate entities** – IEEE 802.1d Spanning Tree protocol is used for TB and SRB Spanning Tree protocol is used for SRB.
- **ST–TB Bridge** – IEEE 802.1d Spanning Tree protocol is used for TB ports and IBM 8209 BPDUs on SRB ports are used to form a single tree of TBs and SR–TBs. SRB Hello BPDUs are allowed to pass on the SR domain but are not processed.

  IBM 8209 bridges filter such frames but this is allowed as it is a two–port bridge with the other port being a TB port.

- **Pure SRT Bridge** – *Only* IEEE 802.1d Spanning Tree protocol is used. SRB Hello BPDUs and IBM 8209 BPDUs are allowed to pass but are not processed.

- **ASRT Bridge** – IEEE 802.1d Spanning Tree protocol is used to make a tree with TBs and SRT bridges. "8209–like" BPDUs are also generated on all SR interfaces.

  These BPDUs are processed as soon as they are received. This causes two BPDUs to be generated and received on all SR interfaces. Since both BPDUs carry the same information, there is no conflict of port information. This lets the ASRT bridge create a spanning tree with IBM 8209 and SR–TB bridges along with other TBs and SRT bridges.

Logical Link Class 2 Support

In LANs, the data link layer is comprised of two sublayers: the medium access control (MAC) and the link layer control (LLC). LLC provides two types of services:

- **LLC1 (Type 1)** – An unacknowledged connectionless service

- **LLC2 (Type2)** – A connection-oriented service

LLC2 provides the following capabilities:

- Initiating new data link connections

- Managing data link connections

- Exchanging data in sequential order (in a guaranteed fashion)

- Executing a level of flow control on the established connections

- Terminating link connections upon request from the service user or unrecoverable link errors.

The LLC sublayer adheres to the IEEE 802.5 standard.

Threading

Threading is a process where the network protocol (IPX, DNA, IP, and AppleTalk) of the Token ring end station discovers a route over segments of a Source-Routing Bridge Network.

Threading is similar to the Source Routing Bridge operation. It is how threading is implemented by the end station that is different. The following sections describe threading for IP, DECnet, IPX, and AppleTalk.

IP Threading with ARP

IP end stations use ARP REQUEST and REPLY packets to discover a RIF. Both IP end stations and the bridges participate in the route discovery and forwarding process. The following steps describe the IP threading process.

1. An IP end station maintains an ARP table and a RIF table. The MAC address in the ARP table is used as a cross-reference for the destination RIF in the RIF table. If a RIF does not exist for that specific MAC address, the end station transmits an ARP REQUEST packet with an ARE (All Routes Explore) or an STE (Spanning Tree Explore) onto the local segment.

2. All bridges on the local segment capture the ARP REQUEST packet and send it over their connected networks.

   As the ARP REQUEST packet continues its search for the destination end station, each bridge that forwards it adds its own bridge number and segment number to the RIF in the packet. As the frame continues to pass through the bridged network, the RIF compiles a list of bridge and segment number pairs describing the path to the destination.

   When the ARP REQUEST packet finally reaches its destination, it contains the exact sequence of bridge and segment numbers from source to destination.

3. When the destination end station receives the frame, it places the MAC address and its RIF into its own ARP and RIF tables. If the destination end station receives any other ARP REQUEST packets from the same source, that packet is dropped.

4. The destination end station then generates an ARP REPLY packet including the RIF and sends it back to the source end station.

5. The source end station receives the learned route path. The MAC address and its RIF are then entered into the ARP and RIF tables. The RIF is then attached to the data packet and forwarded onto the destination.

6. Aging of RIF entries is handled by the IP ARP refresh timer.

DECnet (DNA) Threading

DECnet end stations use ARE to discover a route. Both the DNA end stations and the bridges participate in the route discovery process and forwarding. The following steps describe the DECnet threading process.

1. If there is no entry in the RIF table for the MAC address, an entry is created with the state NO_ROUTE. When this occurs the end station sends the data packet out with an STE attached. The STE is used for discovery without attempting to flood the network with ARE.

2. The end station then transmits an ARE in a loop-back frame for the destination MAC address.

3. All bridges on the local segment capture the STE and loop-back frame and send it over their connected networks.

   As the packets continue their search for the destination end station, each bridge that forwards it adds its own bridge number and segment number to the RIF in the STE and the ARE. As the frame continues to pass through the bridged network, the RIF compiles a list of bridge and segment number pairs describing the path to the destination.

   When the STE and loop-back frame finally reaches the destination, it contains the exact sequence of bridge and segment numbers from the source to the destination.

4. When the destination end station receives the loop-back frame, it places the MAC address and the RIF of the source station into its own RIF table. If a RIF already exists for that entry, it either updates the RIF if that previous entry is an ST_ROUTE (refer to step 7) or it ignores the RIF. In any case, the entry state is changed to HAVE_ROUTE.

5. The destination end station then sends the loop-back reply frame including the specific RIF back to the source end station.

6. The source end station receives the learned specific route path. The RIF is entered into the RIF table and the entry is changed to HAVE_ROUTE.

7. Packets destined for a functional address are sent with an STE. DNA end stations can create a RIF entry using this STE frame. When this happens, the state of the entry is changed to ST_ROUTE. This type of route is less desirable and is replaced as described in step 4.

The DNA end stations contain an independent RIF timer. When this timer expires for a specific RIF entry, an ARE in a loop-back packet is sent out to that specific destination. When the loop-back frame returns, the RIF entry is updated. If the destination end station is on the same ring and the loop-back frame contains no RIF, the loop-back packet is returned with no RIF entry.

IPX Threading

IPX end stations check each packet they receive for a RIF. If the RIF does not exist in the table, they add the RIF to the table and designate that route as HAVE_ROUTE. If the RIF indicates that the packet came from an end station on the local ring, the route is designated as ON_RING.

If the end station needs to send out a packet and there is no entry in the RIF table for the MAC address, the end station transmits the data as an STE.

When the RIF timer expires, the entry in the table is cleared and is not reentered until another packet arrives containing a RIF for that entry.

AppleTalk 1 and 2 Threading

AppleTalk end stations use ARP and XID packets to discover a route. Both the AppleTalk end stations and the bridges participate in the route discovery process and forwarding. The following steps describe the AppleTalk threading process.

1. If a RIF does not exist for a specific MAC address, the end station transmits an ARP REQUEST packet with an ARE (All Routes Explore) onto the local segment.

2. All bridges on the local segment capture the ARP REQUEST packet and send it over their connected networks.

As the ARP REQUEST packet continues its search for the destination end station, each bridge that forwards it adds its own bridge number and segment number to the RIF in the packet. As the frame continues to pass through the bridged network, the RIF compiles a list of bridge and segment number pairs describing the path to the destination.

3. When the destination end station receives the frame, it places the MAC address and its RIF into its own ARP and RIF tables. The state of the entry is designated as HAVE_ROUTE. If the destination end station receives any other ARP REQUEST packet from the same source, that packet is dropped.

4. The destination end station then generates an ARP REPLY packet (including the RIF) and sends it back to the source end station with the direction bit in the RIF flipped.

5. The source end station receives the learned route path. The MAC address and its RIF are then entered into the ARP and RIF tables. The state is designated as HAVE_ROUTE. If the RIF indicates that the packet came from an end station on the local ring, the route is designated as ON_RING.

6. If the RIF timer expires, an XID is sent out with an ARE and the state is changed to DISCOVERING. If no XID reply is received, the entry is discarded.

4

Configuring Bridging

This chapter describes how to configure the Adaptive Source Routing Transparent (ASRT) Bridge protocol and how to use the ASRT configuration commands.

Accessing the ASRT Configuration Environment

For information on how to access the ASRT configuration environment, see the Chapter 1 in the *Network Interface Operations Guide.*

ASRT Configuration Commands

This section summarizes and then explains the ASRT configuration commands. The ASRT configuration commands allow you to specify network parameters for the ASRT bridge and its network interfaces. These commands also allow you to enable and configure the bridge IP Tunnel, NETBIOS Filtering, and LAN Network Manager features.

Note: After you access the bridging configuration process, you may begin entering configuration commands. Whenever you make a change to a user-configurable interface parameter, you must restart the router for this change to take effect.

Enter the ASRT configuration commands at the `ASRT config>` prompt. Configuration commands for IP tunnels are entered at the `tnl config>` prompt. The tunnel prompt is a subset of the major ASRT commands and is accessed by entering the ASRT **tunnel** command (explained below).

Configuration commands for the LNM (LAN Network Manager) feature are entered at the `LNM config>` prompt. This prompt is a subset of the major ASRT commands and is accessed by entering the ASRT **lnm** command (explained below).

Configuration commands for NETBIOS Filtering are entered at the `NETBIOS Filter config>` prompt. This prompt is a subset of the major ASRT commands and is accessed by entering the ASRT **netbios-filter** command (explained below).

Table 4–1 ASRT Configuration Command Summary

Command	Function
? (Help)	Lists all of the ASRT configuration commands, or lists the options associated with specific commands.
Add	Adds station address entries to the permanent database, specific address mapping, LAN/WAN ports, protocol filters, and a tunnel between end stations across an IP internetwork.
Change	Allows the user to change bridge and segment numbers.
Delete	Deletes station address entries, specific address mapping, LAN/WAN ports, protocol filters, and a tunnel between end stations across an IP internetwork.
Disable	Disables bridging functionality, duplicate frames, mapping between group and functional addresses, propagation of Spanning Tree Explorer Frames, source-routing on a given port, reception of spanning tree explorer frames over a tunnel, conversion of source routed frame to transparent frame, transparent (spanning tree) bridging functionality on a given port, and a tunnel between bridges.
Enable	Enables bridging functionality, duplicate frames, mapping between group and functional addresses, propagation of Spanning Tree Explorer Frames, source-routing on a given port, reception of spanning tree explorer frames over a tunnel, conversion of source routed frame to transparent frame, transparent (spanning tree) bridging functionality on a given port, and a tunnel between bridges.
List	Displays information about the complete bridge configuration or about selected configuration parameters.
LNM	Allows access to the LNM (LAN Network Manager) configuration prompt so that LNM configuration commands can be entered.

(continued on next page)

Table 4–1 (Cont.) ASRT Configuration Command Summary

Command	Function
Netbios-filtering	Allows access to the NETBIOS filtering configuration prompt so that NETBIOS filtering configuration commands can be entered.
Set	Sets aging time for dynamic address entries, bridge address, maximum frame size for tunneling, Largest Frame (LF) bit encoding, maximum frame size, Spanning Tree protocol bridge and port parameters, Route Descriptor (RD) values, and filtering database size.
Tunnel	Allows access to the tunnel configuration prompt so that tunnel configuration commands can be entered.
Exit	Exits the ASRT configuration process and returns to the CONFIG environment.

? (Help)

List the commands that are available from the current prompt level. You can also enter a **?** after a specific command name to list its options.

Syntax: ?

```
Example: ?
```

Add

Add the following information to your bridging configuration:

- Station address entries to the permanent database
- Specific address mapping for a given protocol
- LAN/WAN ports
- Protocol filters that selectively filter packets based on their protocol type

For the bridge’s IP tunnel feature, the **add** command lets you create an IP tunnel between end stations across an IP internetwork. This tunnel is counted as one hop between the end stations, no matter how complex the path through the IP internet.

Syntax:	add	address . . .
		mapping . . .
		port . . .
		prot-filter . . .
		tunnel . . .

address ***addr-value***

Adds unique station address entries to the permanent database. These entries are copied into the filtering database as permanent entries when the bridge is restarted. The *addr-value* is the MAC address of the desired entry. It can be an individual address, multicast address, or broadcast address. You are also given the option to specify the outgoing forwarding port map for each incoming port.

Permanent database entries are not destroyed by the power off/on process and are immune to the aging settings. Permanent entries cannot be replaced by dynamic entries.

Example: **add address**

```
Address (in 12-digit hex) []?
Exclude destination address from all ports?(Yes or No):
Use same output port mapping for all input Ports?(Yes or No):
Output port mapping:
   Input Port Number [1]?
   Bridge to all ports?(Yes or No):
   continue to another input port? (Yes or No):
   Input Port Number [2]?
   Bridge to all ports?(Yes or No): n
   Bridge to  port 1 - Yes or No: y
   Bridge to  port 2 - Yes or No: n
   Bridge to  port 3 - Yes or No: y
   continue to another input port? (Yes or No): y
   Input Port Number [3]? 3
   Bridge to all ports?(Yes or No): y
   continue to another input port? (Yes or No): y
Source Address Filtering Applies? (Yes or No): y
```

Note: For any Yes or No question in the prompts, No is the default value. Press RET to enter the default value.

`Exclude destination address ...`	This prompt lets you set destination address filtering for that entry. Answering Yes to the prompt causes filtering of any frames that contain this address as a destination address – no matter which port it came from.
`Use same output mapping...`	Answering Yes to this prompt lets you create *one* outgoing port map for all incoming ports, rather than allowing for mapping to only specific ports. Answering No to this prompt causes further prompting `(Input Port Number [1]?)` to select each input port. From that specific input port prompt, you can then create a unique port map for that input port.
`Input Port 1, Port 2`	Answering No to the previous prompt causes input port-by-input port prompting `(Input Port Number [1]?)` to select each input port and its associated outgoing bridge ports.
`Bridge to all ports?`	Answering Yes to this prompt creates an outgoing port map that includes all ports. When a frame with this address as the destination address is received, it is forwarded to all outgoing forwarding ports except for the incoming port. The following are examples of how this is done according to the port map:

- If a frame is received on port 1 and the port map indicates 1 (for port 1), the frame is filtered.
- If the same frame is received on port 2 and the port map indicates 1 (for port 1), the frame is forwarded to port 1. If a frame is received on port 1 and the matching address entry's port map indicates 1, 2, or 3, the frame is forwarded to ports 2 and 3.
- If the port map indicates no port (NONE/DAF) then the frame is filtered. This is known as destination address filtering (DAF).
- If no address entry is found to match the received frame, it is forwarded to all the forwarding ports, except for the source port).

`Bridge to Port 1, Port 2, etc.`	This prompt lets you associate an address entry with that specific bridge port. Entering Y (for yes) after the prompt maps the address to the specified port to include that port in that address entry's port map. Entering N skips address mapping for that port.
`continue to another bridge port?`	This prompt lets you select the next input port to be configured.
`Source address filtering`	This allows for port-specific address filtering. When SAF is applied (Yes is entered at the prompt), frames received with source addresses that match address entries in the filtering database that have source address filtering enabled is discarded. This mechanism allows a network manager to isolate an end station by prohibiting its traffic to be bridged.

The following sections present specific examples of how the **add address** command is used to manage address entries:

Destination Address Filtering Enabled For Entry

This example shows how to answer the command prompts to select destination address filtering for an entry:

```
ASRT config>**add address 000000334455**
Filter exclusively, no matter what input port?(Yes or No): **y**
Source Address Filtering Applies? (Yes or No): **y**
ASRT config>
```

After adding the address entry, you can verify its status by using the **list range** command. The example below shows that no port map exists for that entry (in bold) and that destination address filtering (DAF) was turned on.

```
ASRT config>list range
Start-Index [1]?
Stop-index [3]?
ADDRESS                ENTRY TYPE      PORT MAP
=======                ==========      ========
01-80-C2-00-00-00      REGISTERED      Input Port:  ALL PORTS
                                       Output ports:


00-00-00-22-33-44      PERMANENT       Input Port:  3
                                       Output ports: 1, 2
                                       Input Port:  4
                                       Output ports:  1, 2


0-0-0-33-44-55      PERMANENT      NONE/DAF
```

Output Port Map Created For Address Entry Having More Than One Input Port

This example shows how to answer the command prompts to create separate output port maps for an address entry that has more than one input port.

```
ASRT config> add address 000000123456
Filter exclusively, no matter what input port?(Yes or No): n
Single output port map for all input Ports?(Yes or No): n
Input Port Number [1]? 1
All Ports?(Yes or No): n
Port[1] – Yes or No: y
Port[2] – Yes or No: y
Port[3] – Yes or No: n
Port[4] – Yes or No: n
continue? (Yes or No): y
Input Port Number [2]? 2
All Ports?(Yes or No): n
Port[1] – Yes or No: n
Port[2] – Yes or No: n
Port[3] – Yes or No: y
Port[4] – Yes or No: y
continue? (Yes or No): n
Source Address Filtering Applies? (Yes or No): n
ASRT config>
```

After adding the address entry, you can verify its status by using the **list range** command. The following example shows an entry (in bold) that has ports 1 and 2 as input ports and has separate port maps for both input ports. Source address filtering (SAF) is also enabled.

```
ASRT config> list range
Start-Index [1]?
Stop-index [3]?
ADDRESS                ENTRY TYPE      PORT MAP
=======                ==========      ========
01-80-C2-00-00-00      REGISTERED      Input Port:  ALL PORTS
                                       Output ports:

01-80-C2-00-00-01      RESERVED        NONE/DAF


0-0-0-12-34-56         PERM/SAF        Input Port:  1
                                       Output ports: 1, 2
                                       Input Port:  2
                                       Output ports: 3, 4
```

Single Output Port Map Created All Incoming Ports Associated With Address Entry

This example shows how to answer the command prompts to create a single output port map for all incoming ports associated with an address entry.

```
ASRT config> add address 000000556677
Filter exclusively, no matter what input port?(Yes or No): n
Single output port map for all input Ports?(Yes or No): y
All Ports?(Yes or No): n
Port[1] – Yes or No: y
Port[2] – Yes or No: y
Port[3] – Yes or No: n
Port[4] – Yes or No: y
Source Address Filtering Applies? (Yes or No): y
ASRT config>
```

After adding the address entry, you can verify its status by using the **list range** command. The following example shows an entry (in bold) that has a single port map for all incoming ports. Source address filtering (SAF) has also been enabled.

```
ASRT config> list range
Start-Index [1]?
Stop-index [3]?
ADDRESS                ENTRY TYPE      PORT MAP
=======                ==========      ========
01-80-C2-00-00-00      REGISTERED      Input Port:  ALL PORTS
                                       Output ports:

01-80-C2-00-00-01      RESERVED        NONE/DAF



0-0-0-55-66-77         PERM/SAF        Input Port:  ALL PORTS
                                       Output ports:  1, 2, 4
```

mapping *dlh-type type-field ga-address fa-address*

Adds specific functional address to group address mapping for a given protocol identifier. The address mapping is converted only on destination addresses crossing token ring to ethernet/FDDI or vice versa.

Note: For every Ether-type mapped value, the corresponding SNAP-type value is added. This is necessary for bidirectional mapping.

dlh-type	(data-link-header type) is a choice for DSAP, Ether-type, or SNAP.
type-field	Protocol type field: • Destination Service Access Point (DSAP) protocol type is entered in a range of 1–FE hexadecimal). • Ethernet (Ether) protocol type is entered in a range of 5DD-FFFF (hexadecimal). • Subnetwork Access Protocol (SNAP) protocol type is entered in 10-digit hexadecimal format.
ga–address	6-byte (12-digit hexadecimal) group/multicast address.
fa–address	Functional address in non-canonical format. Functional addresses are locally administered group addresses. These are most commonly used in token ring networks.

Note: The most commonly used values for DECnet group address-to-functional address mapping are listed as follows:

Ethertype	*Group Address*	*Functional Address*
6002	ab–00–00–02–00–00	C0:00:20:00:00:00
6003	ab–00–00–03–00–00	C0:00:10:00:00:00
6003	ab–00–00–00–04–00	C0:00:08:00:00:00

SNAP	*Group Address*	*Functional Address*
00–00–00–6002	ab–00–00–02–00–00	0:00:20:00:00:00
00–00–00–6003	ab–00–00–03–00–00	C0:00:10:00:00:00
00–00–00–6003	ab–00–00–04–00–00	C0:00:08:00:00:00

Example: **add mapping dsap**

```
Protocol Type in hex (1 – FE) [1]?
Group-Address (in 12-digit hex) [ ]?
Functional address (in non-canonical format) [ ]?
```

Example: **add mapping ether**

```
Protocol Type in hex (5DD - FFFF) [0800]?
Group-Address (in 12-digit hex) []?
Functional address (in non-canonical format) [ ]?
```

Example: **add mapping snap**

```
Address (in 10–digit hex) [0000000800]?
Group-Address (in 12-digit hex) []?
Functional address (in non-canonical format) [ ]?
```

port *interface# port#*

Adds a LAN/WAN port to the bridging configuration. This command associates a port number with the interface number and enables that port's participation in transparent bridging.

Example: **add port 0 4**

```
Interface Number [0]?
Port Number [5]?
```

prot-filter s̲nap e̲ther d̲sap

Allows the bridge to be configured so that it can selectively filter packets based on their protocol type. Filters can be applied to all ports or only selected ports.

This parameter specifies protocol identifiers for which the received frames of that specific protocol are discarded exclusively without applying bridge logic. ARP packets for this protocol type is also discarded. The protocol filter is applied only on the received packets. The protocol filters available include the following:

SNAP Packets	Subnetwork Access Protocol with protocol type entered in 10-digit hexadecimal format.
Ether Packets	Ethernet Type with the protocol type entered in a range of 5DD-FFFF (hexadecimal).
DSAP Packets	Destination Service Access Point protocol with the protocol type entered in a range of 1-FE (hexadecimal).

The routing protocols that are enabled in the router (the ones that are displayed by the configuration command in GWCON) cannot be added for filtering. Common protocol filters and their respective values are displayed below.

DSAP Types

Protocol	SAP (hexadecimal value)
Banyan SAP	BC (used only for 802.5)
Novell IPX SAP	E0 (used only for 802.5)
Net BIOS SAP	F0
ISO Connectionless Internet	FE

SNAP Protocol Identifiers

Protocol	*SNAP OUI/IP (10-digit)*
AppleTalk Phase 2	08-00-07-80-9B
Apple ARP Phase 2	00-00-00-80-F3
Proprietary AppleTalk Phase 1 for FDDI	00-00-93-00-02
Proprietary AppleTalk ARP Phase 1 for FDDI	00-00-93-00-03

Ethernet Types

Protocol	*Ethernet type (hex value)*
IP	0800
ARP	0806
CHAOS	0804
DECnet MOP Dump/Load	6000
DECnet MOP Remote Console	6002
DECnet	6003
DEC LAT	6004
DEC LAVC	6007
XNS	0600
Maintenance Packet Type	7030
Apollo Domain	8019 (Ethernet)
Novell NetWare IPX	8137 (Ethernet)
AppleTalk Phase 1	809B

Apple ARP Phase 1	80F3
Loopback assistance	9000

Example: **add prot-filter dsap (used for DSAP packets)**

```
Protocol Type in hex (1 – FE) [1]?
   Filter packets arriving on all ports?(Yes or No):
   Filter packets arriving on  port 1 – Yes or No:
   Filter packets arriving on  port 2 – Yes or No:
   Filter packets arriving on  port 3 – Yes or No:
```

Example: **add prot-filter ether (used for Ethernet packets)**

```
Protocol Type in hex (600 – FFFF) [0800]?
   Filter packets arriving on all ports?(Yes or No):
   Filter packets arriving on  port 1 – Yes or No:
   Filter packets arriving on  port 2 – Yes or No:
```

Example: **add prot-filter snap (used for SNAP packets)**

```
Address (in 10–digit hex) [0000000800]?
   Filter packets arriving on all ports?(Yes or No):
   Filter packets arriving on  port 1 – Yes or No:
   Filter packets arriving on  port 2 – Yes or No:
   Filter packets arriving on  port 3 – Yes or No:
```

tunnel ***port#***

Creates the user-defined IP tunnel to a bridge port. This tunnel provides a passage for a bridged frame through an IP internetwork. This tunnel is counted as one hop between the bridges, no matter how complex the path through the IP internet. To use the tunnel feature, the IP forwarder must be enabled.

Only one tunnel can be added. It is required that for the port# you use one that is not used for any other LAN/WAN port. Internally, the interface number *255* is ascribed to mark that interface as connected as a “virtual” interface.

Transparent bridging is enabled on this port by default. Source routing can be enabled by using the **enable source-routing** command.

Example: **add tunnel 3**

```
Port Number   [1] ? 3
```

Port Number	A unique port number not being used by the bridge.

Change

Change source routing bridge and segment numbers in the bridging configuration.

Syntax: change bridge . . .
segment . . .

bridge *new-bridge#*

Changes bridge numbers in the bridging configuration.

Example: **change bridge 3**

segment *old-segment# new-segment#*

Changes bridge numbers in the bridging configuration.

Example: **change segment 2 3**

Delete

Delete the following information from your bridging configuration:

- Station address entries to the permanent database
- Specific address mapping for a given protocol
- LAN/WAN ports
- Protocol filters that selectively filter packets based on their protocol type

For the IP tunnel feature, the **delete port** command with the corresponding port number for the tunnel removes the tunnel between bridges across an IP internetwork.

Syntax:	delete	address
		mapping . . .
		port . . .
		prot-filter . . .

address ***addr-value***

Deletes an address entry from the permanent database. The address is the MAC address of the desired entry.

Enter the addr-value (in 12-digit hexadecimal format) of the entry to be deleted and press **RETURN**. Reserved multicast addresses cannot be deleted. If you attempt to delete an address entry that does not exist, you receive the message: `Record matching that address not found`.

Example: **delete address**

mapping ***dlh-type type-field ga-address***

Deletes specific address mapping for given protocol.

dlh-type	(Data-link-header type) is a choice for DSAP, Ether-type, or SNAP.
type-field	Protocol type field. • Destination Service Access Point (DSAP) protocol type is entered in a range of 1 – FE (hexadecimal). • Ethernet (Ether) protocol type is entered in a range of 5DD-FFFF (hexadecimal). • Subnetwork Access Protocol (SNAP) protocol type is entered in 10-digit hexadecimal format.
ga–address	6-byte (12-digit hexadecimal) group/multicast address.

Example: **delete mapping DSAP FE *<group address>***

port ***port#***

Removes a port from a bridging configuration. Since the **enable bridge** command by default configures all LAN devices to participate in bridging, this command allows you to determine which devices do or do not participate in the bridging. The port number value normally is one greater than the interface number.

This command followed by the IP tunnel port# removes an IP tunnel from a bridging configuration.

`Example:` **`delete port 2`**

prot-filter **snap** **ether** **dsap**

Deletes previously specified protocol identifiers used in filtering. You can delete filters for all ports or selected ports. These filters include the following:

SNAP Packets	Subnetwork Access Protocol with protocol type entered in 10-digit hexadecimal format.
Ether Packets	Ethernet Type with the protocol type entered in a range of 5DD–FFFF (hexadecimal).
DSAP Packets	Destination Service Access Point protocol with the protocol type entered in a range of 1–FE (hexadecimal).

`Example:` **`delete prot-filter snap (used for SNAP packets)`**

```
Address (in 10–digit hex) [0000000800]?
   Delete filter on all ports?(Yes or No):
   Delete filter on  port 1 – Yes or No:
   Delete filter on  port 2 – Yes or No:
   Delete filter on  port 3 – Yes or No:
```

`Example:` **`delete prot-filter ether (used for Ethernet packets)`**

```
Protocol Type in hex (600 – FFFF) [0800]?
   Delete filter on all ports?(Yes or No):
   Delete filter on  port 1 – Yes or No:
   Delete filter on  port 2 – Yes or No:
```

Example: **delete prot-filter dsap (used for DSAP packets)**

```
Protocol Type in hex (1 – FE) [1]?
   Delete filter on all ports?(Yes or No):
   Delete filter on  port 1 – Yes or No:
   Delete filter on  port 2 – Yes or No:
   Delete filter on  port 3 – Yes or No:
```

Disable

Disable the following bridge functions:

- Bridging functionality entirely
- Creating duplicate frames for mixed bridging environments (network traffic management)
- Mapping between group address and functional address
- Propagating spanning tree explorer frames
- Source routing on a given port
- Receiving spanning tree explorer frames over a tunnel
- Converting source routed frames to transparent frames and vice versa
- Transparent (spanning tree) bridging functionality on a given port

For the tunnel feature, the **disable** command disables a tunnel between end stations across an IP internetwork.

```
Syntax:   disable   bridge
                    duplicate . . .
                    fa-ga-mapping
                    IBM8209_Spanning_Tree
                    spanning-tree-explorer  . . .
                    source-routing  . . .
                    sr-tb-conversion
                    transparent  . . .
```

bridge

Disables bridging functionality entirely. This command does not remove previously configured bridging values.

Example: **disable bridge**

duplicate *frame-type*

Disables the creation of duplicate frames present in mixed bridging environments. When the SR-TB bridging feature is enabled on an 802.5 interface (with source-routing and transparent bridging enabled), there are inconsistencies created when bridging frames to an unknown (or multicast) destination. It is not known to the bridge whether the destination is behind a source-routing (only) or transparent bridge.

To remedy this situation, the bridge sends out duplicates of these frames (by default). One frame has source-routing fields present (a Spanning Tree Explorer RIF) and the other is formatted for transparent bridging (no RIF is present). The **disable duplicate** command lets you eliminate this duplication by allowing you to disable the creation of one of these types of frames.

Entering STE after the command tells the bridge to refrain from sending out Spanning Tree Explorer frames created for the source-routing environment. Entering TSF after the command tells the bridge to refrain from sending out Transparent Spanning Frames for the transparent bridging environment. In both cases, it is a situation where normally both types of frames are sent out. Disabling transparent bridging on the interface also disables the creation of transparent frames.

`Example:` **`disable duplicate TSF`**

fa-ga-mapping

Disables group address-to-functional address (and vice versa) mapping. You can under certain circumstances want to disable the mapping between group address and functional address globally.

`Example:` **`disable fa-ga-mapping`**

IBM8209_Spanning_Tree

Removes bridges from participating in spanning tree protocols with IBM 8209 bridges.

`Example:` **`disable IBM8209_spanning_tree`**

spanning-tree-explorer ***port#***

Disables a port from allowing propagation of Spanning Tree Explorer Frames if source routing is enabled. This command is used only if transparent bridging is not enabled on the port. In that case, it is automatically known in conformance with the transparent spanning tree.

`Example:` **`disable spanning-tree-explorer 2`**

source-routing ***port#***

Disables source routing on a given port. This command enables an already participating bridge interface to discontinue source routing.

`Example:` **`disable source-routing 2`**

sr-tb-conversion

Disables conversion of source routed frame to transparent frame and vice-versa.

`Example:` **`disable sr-tb-conversion`**

transparent ***port#***

Disables transparent bridging functionality on the given port. This command is useful for cases where an alternative communication method such as source routing is desirable.

Note: This command might bring about an absurd configuration if not used properly. For instance, using it on an ethernet interface results in disabling bridging functionality for that interface. This command is used to bring about SRB and SR-TB bridge functionality.

```
Example: disable transparent 2
```

Enable

Enable the following bridging functions:

- Bridging functionality entirely
- Creating duplicate frames for mixed bridging environments (network traffic management)
- Mapping between group address and functional address
- Propagating spanning tree explorer frames
- Source routing on a given port
- Receiving spanning tree explorer frames over a tunnel
- Converting source routed frame to transparent frame
- Transparent (spanning tree) bridging functionality on a given port

For the IP tunnel feature, the **enable** command enables a tunnel between end stations across an IP internetwork.

Syntax: enable bridge . . .
duplicate
fa-ga-mapping
IBM8209_Spanning_Tree
spanning-tree-explorer . . .
source-routing . . .
sr-tb-conversion
transparent . . .

bridge

Enables transparent bridging functionality on all the LAN devices (interfaces) configured in the bridging router. The port numbers are assigned to each interface as the previous interface number plus 1. For example, if interface 0 is a LAN device its respective port number is 1.

Example: **enable bridge**

duplicate

Enables the generation of duplicate STE (Spanning Tree Explorer) or TSF (Transparent Spanning Frames) frames. This command is available to offset the **disable duplicate** command. Duplicate frame generation is enabled by default.

Example: **enable duplicate**

fa-ga-mapping

Enables group address to functional address (and vice versa) mapping. This mapping is conducted when frames are forwarded between token ring and other media (except serial line). In token ring arenas, functional addresses are more popular, even though they are locally assigned group addresses due to restrictions in hardware. On other media, group addresses are widely used. Under normal circumstances, group address to functional address mapping is inevitable.

Mapping is enabled by default if mapping addresses are added. The **enable/disable mapping** lets users have a choice when it comes to deleting added map records.

Example: **enable fa-ga-mapping**

IBM8209_Spanning_Tree

Allows bridges to participate in Spanning Tree protocols with IBM 8209 bridges.

Example: **`enable IBM8209_spanning_tree`**

spanning-tree-explorer ***port#***

Enables the port to allow propagation of Spanning Tree Explorer Frames if source routing is enabled. This command is valid on token ring and WAN ports only. This feature is enabled by default when source routing is configured on the port.

Example: **`enable spanning-tree-explorer 2`**

source-routing ***port# segment# [bridge#]***

Enables source routing for a given port. This command is typically used when source routing on part of the bridge is desired. If source routing is the only feature desired, transparent bridging on the interface is disabled. For the first instance of the command, entering the bridge number is required. For subsequent times, this input is not required.

port#	Valid port participating in the bridge configuration.
segment#	12-bit number which represents the LAN/WAN to which media is attached. All the media on other bridges attached to this LAN/WAN must be configured with the same value. For correct operations of source routing functionality, it is very important that all the bridges attached to this LAN/WAN have the same perspective of the LAN/WAN identification value.
bridge#	4-bit value unique among all the bridges attached to the same LAN/WAN. This value is required when source routing is enabled on the first interface. For later interfaces, this input is optional. It is recommended that the bridge# be unique on the segment.

Note: If the configuration is a situation where two segments are already configured (a 1:N SRB configuration), you are prompted for an additional virtual-segment# parameter.

Example: **enable source-routing 2 1**

sr-tb-conversion

Allows for compatibility between Source Routing and Transparent Bridging domains. When this feature is enabled, the bridge lets source-routed frames be accepted into a transparent domain by stripping off the RIF field and converting them into transparent frames.

The bridge also gathers routing information concerning source routing stations from the passing source routing frames. This is obtained from the RIF. This RIF information is then used to convert a transparent frame to a source-routed frame. If an RIF is not available for a station, then the frame is sent out as a spanning tree explorer frame in the source routing domain.

In order for the conversion functionality to operate properly, the transparent bridging domain must be given a segment number. All SR-TB bridges that are connected to this domain should also be configured with the same segment number.

Example: **enable sr-tb-conversion**

transparent ***port#***

Enables transparent bridging functionality on the given port. Under normal circumstances, this command is not necessary. This command toggles the **disable transparent** command.

List

Display information about the complete bridge configuration or to display information about selected configuration parameters.

Syntax: list address
bridge
filtering . . .
mapping . . .
permanent . . .
port . . .
prot-filter . . .
protocol
range . . .

address *addr value*

Reads an address entry from the permanent database. The *addr value* is the MAC address of the desired entry.

Example: **list address 000000123456**

```
   00-00-00-12–34–56    PERMANENT  Input Port:  ALL PORTS
                                   Output ports:  1, 2, 3, 4
```

Example: **list address 000000112233**

```
   00–00–00–11–22–33     PERM/SAF    Input Port:  1
                                     Output ports:  1, 2
```

Address	Address entry in 12-digit hexadecimal format.
Entry Type	PERMANENT indicates that the entry is permanent in nature and survives power cycles or system resets. RESERVED indicates that the entry is reserved by the IEEE802.1d committee for future use. Frames destined to reserved addresses are discarded. REGISTERED indicates that the entry is meant for the bridge itself. SAF appears after the entry type if source address filtering is configured.

Input Port	Displays the numbers of input port(s) associated with that address entry.
Output Port	Displays the numbers of output port(s) associated with that address entry. Displays NONE/DAF to indicate that destination address filtering applies because no ports were selected to be associated with that address entry.

bridge

Lists all general information regarding the bridge.

Example: **list bridge**

```
              Source Routing Transparent Bridge Configuration
              ===============================================


Bridge:  ENABLED                        Bridge Behaviour: ADAPTIVE SRT
Bridge Address: 00–00–00–00–00–06       Bridge Priority:(dec) 32768, (hex): 8000
Source Routing Bridge Number: A         No. of Source routing segments: 2
SRB: Max ARE Hop cnt: 14                Max STE Hop cnt: 14
SR–TB Conversion: ENABLED  TB–Virtual Segment:0x107   MTU for TB–Domain: 1470
1:N Source Routing: ACTIVE              Internal–Virtual Segment:0xFF6
SRB LF–bit interpretation: EXTENDED     FA <=> GA Conversion: ENABLED
Spanning Tree protocol Participation: IEEE802.1d
Number of ports added: 5
Port Number:  1    Interface Number: 0   Port Behaviour: STB Only
Port Number:  2    Interface Number: 1   Port Behaviour: STB Only
Port Number:  3    Interface Number: 2   Port Behaviour: No Bridging
Port Number:  4    Interface Number: 3   Port Behaviour: SRB Only
Port Number:  5    Interface Number: 4   Port Behaviour: STB & SRB
```

Bridge	Indicates current state of bridge. Values are ENABLED or DISABLED.
Bridge Behavior	Indicates method of bridging being used by that bridge. The values include STB for Transparent, SRB for Source Routing, and SR-TB for Source Routing-Transparent conversion bridging.
Bridge address	Bridge address specified by the user (if set).

Bridge priority	A high-order 2-octet bridge address found in the Bridge Identifier – either the MAC address obtained from the lowest number port or the address set by the Set Bridge command.
Source Routing Bridge Number	The unique number identifying a bridge. It is used to distinguish between multiple bridges connecting the same two rings.
Number of Source Routing Segments	Indicates the number of Source Routing bridge segments configured for the Source Routing domain.
SRB: Max ARE/STE Hop cnt	The maximum hop count for frames transmitting from the bridge for a given interface associated with source routing bridging.
SR-TB Conversion	Indicates whether the source routing/transparent bridge frame conversion function is enabled or disabled.
TB-Virtual Segment	Indicates the segment number of the transparent bridging domain.
MTU for TB-Domain	Specifies the maximum frame size (maximum transmission units) the transparent bridge can transmit and receive.
1:N Source Routing	Indicates the current state of 1:N Source Routing as ACTIVE or NOT ACTIVE.
Internal Virtual Segment	Displays the virtual segment number configured for 1:N SRB bridging.
SRB LF-bit interpretation	Indicates the largest Frame (LF) bit encoding interpretation mode if source routing is enabled in this bridge. This is listed as either BASIC or EXTENDED.
FA-GA conversion	Indicates whether FA-GA conversion is enabled or disabled.
spanning tree Protocol Participation	Displays the types of Spanning Tree protocols that the bridge participates in.
Number of ports added	Displays the number of bridge ports added to the bridging configuration.

Port Number	Specifies user-defined number assigned to an interface by the Add Port command.
Interface Number	Identifies devices connected to a network segment through the bridge. You must add at least two interfaces to participate in bridging. An interface number of 255 is used for bridging.
Port Behaviour	Indicates method of bridging being used by that port. The values include STB for Transparent, SRB for Source Routing, and SR-TB for Source Routing-Transparent conversion bridging.

filtering

Displays general information about the bridge's filtering database.

Example: **list filtering**

```
Filtering Database Size:   5120
Aging Time (in seconds):   300
Resolution (in seconds):     5
```

Filtering Data-base Size	The maximum number of entries that can be held in the filtering database.
Aging Time	Amount of time (in seconds) specified for aging out (discarding) dynamic entries in the filtering database.
Resolution	How often dynamic entries are scanned to look for expiration according to the aging timer.

mapping *add-type type-field*

Lists specific address mapping for given protocol.

Example: **list mapping SNAP**

add-type	Choice of either DSAP, Ether (Ethernet), or SNAP.

`type-field`	Protocol type field. • Destination Service Access Point (DSAP) protocol type is entered in a range of 1-FE (hexadecimal). • Ethernet (Ether) protocol type is entered in a range of 5DD-FFFF (hexadecimal). • Subnetwork Access Protocol (SNAP) protocol type is entered in 10-digit hexadecimal format.

permanent

Displays the number of entries in the bridge's permanent database.

`Example:` **`list permanent`**

```
Number of Entries in Permanent Database:  17
```

port ***port#***

Displays port information related to ports that are already configured. Port# selects the port you want to list. Specifying no number selects all ports.

`Example:` **`list port`**

```
Port Id (dec)    : 128: 1, (hex): 80–01
Port State       : ENABLED
Port Supports: Transparent Bridging Only
Assoc Interface  : 0
Path Cost        : 0
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Port Id (dec)    : 128: 2, (hex): 80–02
Port State       : ENABLED
Port Supports: Transparent Bridging Only
Assoc Interface  : 1
Path Cost        : 0
```

(continued on next page)

```
   ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   Port Id (dec)    : 128: 3, (hex): 80–03
   Port State       : ENABLED
   Port Supports: No Bridging
   Assoc Interface  : 2
   Path Cost        : 0
   ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   Port Id (dec)    : 128: 4, (hex): 80–04
   Port State       : ENABLED
   Port Supports: Source Routing Bridging Only
   SRB: Segment Number: 0x188      MTU: 4399       STE: ENABLED
   Assoc Interface  : 3
   Path Cost        : 0
   ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   Port Id (dec)    : 128: 5, (hex): 80–05
   Port State       : ENABLED
   Port Supports: Transparent Bridging and Source Routing
   SRB: Segment Number: 0x199      MTU: 4399       STE: ENABLED
   Duplicates Frames Allowed:   STE: No, TSF: Yes
   Assoc Interface  : 4
   Path Cost        : 0
   ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Port ID	The ID consists of two parts: the port priority and the port number. In the example, 128 is the priority and 1, 2, and 3 is the port number. In hexadecimal format, the low order byte denotes the port number and the high order byte denotes the priority.
Port state	Displays current state of specified port(s). This can be either ENABLED or DISABLED.
Port supports	Displays bridging method supported by that port (for example, transparent bridging, source routing bridg-ing).
SRB	Displayed only when SRB is enabled and lists source-routing bridging information. This includes the SRB segment number (in hex), the Maximum Transmission Unit size, and whether the transmission of Spanning Tree Explorer frames is enabled or disabled.
Duplicate Frames Allowed	Displays a breakdown and count of the types of duplicate frames allowed.

Assoc interface	Displays interface number associated with the displayed port.
Path Cost	Cost associated with the port which is use for possible root path cost. The range is 1 to 65535.

prot-filter *port#*

Reads a current list of the filter protocol types. Filters can be listed selectively by port or all ports can be displayed at once. Port# selects the bridge port that you want to list.

Example: **list prot-filter 1**

```
PORT 1
Protocol Class   :  DSAP
Protocol Type    :  01
Protocol State:  :  Filtered
Port Map         :  1, 2, 3
= = = = = = = = = = = = = = = = = = = = = = =
```

Port Number	Port number is displayed for each port if all ports are selected to be displayed.
Protocol Class	Displays protocol class – either SNAP, Ether, or DSAP.
Protocol Type	Displays protocol ID in hexadecimal format.
Protocol State	Denotes that protocol is being filtered for selected port.
Port Map	Displays the numbers of the ports where this type of protocol filter is present.

protocol

Displays bridge information related to the Spanning Tree protocol.

Example: **list protocol**

```
Bridge Identifiers: 8000.000000010203
Bridge-Max-Age (in seconds): 20
Bridge-Hello-Time (in seconds): 2
Bridge-Forward-Delay (in seconds): 15
```

Note: Each of these bridge-related parameters is also described in detail in the previous chapter.

`Bridge Identifier`	8-byte value in ASCII format. If you did not set the bridge address prior to displaying this information, the low order six bytes are displayed as zero denoting that the default MAC address is being used. When a bridge was selected as the root bridge, the bridge max age and bridge hello time are transmitted by it to all the bridges in the network through the HELLO BPDUs.
`Bridge Maximum Age`	Maximum age (period of time) that is used to time out Spanning Tree protocol related information.
`Bridge Hello Timer`	Time interval between HELLO BPDUs.
`Bridge Forward Delay`	Time interval used before changing to another state (if this bridge becomes the root).

range *start-index stop-index*

Reads a range of address entries from the permanent database. To do this, first determine the size of the database by using the **list permanent** command. From this value you can then determine a start index value for your entry range. The start index is one to the size of the database. You can then choose a stop index for displaying a limited number of entries. This input is optional. If the stop index is not provided the default value is the size of the database.

Address entries contain the following information:

```
Example: list range

   Start-Index [1]     1
   Stop-Index  [5] ?   6

   ADDRESS             ENTRY TYPE            PORT MAP

   00:00:00:12:13:11   REGISTERED            ALL PORTS
   00:00:00:12:13:12   RESERVED              NONE/DAF
   00:00:00:12:13:13   RESERVED              NONE/DAF
   00:00:00:12:13:14   RESERVED              NONE/DAF
   00:00:00:12:13:15   PERMANENT             NONE/DAF
   00–00–00–12–34–56   PERMANENT

   Input Port:  ALL PORTS
   Output ports:  1,2,3,4
```

Address	6-byte MAC address of the entry.
Type of Entry	Specifies one of the following types: • **Reserved** – Entries reserved by the IEEE802.1d committee • **Registered** – Entries consist of unicast addresses belonging to proprietary communications hardware attached to the box or multicast addresses enabled by protocol forwarders • **Permanent** – Entries entered by the user in the configuration process that survive power cycles or system resets • **Static** – Entries entered by the user in the console process that do not survive power cycles or system resets and are ageless • **Dynamic** – Entries learned by the bridge dynamically that do not survive power cycles or system resets and that have an age associated with the entry • **Free** – Locations in database that are free to be filled by address entries
Port Map	Displays outgoing port map for all incoming ports.

LNM

Access the LNM configuration prompt. LNM configuration commands are entered at this new prompt.

Syntax: lnm

`Example:` **`lnm`**

Netbios-filtering

Access the NETBIOS filtering configuration prompt. NETBIOS Filtering configuration commands are entered at this new prompt.

Syntax: netbios-filtering

`Example:` **`netbios-filtering`**

Note: If the NETBIOS filtering feature has not been purchased for your bridging router software load, you receive the following message if you try to use this command:

```
NETBIOS Filtering is not available in this load.
```

Set

Set certain values, functions, and parameters associated with bridge configuration. These include the following:

- Aging time for dynamic address entries in the filtering database
- Bridge address
- Largest Frame (LF) bit encoding interpretation for source routing
- MAC Service Data Unit (MSDU) size
- Spanning Tree protocol bridge and port parameters

- Route Descriptor (RD) limit

Syntax: set age
bridge
LF-bit-interpretation . . .
port
maximum-packet-size . . .
protocol bridge
protocol port . . .
route-descriptor-limit . . .

age

Sets the time for aging out dynamic entries in the filtering database when the port with the entry is in the forwarding state. This age is also used for aging RIF entries in the RIF table in the case of an SR-TB bridge personality.

Enter the desired value after each prompt and press **RETURN**. The default value for the aging timer is 300 seconds with a range of 1–1,000,000 seconds. The default value for the resolution parameter is 5 with a range of 1 to 60 seconds.

```
Example: set age

    seconds [300]  ? 300
    resolution [5] ?  5
```

bridge

Sets the bridge address. This is the low order 6-octet bridge address found in the bridge identifier. By default, the bridge-addr-value is set to the Media Access Control (MAC) address of the lowest numbered port at initialization time. You can use this command to override the use of the default address and enter your own unique address.

In cases where a serial line interface (or tunnel) is the lowest numbered port, it is mandatory to use this command so that the bridge has a unique address when restarted. This process is necessary because serial lines do not have their own MAC address.

At the prompt, enter the bridge address in 12-digit hexadecimal format and press RET. Do not use dashes or colons to separate each octet.

Note: Each bridge in the network must have a unique address for the Spanning Tree protocol to operate properly.

If you enter the address in the wrong format you receive the message `Illegal Address`. If you enter no address at the prompt, you receive the message `Zero length address supplied` and the bridge maintains its previous value. To return the bridge address to the default value, enter an address of all zeroes.

Example: **set bridge**

```
   Bridge Address (in 12-digit hex)[]?
```

LF-bit-interpretation ***encode-mode***

Sets the Largest Frame (LF) bit encoding interpretation if source routing is enabled in this bridge.

Example: **set LF-bit-interpretation basic**

Encode-mode	Entered as either basic or extended. In the basic mode only three bits of the routing control field are used. This is the common practice in source routing bridges that exist today. In extended mode, six bits of the routing control field are used to represent the maximum data unit that the bridge supports. The default value is extended. Extended and Basic nodes are compatible.

maximum-packet-size ***port#*** ***msdu-size***

Sets the largest MAC Service Data Unit (MSDU) size for the port, if source routing is enabled on this port. Obviously, MSDU value setting has no implication on traditionally transparent media. An MSDU value greater than the packet size configured in the router is treated as an error.

If this parameter is not set, the default value used is the size configured as the packet size for that interface.

Example: **set maximum-packet-size 1 4399**

```
   Port Number [1]? set ?
   MSDU size [4399]?
   MSDU reduced to 4399
```

port ***status***

Begins the port's participation in the Spanning Tree protocol. This is done by entering a status value of block. This places the port in the blocked status as a starting point. The actual state of the port is later determined by the Spanning Tree protocol as it determines its topology. Entering a status value of disable removes the port from participating in the spanning tree.

Example: **`set port block`**

protocol ***option***

Modifies the Spanning Tree protocol bridge or port parameters for a new configuration or to adjust the configuration parameters to suit a specific topology.

Enter bridge as the option to modify bridge parameters. The bridge related parameters that can be modified with this command are described below.

When setting these values, make sure that the following relationships exist between the parameters or else the input is rejected:

2 X (Bridge Forward Delay - 1 second) $\geq$ *Bridge Maximum Age*

Bridge Maximum Age $\geq$ *2 X (Bridge Hello Time + 1 second*)

Example: **`set protocol bridge`**

```
   Bridge Max-Age [20]  20
   Bridge Hello Time [2]   2
   Bridge Forward Delay [15] 15
   Bridge Priority [32768]   1
```

Bridge Maximum Age	Maximum age (period of time) used to time out Spanning Tree protocol related information.
Bridge Hello Timer	Time interval between HELLO BPDUs.

Bridge Forward Delay	Time interval used before changing to another state (if this bridge becomes the root).
Bridge Priority	A high-order 2-octet bridge address found in the Bridge Identifier – either the MAC address obtained from the lowest number port, or the address set by the Set Bridge command.

Enter port as the option to modify the Spanning Tree protocol port parameters. Enter the desired value at each prompt and press **RETURN**.

Example: **set protocol port**

```
Port Number [1] ? 1
Port-Path-Cost [1] ?  1
Port Priority [128] ?  1
```

Port Number	Bridge port number; selects the port for which the path cost and port priority is changed.
Path Cost	Cost associated with the port that is used for possible root path cost. The range is 1 to 65535.
Port Priority	Identifies port priority for the specified port. The range is 0 to 255.

route-descriptor-limit ***limit-type RD-limit-value***

Allows the user to associate a maximum Route Descriptor (RD) length for All Route Explorer (ARE) or Spanning Tree Explorer (STE) frames forwarded by the bridge if source routing is enabled.

Example: **set route-descriptor-limit ARE 14**

Limit-type	Entered either as ARE or STE, depending on whether the RD-limit-value is applied to All Route Explorer (ARE) or Spanning Tree Explorer (STE) frames.
RD-limit-value	Specifies the maximum number of RDs contained in the Routing Information Field (RIF) of the frame type specified by the RD limit type. This field takes values from 0 to 14. The default RD limit value for ARE and STE frames is 14.

Tunnel

Access the Tunnel configuration prompt for a specific tunnel. Tunnel configuration commands are entered at this new prompt. See the “Tunnel Configuration Commands” section of this chapter for an explanation of each of these commands.

Syntax: tunnel *tunnel–id*

```
Example: tunnel 2
```

Exit

Return to the previous prompt level.

Syntax: exit

```
Example: exit
```

LNM Configuration Commands

This section summarizes and then explains the LNM (LAN Network Manager) configuration commands. These commands allow you to configure network parameters for the LNM.

Configuration commands are entered at the `LNM config>` prompt. This prompt is accessed by entering the **lnm** command at the `ASRT config>` prompt.

Table 4–2 LNM Configuration Commands

Command	Function
? (Help)	Lists all of the LNM configuration commands, or lists the options associated with specific commands.
Disable	Disables all LNM agents on a specified port or disables specified LNM agents (RPS, CRS, or REM) on a specified port. Also disables the setting of certain LNM parameters from the remote LNM application linked to the bridge. This command applies globally to all instances of LNM within the bridge.

Enable	Enables all LNM agents on a specified port or enables specified LNM agents (RPS, CRS, or REM) on a specified port. Also enables the setting of the reporting link passwords from the remote LNM application linked to the bridge. This command applies globally to all instances of LNM within the bridge.
List	Displays the LNM agents that are enabled for the specified port. Also displays passwords that are configured for the reporting links of a specified bridge port.
Set	Sets the password to be used for the specified reporting link number.
Exit	Exits the LNM configuration process and returns to the ASRT environment.

? (Help)

List the commands that are available from the current prompt level. You can also enter a **?** after a specific command name to list its options.

Syntax: ?

```
Example: list ?
```

Enable

Enable all LNM agents on a specified port or to enable specified LNM agents (RPS, CRS, or REM) on a specified port.

This command also enables the setting of the reporting link passwords from the remote LNM application linked to the bridge.

Syntax: enable *AGENT port#*
lnm *port#*
configuration-remote-change

AGENT port#

Enables the specified LNM agent (either RPS, CRS, or REM) on the specified port. If the interface is not a token ring then the message `Port number XX is not token ring` is displayed. If this is the case, the command has no effect.

If the port is not configured, the message `Port number XX does not exist` is displayed and the command has no effect.

If the specified agent is already enabled for the specified port, the message `Already enabled` is displayed.

```
Example: enable REM

   Port Number [1]? 1
```

lnm *port#*

Enables all LNM agents on the specified bridge port. If the interface is not a token ring, then the message `Port number XX is not token ring` is displayed. If this is the case, the command has no effect.

If the port is not configured, the message `Port number XX does not exist` is displayed and the command has no effect.

If the specified agent is already enabled for the specified port, the message `Already enabled` is displayed.

```
Example: enable lnm

   Port Number [1]? 1
```

configuration-remote-change

Enables the setting of the reporting link passwords from the remote LNM application linked to the bridge. The default setting is to not allow the setting of LNM configuration parameters remotely. This command applies globally to all instances of LNM within the bridge.

```
Example: enable configuration–remote–change
```

Disable

Disable all LNM agents on a specified port or to disable specified LNM agents (RPS, CRS, or REM) on a specified port.

This command also disables the setting of the reporting link passwords from the remote LNM application linked to the bridge.

Syntax: disable AGENT *port#*
lnm *port#*
configuration-remote-change

AGENT *port#*

Disables the specified LNM agent (either RPS, CRS, or REM) on the specified port. If the port is not configured then the message `LNM not configured for Port XX` is displayed. If this is the case, the command has no effect.

```
Example: disable REM 1
```

lnm *port#*

Disables LNM on the specified bridge port. If the port is not configured, then the message `LNM not configured for Port XX` is displayed and the command has no effect.

If none of the LNM agents are enabled for the specified port, then the message `LNM not configured for Port XX` is displayed.

```
Example: disable lnm 1
```

configuration-remote-change

Disables the setting of the reporting link passwords from the remote LNM application linked to the bridge. This command applies globally to all instances of LNM within the bridge.

```
Example: disable configuration–remote–change
```

List

Display the LNM agents that are enabled for the specified port. This command also displays passwords that are configured for the bridge.

Syntax: list password

password

Displays the passwords that are configured for the reporting links of the bridge. Information concerning whether or not the the passwords may be changed by the remote LNM application is also displayed.

Example: **list password**

```
Reporting Link     Password
     0             00000000
     1             00000000
     2             00000000
     3             00000000
LNM Remote Configuration Change: Enabled
```

Set

Set the password to be used for the specified reporting link number. The link number may be 0, 1, 2 or 3. Link 0 is used for the controlling link. Links 1, 2, and 3 are used for observing links.

The password must consist of six to eight characters, and must match the password used by LNM when it establishes a reporting link with the bridge. If the password is not set for a link, it defaults to the string "00000000."

Syntax: set password *link# password*

Example: **set password 1 1 guesswho**

Exit

Return to the previous `ASRT>` prompt level.

Syntax: exit

Example: **?**

NETBIOS Filtering Configuration Commands

This section summarizes and then explains all of the NETBIOS filtering configuration commands. These commands let you configure NETBIOS filtering as an added feature to ASRT bridging.

Configuration commands are entered at the `NETBIOS Filter config>` prompt. This prompt is accessed by entering the **netbios-filtering** command at the `ASRT config>` prompt.

Table 4–3 NETBIOS Filtering Configuration Commands

Command	Function
? (Help)	Lists all of the NETBIOS filtering configuration commands, or lists the options associated with specific commands.
Create	Creates byte filter and host-name filter lists for NETBIOS filtering.
Delete	Deletes byte filter and host-name filter lists for NETBIOS filtering.
Filter-on	Assigns a created filter to a specific port. This filter can then be applied to all NETBIOS packets input *OR* output on the specified port.
List	Displays all information concerning created filters.
Update	Adds information to or deletes information from a host-name or byte filter list.
Exit	Exits the NETBIOS filtering configuration process and returns to the ASRT environment.

? (Help)

Obtain a list of the commands available from that prompt level. You can also enter this command after specific command names to obtain a listing of the command options available for that command.

Syntax: ?

```
Example: ?
```

Create

Build byte filter and host-name filter lists for NETBIOS filtering.

Syntax: create byte-filter-list *filter-list*
name-filter-list *filter-list*

byte-filter-list ***filter-list***

Creates a byte filter list name for NETBIOS filtering. Filter-list is a character string of the user's choice. You can use up to 16 characters to identify the byte filter list being built. Filter-list must be a unique string that was not previously used with the **create byte-filter-list** or **create name-filter-list** command.

Example: **`create byte-filter-list newyork`**

name-filter-list ***filter-list***

Creates a host-name filter list name for NETBIOS filtering. Filter-list is a character string of the user's choice. You can use up to 16 characters to identify the byte filter list being built. Filter-list must be a unique string that was not previously used with the **create byte-filter-list** or **create name-filter-list** command.

Example: **`create name-filter-list newyork`**

Delete

Delete byte filter lists, host-name filter lists, and filters created using the **filter-on input** or **filter-on output** command. The command removes all information associated with byte and host-name filter lists. It also frees the user-defined string as a name for a new filter list.

Syntax: delete byte-filter-list *filter-list*
name-filter-list *filter-list*
filter input *port#*
filter output *port#*

byte-filter-list ***filter-list***

Deletes a byte filter list created for NETBIOS filtering. Filter-list is the user-defined string being used to identify the byte filter list being deleted.

Example: **`delete byte-filter-list newyork`**

name-filter-list ***filter-list***

Deletes a host-name filter list created for NETBIOS filtering. Filter-list is the user-defined string that is used to identify the name-filter-list being deleted.

`Example:` **`delete name-filter-list newyork`**

filter input ***port#***

Deletes a filter that was created using the **filter-on input** command. The command removes all information associated with the filter and fills any gaps in filter numbers that were created.

`Example:` **`delete filter input 2`**

filter output ***port#***

Deletes a filter that was created using the **filter-on output** command. The command removes all information associated with the filter and fills any gaps in filter numbers that were created.

`Example:` **`delete filter output 2`**

Disable

Globally disable NETBIOS filtering on the bridging router.

Syntax: disable netbios-filtering

netbios-filtering

Disables NETBIOS filtering on the bridging router.

`Example:` **`disable netbios-filtering`**

Enable

Globally enable NETBIOS filtering on the bridging router.

Syntax: enable netbios-filtering

netbios-filtering

Enables NETBIOS filtering on the bridging router.

`Example:` **`enable netbios-filtering`**

Filter-on

Assign one or more previously configured filter lists to the input or output of a specific port.

Syntax: filter-on input *port# filter-list <operator filter-list . . . >*
output *port# filter-list <operator filter-list . . . >*

input *port# filter-list <operator filter-list . . . >*

This command assigns one or more filter lists to a specific port. The resulting filter is then applied to all NETBIOS packets input on the specified port.

Port# is a configured bridging port number on the router. Filter-list is a string previously entered through the **create** command. An optional operator is entered as either AND or OR. The operator is entered in all capital letters. If an operator is present, it must be followed by a filter-list name. The port number (obtained from the **list** command) is used to identify this filter.

Note: Multiple operators can be used. This creates a complex filter. If one or more operators are present, they must all be entered at the same time on the same command line.

The filter created by this command is applied to all NETBIOS packets input on the specified port number. Each filter-list on the command line is evaluated left to right along with any operators that are present. An Inclusive evaluation of a filter list is equivalent to a TRUE condition and an Exclusive evaluation is equivalent to a FALSE condition. If the result of the evaluation of the filter-list(s) is TRUE, the packet is bridged. Otherwise, the packet is filtered (dropped).

If the packet is not one of the types supported by NETBIOS filtering then all host-name filter lists for this filter are designated Inclusive (TRUE). If an input filter already exists for specified port number, an error message is displayed.

```
Example: filter-on input 2 newyork AND boston
```

output *port# filter-list <operator filter-list . . . >*

This command assigns a created filter to a specific port. This filter is then applied to all NETBIOS packets output on that port.

Port# is a configured bridging port number on the router. Filter-list is a string previously entered through the **create** command. An optional operator is entered as either AND or OR. The operator is entered in all capital letters. If an operator is present, it must be followed by a filter-list name. The port number (obtained from the **list** command) is used to identify this filter.

Note: Multiple operators can be used. This creates a complex filter. If one or more operators are present, they must all be entered at the same time on the same command line.

The filter created by this command is applied to all NETBIOS packets output on the specified port number. Each filter-list on the command line is evaluated left to right along with any operators that are present. An Inclusive evaluation of a filter list is equivalent to a TRUE condition and an Exclusive evaluation is equivalent to a FALSE condition. If the result of the evaluation of the filter-list(s) is TRUE, the packet is bridged. Otherwise, the packet is filtered (dropped).

If the packet is not one of the types supported by NETBIOS filtering then all host-name filter lists for this filter are designated Inclusive (TRUE). If an output filter already exists for specified port number, an error message is displayed.

`Example:` **`filter-on output 2 newyork OR boston`**

List

Display all information concerning created filters.

Syntax: list

`Example:` **`list`**

```
NETBIOS Filtering: Disabled
NETBIOS Filter Lists
--------------------
    Handle           Type
    nlist            Name
    newyork          Byte
NETBIOS Filters
---------------
    Port #      Direction      Filter List Handle(s)
       3          Output       nlist
```

`NETBIOS Filtering:`	Displays whether NETBIOS filtering is enabled or disabled.
`NETBIOS Filter Lists`	Displays the user-defined name (handle) of the configured filter lists. For type, Name indicates a host-name filter list and Byte indicates a byte filter list.
`NETBIOS Filters`	Displays the assigned port number and direction (input or output) of each filter. Filter List Handle(s) displays the name(s) of the filter list(s) making up the filter.

Update

Add or delete information from host-name or byte filter lists. The filter-list is a string previously entered through the **create byte- (or name-) filter-list** prompt. This command brings you to the `NETBIOS Byte (or Name)` *`filter-list`* `Config>` command level which lets you perform update tasks to the specified filter list. At this prompt level you can add, delete, list, or move filter-items from byte and host-name filter lists. You can also set the default value of each filter list to Inclusive or Exclusive.

Using the **add** subcommand creates a filter item within the filter list. The first filter item created is assigned number 1, the next one is assigned number 2, and so forth. After a successful **add** subcommand is entered by the user, the router displays the number of the filter item just added.

Note: Adding more filter items to filter lists adds to processing time (due to the time it takes to evaluate each filter item in the list) and affects performance in heavy NETBIOS traffic.

The order in which filter items are specified for a given filter list is important as this determines the way in which the filter items are applied to a packet The first match that occurs stops the application of filter items and the filter list is evaluated as either Inclusive or Exclusive (depending on the Inclusive or Exclusive designation of the matched filter item). If none of the filter items of a filter list produce a match, then the default condition (Inclusive or Exclusive) of the filter list is returned.

The **delete** subcommand specifies the number of a filter item to be deleted from the filter list. When a **delete** subcommand is given, any hole created in the list is filled in. If filter items 1, 2, 3, and 4 exist and filter item 3 is deleted, then filter item 4 is renumbered to 3.

The **default** subcommand lets you change the default setting of the filter list to either Inclusive or Exclusive. If a filter list evaluates as Inclusive, then the packet being considered is bridged. Otherwise, the packet is filtered.

The **move** subcommand is available to renumber filter items within a filter list. The first argument to the **move** subcommand is the number of the filter list to be moved. The second argument to the **move** subcommand is the number of the filter list after which the first filter list is moved.

Syntax:	update	byte-filter-list . . .
		name-filter-list . . .

byte-filter-list ***filter-list***

Updates information belonging to a byte filter-list. The filter-list parameter is a string previously entered through the **create byte-filter-list** prompt. This command brings you to the next `NETBIOS BYTE filter-list Config>` command level (see example). At this level you can perform update tasks to the specified filter-list.

```
Example: update byte-filter-list newyork

   NETBIOS Byte newyork Config>
```

At this new prompt level you can execute several commands. Each available command is listed in the **"Update Byte-Filter** Command Options" section which follows. The correct syntax is listed followed by a description of that command and its required parameters.

name-filter-list ***filter-list***

Updates information belonging to a name-filter list. This command is identical to the previous command, with the exception that the command specifies a name-filter list rather than a byte-filter list. The filter-list parameter is a string previously entered through the **create name-filter-list** prompt. This command brings you to the next `NETBIOS Name filter-list Config>` command level (see example). At this level you can perform update tasks to the specified filter-list.

```
Example: update name-filter-list accounting
```

```
NETBIOS Name accounting Config>
```

At this new prompt level you can execute several commands. Each available command is listed in the “**Update Name-Filter** Command Options” section which follows. The correct syntax is listed followed by a description of that command and its required parameters.

Update Byte-Filter-List Command Options

The following section lists the command options available for the **update byte-filter-list** command:

add inclusive *byte-offset hex-pattern <hex mask>*

Adds a filter item to the byte filter list. If the byte filter item that is added produces a match with a NETBIOS packet, the filter list it belongs to evaluates to Inclusive (True).

- Byte-offset specifies the number of bytes (in decimal) to offset into the packet being filtered. This starts at the NETBIOS header of the packet.
- Hex-pattern is a hexadecimal number used to compare with the bytes starting at the byte-offset offset of the NETBIOS header. Syntax rules for hex-pattern include no 0x in front, a maximum of 32 numbers, and an even number of hex numbers.
- Hex-mask, if present, must be the same length as hex-pattern and is logically ANDed with the bytes in the packet starting at byte-offset before the result is compared for equality with hex_pattern. If the hex-mask argument is omitted, it is considered to be all binary 1’s.

If the offset and pattern of a byte filter item represent bytes that do not exist in a NETBIOS packet (if the packet is shorter than was intended when setting up a byte-filter list), then the filter item is not applied to the packet and the packet is not filtered. If a series of byte filter items is used to set up a single NETBIOS filter list, then a packet is not tested for filtering if any of the byte filter items within the NETBIOS filter list represent bytes that do not exist in the NETBIOS packet.

add exclusive *byte-offset hex-pattern <hex mask>*

Adds a filter item to the byte filter list. This command is identical to the preceding command, with the exception that if the result of the comparison between the filter item and a NETBIOS packet results in a match, then the filter list evaluates to Exclusive (False). Datagram Broadcast Packets can be specified to be discarded by using this command with a byte offset of 4 and a byte pattern of 09.

- Byte-offset specifies the number of bytes (in decimal) to offset into the packet being filtered. This starts at the NETBIOS header of the packet.
- Hex-pattern is a hexadecimal number used to compare with the bytes starting at the byte-offset offset of the NETBIOS header. Syntax rules for hex-pattern include no 0x in front, a maximum of 32 numbers, and an even number of hex numbers.
- Hex-mask, if present, must be the same length as hex-pattern and is logically ANDed with the bytes in the packet starting at byte-offset before the result is compared for equality with hex_pattern. If the hex-mask argument is omitted, it is considered to be all binary 1's.

If the offset and pattern of a byte filter item represent bytes that do not exist in a NETBIOS packet (if the packet is shorter than was intended when setting up a byte-filter list), then the filter item is not applied to the packet and the packet is not filtered. If a series of byte filter items is used to set up a single NETBIOS filter list, then a packet is not tested for filtering if any of the byte filter items within the NETBIOS filter list represent bytes that do not exist in the NETBIOS packet.

default include

Changes the default setting of the filter list to inclusive. This command indicates that if no filter items of the filter list match the contents of the packet being considered for filtering, the filter list is evaluated as Inclusive. This is the default setting.

default exclude

Changes the default setting of the filter list to exclusive. This command indicates that, if no filter items of the filter list match the contents of the packet being considered for filtering, the filter list is evaluated as Exclusive.

delete filter-item

Deletes a filter item from the filter list. Filter-item is a decimal number representing a filter item that was previously created by the **add** command.

list

Displays information related to filter items in the specified filter list.

```
BYTE Filter List Name:     Enginering
BYTE Filter List Default:  Exclusive



Filter Item #  Inc/Ex    Byte Offset    Pattern         Mask

   1          Inclusive        14         0x123456       0xFFFF00
   2          Exclusive         0         0x9876         0xFFFF
   3          Exclusive        28         0x1000000      0xFF00FF00
```

move filter-item1 filter-item2

Reorders filter items within the filter list. The filter item whose number is specified by filter-item1 is moved and renumbered to be just after filter item2.

exit

Exits to the previous command prompt level.

Update Name-Filter-List Command Options

The following section lists the command options available for the **update name-filter-list** command:

add inclusive ASCII host-name <LAST-hex-number>

Adds a filter item to the host-name filter list. With this command, the host-name fields of the NETBIOS packets are compared to the host-name given in this command.   The following list shows how these comparisons are made:

- ADD_GROUP_NAME_QUERY: Source NETBIOS name field is examined
- ADD_NAME_QUERY: Source NETBIOS name field is examined
- DATAGRAM: Destination NETBIOS name field is examined
- NAME_QUERY: Destination NETBIOS name field is examined

If there is a match (taking into account wildcard designations in this command) then the filter list evaluates to Inclusive. If not, the next filter item of the filter list (if any) of the filter is applied to the packet. If the packet is not one of the four types supported by NETBIOS Name filtering then the packet is bridged.

- Host-name is an ASCII string up to 16 characters long. A "?" can be used in host-name to indicate a single character wildcard. A "*" can be used as the final character of host-name to indicate a wildcard for the remainder of the host-name. If host-name contains fewer than 15 characters, it is padded to the 15th character with ASCII spaces. Host-name can contain any character but the following: . / \ [ ] : | < > + = ; , <space>

- LAST-hex-number can be used if host-name contains fewer than 16 characters. It is a hexadecimal number (with no x in front of it) that indicates the value used for the last character. If the LAST argument is not specified on a host-name less than 16 characters, then a "?" wildcard is supplied for the 16th character.

add inclusive HEX hexstring

Adds a filter item to the host-name filter list. This command is functionally the same as **add inclusive ASCII** command. The representation of host-name is different. This command supplies the host-name as a series of hexadecimal numbers (with no 0x in front).

- Hexstring must consist of an even number of hexadecimal numbers. If the user does not supply a full 32 hexadecimal numbers, ASCII blanks are padded to the 29th and 30th numbers and a wildcard is supplied as the 31st and 32nd (16th byte) numbers. A wildcard for a single byte can be specified by "??".

add exclusive ASCII host-name <LAST-hex-number>

Adds a filter item to the host-name filter list. This command is identical to the **add inclusive ASCII** command, with the exception that packets which are matched against this filter item produce an Exclusive result for the filter list.

- Host-name is an ASCII string up to 16 characters long. A "?" can be used in host-name to indicate a single character wildcard. A "*" can be used as the final character of host-name to indicate a wildcard for the remainder of the host-name. If host-name contains fewer than 15 characters, it is padded to the 15th character with ASCII spaces. Host-name can contain any character but the following: . / \ [ ] : |< > + = ; , <space>

- LAST-hex-number can be used if host-name contains fewer than 16 characters. It is a hexadecimal number (with no x in front of it) which indicates the value to be used for the last character. If the LAST argument is not specified on a host-name less than 16 characters, then a "?" wildcard is supplied for the 16th character.

add exclusive HEX hexstring

Adds a filter item to the name filter list. This command is functionally the same as **add inclusive hex** command with the following exception: Packets that are matched against this filter item produce an Exclusive result for the filter list.

Hexstring must consist of an even number of hexadecimal numbers. If the user does not supply a full 32 hexadecimal numbers, ASCII blanks are padded to the 29th and 30th numbers and a wildcard is supplied as the 31st and 32nd (16th byte) numbers. A wildcard for a single byte can be specified by "??".

default include

Changes the default setting of the filter list to inclusive. This command indicates that if no filter items of the filter list match the contents of the packet being considered for filtering, the filter list evaluates to Inclusive. This is the default setting.

default exclude

Changes the default setting of the filter list to "exclusive." This command indicates that if no filter items of the filter list match the contents of the packet being considered for filtering, the filter list is evaluated as Exclusive.

delete filter-item

Deletes a filter item from the filter list. Filter-item is a decimal number representing a filter item that was previously created by the **add** command.

list

Displays information related to filter items in the specified filterlist.

```
    NAME Filter List Name: nlist
    NAME Filter List Default: Exclusive

     Filter Item #   Type    Inc/Ex        Hostname       Last Char

           1         ASCII   Inclusive     EROS
           2         ASCII   Inclusive     ATHENA
           3         ASCII   Exclusive     FOOBAR
```

move filter-item1 filter-item2

Reorders filter items within the filter list. The filter item whose number is specified by *filter-item1* is moved and renumbered to be just after *filter-item2*.

exit

Exits to the previous command prompt level.

Exit

Return to the previous `ASRT>` prompt level.

Syntax: exit

`Example:` **`exit`**

Tunnel Configuration Commands

This section summarizes and then explains the Tunnel configuration commands. The Tunnel configuration commands allow you to specify network parameters for specified tunnels that transmit bridging frames over IP.

Configuration commands for specifically defined tunnels are entered at the `Tunnel(#)config>` prompt. This prompt is accessed by entering the **tunnel** command at the `ASRT config>` prompt.

Table 4–4 Tunnel Configuration Commands

Command	Function
? (Help)	Lists all of the Tunnel configuration commands, or lists the options associated with specific commands.
Add	Adds the IP address of destination bridges participating in an IP unicast or multicast addressing configuration for bridging over IP.
Delete	Deletes the IP address of a destination bridge participating in an IP unicast or multicast addressing configuration for bridging over IP.
List	Displays the IP addresses of end stations participating in an IP unicast or multicast addressing configuration for bridging over IP. Also displays the size (in number of bytes) of bridging packets being routed through an IP tunnel and whether or not multicast addressing is enabled or disabled.
Exit	Exits the tunnel configuration process and returns to the ASRT environment.

Tunneling and Multicast Packets

For tunnel configurations where multicast packets are involved, the source address of the multicast packets must lie on a network segment that is capable of the Internet Group Management Protocol (IGMP).

IGMP is not defined on X.25 or Frame relay configurations so when running multicast applications on the router (the MOSPF tunnel), care must be taken that one of the following conditions occur:

- The source is one of the LAN segment addresses.
- The source is the internal IP address.

The first condition can be ensured by using the IP **set router–id** configuration command. The second condition can be ensured by using the IP **set internal-ip-address** configuration command.

In all cases, the second option is preferred and the first is only used if some of the routers in the network do not like host addresses (occurs in mixed vendor networks).

? (Help)

List the commands that are available from the current prompt level. You can also enter a **?** after a specific command name to list its options.

Syntax: ?

`Example:` **`?`**

Add

Add the IP address of end stations participating in a unicast or multicast IP addressing configuration.

For IP unicast addressing, the tunneling configuration requires you supply IP addresses of destination bridges. This record is used by the router software to convert the segment number in the RIF (Routing Information Field) in a source routed frame to the corresponding IP address of the destination bridge. For transparent bridging frames, it identifies the other endpoint of the tunnel.

For IP multicast addressing, the tunneling configuration requires only the IP multicast address reserved for tunneling. Encapsulation uses three groups of IP multicast addresses. The first group is for sending All Route Explorer (ARE) frames, the second group for sending Spanning Tree Explorer (STE) frames, and the third group for Specifically Routed Frames (SRF).

Note: The bridging router software transparently differentiates between unicast and multicast addresses.

Syntax: add address *IP-address*

`Example:` **`add address 128.185.144.37`**

Delete

Delete the IP address of bridges participating in a unicast or multicast IP addressing configuration.

Syntax: delete address *IP-address*

`Example:` **`delete address 128.185.144.37`**

List

Display the IP addresses of bridges participating in an IP unicast or multicast addressing configuration for tunneling over IP. This command can also be used to display the current size of IP packets being sent through the tunnels and displays whether or not IP is enabled or disabled.

Syntax: list address
all
packet-size

address

Lists the IP addresses of bridges participating in an IP unicast or multicast addressing configuration for tunneling over IP.

Example: **list address**

```
IP Tunnel Addresses
128.185.179.51      128.185.170.51      128.185.142.39 128.185.143.39
    224.0.0.5
```

all

Lists all unicast IP addresses, configured multicast addresses, and the tunnel packet size.

Example: **list all**

```
IP Tunnel Addresses
 128.185.179.51      128.185.170.51      128.185.142.39
 128.185.143.39       224.0.0.5

Frame size for the tunnel 2120
```

packet-size

Lists the frame size of the tunnel.

Example: **list packet-size**

```
Frame size for the tunnel 2120
```

Exit

Return to the previous `ASRT>` prompt level.

Syntax: exit

```
Example: exit
```

5

Basic Bridging Configurations

This chapter describes how to create basic configurations for the Adaptive Source Routing Transparent (ASRT) Bridge using the ASRT configuration commands.

If you need more information about the ASRT bridge configuration commands, refer to the previous chapter. For more information about the ASRT bridge, refer to the *Routing Protocols Reference Guide*.

Accessing the ASRT Configuration Environment

For information on how to access the ASRT configuration environment, see Chapter 1 in the *Network Interface Operations Guide.*

Note: After you access the bridging configuration process, you may begin entering configuration commands. Whenever you make a change to a user-configurable interface parameter, you must restart the router for this change to take effect.

Basic Bridging Configuration Procedures

The philosophy behind the design of the ASRT bridge is to allow the basic configuration of bridging option by using as few commands as possible. Using the **enable bridge** command begins this process by letting all properly configured devices participate in transparent bridging. In addition, all default values for the spanning tree algorithm are enabled.

Bridging functionality beyond transparent bridging is then enabled on a “per interface” basis. Obviously, when source routing is enabled, the usual user input such as segment number, or bridge number is still required and must be entered beyond the basic commands that are explained.

Bridging Interfaces

The interfaces over which bridging is supported include combinations of one or more of the following:

- Ethernet
- Token ring
- FDDI
- Serial line

The Ethernet and FDDI interfaces typically support transparent bridging while token ring interfaces can support source routing and transparent bridging.

The serial line interface provides point-to-point connectivity (using a proprietary protocol) for transparent and source routing traffic. It is important to note that a bridge configuration over a serial line must be consistent at both end points. This means that both endpoints are configured as follows:

- Transparent to transparent
- Source routing to source routing
- Source routing/transparent to source routing/transparent

It is best if the serial line is configured for both bridging methods if mixed bridging is desired. Another suggested guideline is to make sure that bridging routers are consistent in their bridging method or in their routing of particular protocols.

The information immediately following outlines the initial steps required to enable the bridging options offered by the ASRT bridge. Details on making further configuration changes is covered in the command sections of this chapter. After completing these tasks, the router must be restarted for the new configuration to take effect.

Note: Transparent bridging over token ring, frame relay, X.25, and PPP is not supported for this release. This restriction can be overcome by configuring the IP tunnel feature. The commands for this feature are explained later in this chapter.

Enabling the Transparent Bridge

Use the following commands to enable transparent bridging:

- Enable bridge to enable transparent bridging on all Local Area Network (LAN) interfaces. Wide Area Network (WAN) interfaces (such as serial lines) can be included by using the **add port** command.
- Disable transparent *port#* to exclude specified token ring interfaces from participating in transparent bridging. Repeat the command for all interfaces you want excluded from the transparent bridging configuration.

Enabling the Source Routing Bridge

Use the following commands to enable source routing bridging:

- Enable bridge to enable bridging on all Local Area Network interfaces. WAN interfaces (serial lines) can be included by using the **add port** command.
- Disable transparent *port#* to disable transparent bridging on all ports.
- Enable source-routing bridge *port# segment# [bridge#]* to enable source-routing for given ports. When source-routing is enabled on more than two ports, an additional segment number is required to assign an internal virtual segment needed for 1:N SRB configurations.

  If source routing is the only feature desired, transparent bridging on the interfaces is disabled.

Note: Do not include interfaces that traditionally do not support source routing. For example, if transparent bridging is disabled and source routing is enabled on an Ethernet port, the bridging facility is disabled for this port.

Enabling the SR-TB Bridge

Use the following commands to enable SR-TB bridging:

- Enable bridge to enable bridging on all Local Area Network interfaces. WAN interfaces (serial lines) can be included by using the **add port** command.

- Disable transparent *port#* to disable transparent bridging on all underlying source routing interfaces.

- Enable source-routing bridge *port# segment# [bridge#]* to enable source-routing for given ports. When source-routing is enabled on more than two ports, an additional segment number is required to assign an internal virtual segment needed for 1:N SRB configurations.

- Enable sr-tb-conversion *segment#* to enable conversion of source-routed frames to transparent frames and vice versa. You are also required to assign a segment number to represent the *entire* transparent (Ethernet /FDDI) bridging domain.

After completing any of the procedures just described, it is advised that you use the **list bridge** command to display the current bridge configuration. This lets you verify and check your configuration.

For more information on all of the commands just mentioned, refer to the “Configuring Bridging” chapter in this guide.

ASRT Configuration Matrix

With an ASRT Bridge, the collection of configuration parameters for the bridge and all connected interfaces produce a *bridge personality* for that bridge. The following matrix provides a guide to the configuration settings needed for each interface type to produce the desired bridge personality to handle your network.

Table 5–1 Bridge Configuration Settings Matrix

BRIDGE PERSONALITY	SR <--> TB CONVERSION ENABLED?	INTERFACE TYPE & BRIDGING METHOD SETTING				
		Token Ring 1	Token Ring 2	Ethernet	FDDI	Serial Line or Tunnel
STB	NO	TB	TB	TB	TB	TB
SRB	NO	SR	SR	—	—	SR
STB & SRB	NO	SR	SR	TB	TB	TB or SR
SR <—> TB	YES	SR	SR	TB	TB	TB
SR <—> TB	YES	SR	TB	TB	TB	SR
SRT	NO	SR & TB	SR & TB	TB	TB	SR & TB
ASRT	YES	SR & TB	SR	TB	TB	SR & TB
ASRT	YES	SR	SR & TB	TB	TB	SR & TB
ASRT	YES	SR or TB	SR or TB	TB	TB	SR & TB

BRIDGE PERSONALITY KEY

STB	Transparent (Spanning Tree) Bridge
SRB	Source Routing Bridge
SR <—> TB	Source Routing-Transparent Conversion Bridge
SRT	Source Routing Transparent Bridge
ASRT	Source Routing Transparent Bridge

BRIDGING METHOD KEY

SR	Source Routing
TB	Transparent Bridging

Basic Configuration Procedures for NETBIOS Filtering

The following sections provide two examples of how to setup NETBIOS filtering. The first example explains how to create a host-name filter. The second example demonstrates how to configure a byte filter. For more information on the commands used in these examples, refer to the "Configuring Bridging" chapter in this guide. All commands are entered at the `NETBIOS filtering config>` prompt.

Example 1: Creating a Host Name Filter

Use the following procedure as a guideline for creating a host-name filter.

Create An Empty Name Filter List

1. Create an empty filter list by using the **create name-filter-list** command.

```
NETBIOS Filter config>create name-filter-list
Handle for Name Filter List []? boston
```

Add the Filter Items to the Name Filter List

2. Use the **update** command to get to the prompt for that specific filter list prompt. From this prompt you are able to add filter items to the filter list.

```
NETBIOS Filter config>update boston
Name Filter List Configuration
NETBIOS Name boston config>
```

3. Begin adding filter items to the filter list by using the **add** command. The way filter items are configured determines which NETBIOS packets are bridged or dropped. Host name filter items are configured with the following parameters entered in this order:

   – Inclusive (bridged) or Exclusive (dropped).

   – ASCII or HEX – How the host-name is represented.

   – host-name – The actual host-name represented in either an ASCII or hex string (see the command section that follows for actual syntax). This entry is case sensitive.

– <LAST-hex-number> – An optional parameter that can be used with ASCII strings containing fewer than 16 characters.

The following example adds a filter item to the Host Name Filter list **boston**, which allows packets containing the host-name **westport** (an ASCII string) to be bridged (configured as **inclusive**). No <Last-hex-number> parameter was configured for this entry.

```
NETBIOS Name boston config>add inclusive ascii


Hostname []? westport
Special 16th character in ASCII hex (<CR> for no special char) []? <CR>
```

Note: All parameters can be entered as one string on the command line if you do not want to be prompted. Be sure to use a space between each parameter.

Verify the Filter Item Entry

4. Use the **list** command to verify your entry:

```
NETBIOS Name boston config>list


NAME Filter List Name: boston
NAME Filter List Default: Inclusive


 Item #   Type    Inc/Ex    Hostname         Last Char


   1      ASCII    Inc      westport
```

Add Additional Filter Items to the Filter List

5. Repeat step 1 through step 4 to add additional filter items to the filter list. After you add filter items to the filter list, use the **exit** command to return to the previous `NETBIOS Filter config>` prompt.

```
NETBIOS Name boston config>exit
NETBIOS Filter config>
```

Notes: The order in which filter items are specified for a given filter list is important as this determines the way in which the filter items are applied to a packet. The first match that occurs stops the application of filter items and the filter list is evaluated as either Inclusive or Exclusive (depending on the Inclusive or Exclusive designation of the matched filter item). If none of the filter items of a filter list produce a match, then the default condition (Inclusive or Exclusive) of the filter list is returned.

Specifying the most common filter items at the front of the list also ensures match efficiency by making sure that the evaluation process makes a match at the beginning of the list rather than having to check the whole list before a match is made.

Add the Filter to Your Configuration

6. The filter list containing the filter item(s) is now added as a filter to your bridging router configuration. Use the **filter-on** command to do this. Host name filters are configured with the following parameters (entered in this order):

   – INPUT (to filter all NETBIOS packets received on that port) or OUTPUT (to filter all NETBIOS packets transmitted on that port)

   – Port# – the desired configured bridging port number on the router.

   – Filter-list is the name of the filter list (containing filter items) that you want included in this filter.

   – An optional operator entered as either “AND” or “OR. ” The operator is entered in all capital letters. If an operator is present, it must be followed by a filter-list name. Filters with more than one filter list are called complex filters. These are explained in more detail in the command section that follows.

   The following example adds a host-name filter that affects packets input on port #3. It is comprised of the host-name filter list **boston**. All packets input on port #3 are evaluated according to the rules provide by the filter items contained in the filter list **boston**. This means that all packets input on port #3 containing the host-name **westport** is bridged**.**

```
NETBIOS Filter config>filter-on input
Port Number [1]? 3
Filter List []? boston
```

Note: All parameters can be entered as one string on the command line if you do not want to be prompted. Be sure to use a space between each parameter.

Verify the Newly Created Filter

7. Use the **list** command to verify your entry:

```
NETBIOS Filter config>list

NETBIOS Filtering: Disabled

NETBIOS Filter Lists
--------------------

    Handle          Type
    nlist           Name
    newyork         Name
    HELLO           Byte
    boston          Name

NETBIOS Filters
---------------

    Port #     Direction      Filter List Handle(s)
       3         Output       nlist
       1         Input        newyork OR HELLO
       3         Input        boston
```

Globally Enable NETBIOS Filtering

8. Use the **enable** command to globally enable on the bridging router.

```
NETBIOS Filter config>enable netbios–filtering
```

Restart the Router to Activate All NETBIOS Filtering Configuration Changes

9. Use the **exit** command followed by CTRL/*p* to return to the MOS asterisk prompt. From this prompt, enter the **restart** command to activate all software changes made during the NETBIOS Filtering configuration process.

```
NETBIOS Filter config>exit
ASRT config>exit
Config> <ctrl-p>
* restart
```

Example 2: Creating a Byte Filter

Use the following procedure as a guideline to creating a byte filter. Entering of all commands starts at the `NETBIOS filtering config>` prompt.

Create An Empty Byte Filter List

1. Create an empty filter list by using the **create byte-filter-list** command.

```
NETBIOS Filter config>create byte-filter-list
Handle for Byte Filter List []? westport
```

Add the Filter Items to the Byte Filter List

2. Use the **update** command to get to the prompt for that specific filter list prompt. From this prompt you are able to add filter items to the filter list.

```
NETBIOS Filter config>update westport
Byte Filter List Configuration
NETBIOS Byte westport config>
```

3. Begin adding filter items to the filter list by using the **add** command. The way filter items are configured determines which NETBIOS packets are bridged or dropped. Byte filter items are configured with the following parameters (entered in this order):

   – Inclusive (bridged) or Exclusive (dropped).

   – Byte-offset – The number of bytes (in decimal) to offset into the packet being filtered. This starts at the NETBIOS header of the packet.

– Hex-pattern – A hexadecimal number used to compare with the bytes starting at the byte-offset offset of the NETBIOS header. See the NETBIOS command section for syntax rules.

– Hex-mask – (If present) must be the same length as hex-pattern and is logically ANDed with the bytes in the packet starting at byte-offset before the result is compared for equality with hex_pattern. If the hex-mask argument is omitted, it is considered to be all binary 1's.

The following example adds a filter item to the Byte filter list **westport** which allows packets with a hex pattern 0x12345678 at byte offset of 0 to be bridged (configured as **inclusive).** No hex mask is present.

```
NETBIOS Byte westport config>add inclusive
Byte Offset [0]?
Hex Pattern []? 12345678
Hex Mask (<CR> for no mask) []?
```

Note: All parameters can be entered as one string on the command line if you do not want to be prompted. Be sure to use a space between each parameter.

Verify the Filter Item Entry

4. Use the **list** command to verify your entry:

```
NETBIOS Byte westport config>list

BYTE Filter List Name: westport
BYTE Filter List Default: Inclusive

 Item #   Inc/Ex   Offset  Pattern               Mask

   1       Inc        0    0x12345678            0xFFFFFFFF
```

Add Additional Filter Items to the Filter List

5. Repeat the first four steps just described to add additional filter items to the filter list. After you add filter items to the filter list, use the **exit** command to return to the previous `NETBIOS Filter config>` prompt.

```
NETBIOS Byte westport config>exit
NETBIOS Filter config>
```

Notes: The order in which filter items are specified for a given filter list is important, as this determines the way in which the filter items are applied to a packet. The first match that occurs stops the application of filter items and the filter list is evaluated as either Inclusive or Exclusive (depending on the Inclusive or Exclusive designation of the matched filter item). If none of the filter items of a filter list produce a match, then the default condition (Inclusive or Exclusive) of the filter list is returned.

Specifying the most common filter items at the front of the list also ensure match efficiency by making sure that the evaluation process makes a match at the beginning of the list rather than having to check the whole list before a match is made.

Add the Filter to Your Configuration

6. The filter list containing the filter item(s) is now be added as a filter to your bridging router configuration. Use the **filter-on** command to do this. Host name filters are configured with the following parameters (entered in this order):

   – INPUT (to filter all packets received on that port) or OUTPUT (to filter all packets transmitted on that port)

   – Port# – The desired configured bridging port number on the router.

   – Filter-list is the name of the filter list (containing filter items) that you want to be included in this filter

   – An optional operator entered as either "AND" or "OR. " The operator is entered in all capital letters. If an operator is present, it must be followed by a filter-list name. Filters with more than one filter list are called complex filters. These are explained in more detail in the command section that follows.

   The following example adds a host-name filter that affects packets *output* on port #3. It is comprised of the byte filter list **westport**. All packets output on port #3 are evaluated according to the rules provided by the filter items contained in the filter list **boston**.

```
NETBIOS Filter config>filter-on output
Port Number [1]? 3
Filter List []? westport
```

Note: All parameters can be entered as one string on the command line if you do not want to be prompted. Be sure to use a space between each parameter.

Verify the Newly Created Filter

7. Use the **list** command to verify your entry:

```
NETBIOS Filter config>list

NETBIOS Filtering: Disabled

NETBIOS Filter Lists
--------------------

    Handle           Type

    nlist            Name
    newyork          Name
    HELLO            Byte
    westport         Byte

NETBIOS Filters
---------------

    Port #      Direction      Filter List Handle(s)

       3          Output       nlist
       1          Input        newyork OR HELLO
       3          Output       westport
```

Globally Enable NETBIOS Filtering

8. Use the **enable** command to globally enable on the bridging router.

```
NETBIOS Filter config>enable netbios-filtering
```

Restart the Router to Activate All NETBIOS Filtering Configuration Changes

9. Use the **exit** command followed by CTRL/*p* to return to the MOS asterisk prompt. From this prompt, enter the **restart** command to activate all software changes made during the NETBIOS Filtering configuration process.

```
NETBIOS Filter config>exit
ASRT config>exit
Config> <ctrl-p>
* restart
```

Basic LNM Configuration

This section summarizes the procedure necessary for basic configuration of the LNM feature on your Bridging Router.

Obtain the MAC Address Required For Network Manager Software

1. Use the **list lnm status** command at the ASRT console prompt (`ASRT>`) to obtain the MAC addresses required by the Network Manager software running on the Network Manager Station. For example:

```
ASRT>list lnm status
Port Number [1]? 1
Port 1
LNM Agents Enabled: RPS CRS REM
Reporting Link          State          LNM Station Address
      0                 ACTIVE         10:00:5A:F1:02:37
      1                 AVAILABLE
      2                 AVAILABLE
      3                 AVAILABLE
MAC Addresses to use when configuring LNM Manager:
      0:00:C9:08:35:47
      40:00:D9:08:35:47
```

The MAC addresses displayed on the bottom two lines (shown in bold the example) of the output are used by the Network Manager to configure it to the LNM agents present in the router.

Note: These addresses must be entered exactly as they appear in the output or LNM will not configure correctly.

Enable The LNM Agents on the Bridging Router

2. Use the **enable lnm** command at the `LNM config>` prompt to enable the LNM agents on the desired port of your bridging router. For example:

```
LNM config>enable LNM
Port Number [1]? 1
```

The default setting has all LNM agents enabled.

Check the Configuration By Displaying Enabled LNM Agents

3. Use the **list port** command at the `LNM config>` prompt to display which LNM agents are enabled on your configured port. For example:

```
LNM config>list port
Port Number [1]? 1
LNM Agents Enabled: RPS CRS REM
```

6

Monitoring Bridging

This chapter describes how to monitor the ASRT (Adaptive Source Routing Transparent) Bridge and how to use the ASRT console commands. Console commands for the bridging router’s tunnel and NETBIOS filtering features are also included as part of the general ASRT console command set.

If you need more information about the ASRT bridge and its bridging options, refer to the *Routing Protocols Reference Guide*.

Accessing the ASRT Console Environment

For information on how to access the ASRT console environment, see the chapter entitled “Getting Started” in the *System Software Guide.*

ASRT Console Commands

This section explains the ASRT console commands. These commands allow you to view and modify parameters from the active console. Information you modify with the console commands is not permanent. The configuration is reset to that stored in non-volatile memory when you restart the bridging router.

You can use these commands to temporarily modify the configuration without losing configuration information in the bridge memory. The `ASRT>` prompt displays for all ASRT console commands.

Note: For commands requiring you to enter MAC Addresses, the addresses can be entered in the following formats:

IEEE 802 canonical bit order	00–00–00–12–34–56
IEEE 802 canonical bit order (shorthand format)	000000123456
IBM token ring native bit order (non-canonical)	00:00:00:12:34:56

Table 6–1 ASRT Console Commands Summary

Command	Function
? (Help)	Lists all the ASRT console commands or lists the options associated with specific commands.
Add	Adds permanent (static) address entries to the bridging router's permanent database.
Cache	Displays cache entries for a specified port.
Delete	Deletes MAC addresses entries from the bridging router database.
Flip	Flips MAC address from canonical to 802.5 (non-canonical or IBM) bit order.
List	Displays information about the complete bridge configuration or about selected configuration options.
Netbios-filter	Brings the user to the NETBIOS filtering prompt where NETBIOS filter console commands may be entered.
Exit	Exits the ASRT console process and returns to the GWCON environment.

? (Help)

List the commands that are available from the current prompt level. You can also enter a **?** after a specific command name to list its options.

Syntax: ?

```
Example: list ?
```

Add

Add static address entries to the bridging router’s permanent database.

Syntax: add static-entry . . .
destination-address-filter

static-entry ***mac_address input_port [output_ports]***

Adds static address entries to the bridging router’s permanent database. Enter the command followed by the MAC address of the static entry and the input port number (an optional output port number may also be entered).

To create a static entry with multiple port maps (1 per input port), use this command several times.

```
Example: add static-entry 2 3

    MAC address [00-00-00–00–00–00]?
    Input port, 0 for all [0]?
    Output port, 0 for none [0]? 2
    Output port, 0 to end [0]?
```

destination-address-filter ***mac_address***

Adds a destination address filter to the bridging router’s permanent database. Enter the command followed by the MAC address of the entry.

```
Example: add destination-address-filter

    Destination MAC address [00–00–00–00–00–00]?
```

Cache

Display the contents of a selected bridging port routing cache. If the port does not possess a cache, the following message appears: `Port X does not have a cache`.

Syntax: cache *port#*

```
Example: cache 1
```

```
Port number [1]? **3**

MAC Address    MC*   Entry Type       Age  Port(s)

00-00-93-00-C0-D0   PERMANENT          0  3 (FDDI/0)
00-00-00-11-22-33   STATIC             0  3 (FDDI/0)
```

MAC Address	6-byte MAC address of the entry
Entry Type	Specifies one of the following address entry types: • **Reserved** – Entries reserved by the IEEE802.1D Standard. • **Registered** – Entries consist of unicast addresses belonging to proprietary communications hardware attached to the box or multicast addresses enabled by protocol forwarders. • **Permanent** – Entries entered by the user in the configuration process that survive power cycles or system resets. • **Static** – Entries entered by the user in the console process that do not survive power cycles or system resets and are not effected by the aging timer. • **Dynamic** – Entries dynamically learned by the bridge that do not survive power cycles or system resets and that have an age associated with the entry. • **Free** – Locations in database that are free to be filled by address entries. • **Unknown** – Entry types unknown to the bridge. May be possible bugs and illegal addresses.
Age	Age in seconds of each dynamic entry. Age is decremented at each resolution intervals.
Port(s)	Specifies the port number associated with that entry and displays the interface name (the interface containing the cache).

Delete

Delete MAC address entries from the bridging router’s filtering database. This also includes dynamic and static address entries. After entering the command, you are prompted for the desired MAC address. You may also follow the command with the MAC address you want to delete and skip the prompt.

Syntax: delete *MAC-address*

Example: **delete 00–00–93–10–04–15**

```
MAC address [00–00–00–00–00–00]?
```

Flip

View specific MAC addresses in the canonical and non-canonical format by “flipping ” the address bit order. This command is useful for translating IEEE 802.5 addresses in their typical non-canonical format to the canonical format universally used by the bridge console and ELS (and vice versa).

Syntax: flip *MAC-address*

Example: **flip 00–00–00–33–44–55**

```
MAC address [00–00–00–00–00–00]? 00000334455
IEEE 802 canonical bit order:    00–00–00–33–44–55
IBM Token–Ring  native bit order: 00:00:00:CC:22:AA
```

List

Display information about the bridging router configuration or to display information about selected configuration or bridging options.

```
Syntax:   list     adaptive-bridge . . .
                   bridge . . .
                   conversion
                   database
                   filtering
                   source-routing
                   spanning-tree-protocol
                   transparent
                   tunnel
```

adaptive-bridge ***datagroup-option [sub-option]***

Lists all general information regarding the SR-TB bridge that converts between various types of bridging. There are a number of general datagroup options that are displayed under the **list adaptive-bridge**. These include the following:

- **Config** – Displays general information regarding the SR–TB bridge.
- **Counters** – Displays all SR–TB bridge counters.
- **Database** – Displays contents of the SR–TB bridge RIF database.

The following examples illustrates each of the **list adaptive-bridge** display options.

Example: **list adaptive-bridge config**

```
   Adaptive bridge:          Disabled
   Translation database size:  0
   Aging time:                 300 seconds
   Aging granularity           5 seconds


   Port  Segment  Interface   State     MTU
    1       1      Eth /0       Up       0
    2       3      Eth /1       Up       0
```

Conversion bridge	Shows the current state of the SR-TB conversion bridge. This value is displayed as either Enabled or Disabled.
Translation database size	Displays the current size of the SR-TB database, which contains MAC addresses and associated RIFs for the source-routing domain.
Aging time	Displays the aging timer setting in seconds. All SR-TB RIF database entries which exceed this time limit are discarded.
Aging granularity	Displays how often entries are scanned to look for expiration according to the aging timer.
Port	Displays the number of a port associated with conversion bridging.
Segment	Displays the source-routing segment number assigned to the port associated with conversion bridging.
Interface	Identifies the device connected to a conversion bridge network segment.
State	Indicates the current state of the conversion bridge port.
MTU	Specifies the maximum frame size (from the end of the RIF to the beginning of the FCS) that the conversion-bridge can transmit and receive.

Example: **list adaptive-bridge counters**

```
   Hash collision count: 28
   Adaptive. database overflow  count: 0
```

Hash Collision Count	Displays number of addresses that were stored (hashed) to the same location in the hash table. This number is accumulative and reflects the total number of hash collision incidents that occurred. Increases in this number may indicate a potential table size problem.
Adaptive Database Overflow	Displays the number of times that an address was overwritten as the conversion database table ran out of table space.

The database option of the **list adaptive bridge** command lets you select certain portions of the adaptive bridge RIF database to display. This is due to the potential size of the database. The display options include the following:

- **Address** – Displays the conversion bridge database related to that specific MAC address.
- **All** – Displays the entire database.
- **Port** – Displays all conversion bridge entries a specific port.
- **Segment** – Displays all conversion bridge entries associated with the port having the specified segment number.

The following examples illustrate each of the **list adaptive–bridge database** command options.

Note: These are only displayed if adaptive bridging is enabled.

```
list adaptive-bridge database address mac-address

list adaptive-bridge database all

list adaptive-bridge database port segment#

list adaptive-bridge database segment segment#
```

Each entry is displayed on two lines followed by a blank line. The following information is displayed for each entry:

```
Canonical Address  Interface  Port  Seg    Age  RIF: Type Direct
Length  LF

IBM MAC Address    RIF
```

Canonical address	Lists the MAC address of the node corresponding to this entry. This is displayed in IEEE 802 canonical (hexadecimal) format.
Interface	Displays the name of the network interface that learned this entry.
Port number	Displays the port number of the port that learned this address entry.
Segment	Displays the number of the segment that learned this address.
Age	Displays the entry age in seconds.
RIF Type	Displays the RIF type as SRF, STE, or ARE.
RIF Direction	Displays the RIF direction as Forward or Reverse.
RIF Length	Displays the RIF length in bytes.
RIF LF	Displays the largest frame value encoded in the RIF.
IBM MAC Address	Shows the MAC address of the node corresponding to this entry. This is displayed in the IBM token ring native bit order.
RIF	Displays the Routing Information Field learned from this node.

bridge

Lists all general information regarding the bridge router configuration.

Example: **list bridge**

```
Bridge ID (prio/add):  32768/08–00–2B–B1–E2–D1
Bridge state:          Enabled
Bridge type:           SRB
Bridge capability:     ASRT
Number of ports:       2
STP Participation:     IBM–SRB proprietary


                                                  Maximum
Port  Interface  State  MAC Address        Modes      MSDU  Segment   LNM
   1  TKR /0        Up   08–00–2B–10–AC–E2    SR      2096  010       ENA
   2  SL /1         Up   08–00–2B–00–00–00    SR      2054  002


SR bridge number:      1
SR virtual segment:    000
Adaptive segment:      000
```

Bridge ID	Unique ID used by the spanning tree algorithm in determining the spanning tree. Each bridge in the network is assigned a unique bridge identifier. The bridge priority is displayed in decimal followed by the hex address.
Bridge State	Indicates whether bridging is enabled or disabled.
Bridge Type	Displays the configured bridge type: NONE, SRB, TB, SRT, ADAPT, A/SRB, A/TB, or ASRT.
Number of Ports	Displays the number of ports configured.
Port	Specifies user-defined number assigned to an interface by the Add Port command.
Interface	Identifies devices connected to a network segment through the bridge.
State	Indicates the current state of the port: UP or DOWN.
MAC address	Displays the MAC address associated with that port in canonical bit order.

Modes	Displays the bridging mode for that port. • ***T*** – Indicates transparent bridging. • **SR** – Indicates source routing. • ***A*** – Indicates adaptive bridging.
MSDU	Specifies the maximum frame size (including the MAC header but not the FCS field) the source-routing bridge can transmit and receive on this interface.
Segment	Displays the source routing bridge segment number assigned to the port (if any).
SR bridge number	Displays the user-assigned source routing bridge number.
SR virtual segment	Displays the source-routing bridge virtual segment number show (if any).
Adaptive segment	Displays the number of the segment which is used in the source-routing domain to route to the transparent domain (through conversion).

conversion *datagroup-option*

Displays general information about the bridge's rules for converting frame formats based on the frame type. There are a number of general datagroups which may be displayed under the **list conversion** command. These include the following:

- **All** – Displays all rules.
- **Ethertype** – Displays rules for all Ethernet types or for a specific Ethernet type.
- **SAP** – Displays rules for a specific 802.2 SAP type.
- **SNAP** – Displays rules for a specific 802.2 SNAP type.

The following examples break down each of the **list conversion** display options.

```
list conversion all

list conversion ethertype

list conversion SAP

list conversion SNAP
```

database *datagroup-option*

Lists the contents of transparent filtering databases. There are a number of datagroups that can be displayed under the **list database** command. These include the following:

- **All-Ports** – Displays the entire transparent bridging database.
- **Dynamic** – Displays all dynamic (learned) address database entries.
- **Local** – Displays all local (reserved) address database entries.
- **Permanent** – Displays all permanent address database entries.
- **Port** – Displays address entries for a specific port.
- **Range** – Displays a range of database entries from the total transparent bridging filtering address database. A starting and stop MAC address is given to define the range. All entries falling within this range are displayed.
- **Static** – Displays static entries from the address database.

The following examples break down the **list database** command options. The first example also shows the related output.

Example: **list database all**

```
   MAC Address    MC*  Entry Type       Age  Port(s)


   00–00–00–00–AA–AA   Dynamic          295  4 (Eth /0)
   00–00–00–12–34–56   Perm/Source filter    2 (TKR /0)  –>  3–4
                                             1–2
   00–00–00–22–33–44   Permanent             1–2
                                             1–2
   00–00–00–33–44–55   Perm Dest filter      All
   00–00–00–55–66–77   Perm/Source filter    1–2,4


   08–00–2B–00–C0–D0   Registered            3 (FDDI/0)
   08–00–2B–10–04–15   Registered            1 (Eth /0)
   00–00–93–10–E4–F9   Dynamic          300  1 (Eth /0)
   00–00–93–90–04–A6   Dynamic          300  1 (Eth /0)
   00–00–A7–10–68–28   Dynamic          270  1 (Eth /0)
   01–80–C2–00–00–00*  Registered            1,3
   01–80–C2–00–00–01*  Reserved              All
   01–80–C2–00–00–02*  Reserved              All
   01–80–C2–00–00–03*  Reserved              All
   01–80–C2–00–00–0D*  Reserved              All
   01–80–C2–00–00–0E*  Reserved              All
   01–80–C2–00–00–0F*  Reserved              All
   03–00–00–00–80–00*  Reserved              All
   08–00–17–00–35–F9   Dynamic          300  1 (Eth /0)
   08–00–17–00–4D–DA   Dynamic          300  1 (Eth /0)
```

Note: The fields described below are displayed for all of the **list database** command options.

MAC Address	Specifies the address entry in IEEE 802 canonical format.
*MC**	An asterisk following an address entry indicates that the entry is flagged as a multicast address.

`Entry Type`

Specifies one of the following types:

- **Reserved** – Entries reserved by the IEEE802.1D standard.
- **Registered** – Entries consist of unicast addresses belonging to interfaces participating in the bridge or multicast addresses enabled by protocol forwarders.
- **Permanent** – Entries entered by the user in the configuration process that survive power cycles or system resets.
- **Static** – Entries entered by the user in the console process that do not survive power cycles or system resets and are ageless.
- **Dynamic** – Entries dynamically learned by the bridge that do not survive power cycles or system resets and that have an age associated with the entry.
- **Free** – This type is not used and is not normally seen except in occasional race conditions between the console and the bridge.
- **Unknown** – Unknown entry type. May indicate a software bug. Report the hex entry type to Customer Service.

`Age`

Refers to the age (in seconds) of each dynamic entry. Age is decremented at each resolution interval. The true age is calculated by subtracting this value from the max_age value.

`Port(s)`

Specifies the outgoing port number(s) for that entry. Device type is also listed for single port entries. If dynamic entry on IP tunnel, the port is "5" for IP tunnel.

Examples:

```
            list database dynamic

            list database local

            list database permanent

            list database port port#

            list database static

            list database range
```

```
First MAC address [00-00-00-00-00-00]? 0-00-93-00-C0-D0
Last MAC address [FF-FF-FF-FF-FF-FF]? 1-80-C2-00-00-00


MAC Address    MC*  Entry Type       Age  Port(s)


00-00-93-00-C0-D0   Registered            3 (FDDI/0)
00-00-93-10-04-15   Registered            1 (Eth /0)
01-80-C2-00-00-00   Registered            1,3
```

filtering ***datagroup-option***

Displays general information about the bridge's protocol filtering databases. There are a number of general datagroups that can be displayed under the **list filtering** command. These include the following:

- **Al**l – Displays all filtering database entries.
- **Ethertype** – Displays Ethernet protocol type filter database entries.
- **SAP** – Displays SAP 802.2 SAP type entries.
- **SNAP** – Displays SNAP 802.2 SNAP type entries.

The following examples break down each of the **list filtering** display options.

Example: **list filtering all**

```
Ethernet type 0800 is routed on ports 1
IEEE 802.2 destination SAP 42 is routed on ports 1
IEEE 802 SNAP PID 00-00-00-08-00 is routed on ports 2-3
```

Descriptors used in explaining how packets are communicated include the following:

- **Routed** – Describes packets that are passed to a routing forwarder to be forwarded.
- **Filtered** – Describes packets that are administratively filtered by user set protocol filters.
- **Bridged and routed** – Describes a protocol identifier containing a protocol entity within the system that is not a forwarder. An example of this would be a link level echo protocol. Unicast packets from this protocol are bridged or locally processed if being sent to a registered address. Multicast packets are forwarded and locally processed for a registered multicast address.

`Example:` **`list filtering ethertype`**

```
Ethernet type (in hexadecimal), 0 for all [0]? O800
Ethernet type 0800 is routed on ports 1
```

`Example:` **`list filtering SAP`**

```
SAP (in hexadecimal), 100 for all [100]? 42
IEEE 802.2 destination SAP 42 is routed on ports 1
```

`Example:` **`list filtering SNAP`**

```
SNAP Protocol ID, return for all [00–00–00–00–00]?
IEEE 802 SNAP PID 00–00–00–08–00 is routed on ports 2–3
```

source-routing

Displays source-routing bridge configuration information. There are a number of general datagroup options that can be displayed under the **list source-routing** command. These include the following:

- **Configuration** – Displays general information regarding the SRB bridge.
- **Counters** – Displays all SRB bridge counters.

- **State** – Displays contents of all related SR–TB bridge databases.

The following examples illustrate the output from each of the **list source-routing** display options.

Example: **`list source-routing configuration`**

```
   Bridge number:              1
   Bridge state:               Enabled
   Maximum STE hop count       14
   Maximum ARE hop count       14
   Virtual segment:            003

   Port  Segment  Interface   State     MTU    STE Forwarding LNM
    2    001      TKR /0      Enabled   4399   Yes            ENA
    3    002      TKR /1      Enabled   4399   Yes
    –    003      Adaptive    Enabled   1470   Yes
```

Bridge number	Displays the bridge number (in hexadecimal) assigned to this bridge.
Bridge State	Indicates whether bridging is enabled or disabled.
Maximum STE hop count	Displays the maximum hop count for Spanning Tree Explorer frames transmitting from the bridge for a given interface associated with source routing bridging.
Maximum ARE hop count	Displays the maximum hop count for All-Route Explorer frames transmitting from the bridge for a given interface associated with source routing bridging.
Virtual segment	Displays the virtual segment number assigned for 1:N bridging.
Port	Lists the numbers of ports associated with source routing bridging
Segment	Lists the assigned segment numbers for networks associated with source routing bridging.
Interface	Lists the associated interface names. Adaptive is listed for interfaces participating in the SR-TB feature.
State	Lists the current port state (Enabled or Disabled).

MTU	Lists the MTU size set for that port.
STE Forwarding	Indicates whether Spanning Tree Explorers received on this port are forwarded (Yes) and whether STEs from other ports go out this port.

The counters option has further subgroups of information that can be displayed using the **list source-routing** command. These include the following:

- **All-ports** – Displays counters for all ports.
- **Port** – Displays counters for a specific port.
- **Segment** – Displays counters for the port corresponding to a specific segment.

The following examples illustrate each of the **list source-routing** display options.

Example: **list source-routing counters all-ports**

```
SRT>list source counters all–ports
Counters for port 2, segment 001, interface TKR /0:
SRF frames received:          0    sent:          0
STE frames received:          0    sent:          0
ARE frames received:        648    sent:          0
SR frames sent as TB:                             0
TB frames sent as SR:                          2057
Dropped, input queue overflow:                    0
Dropped, source address filtering:                0
Dropped, invalid RIF length:                        0
Dropped, duplicate segment:                   2594
Dropped, segment mismatch:                        0
Dropped, Duplicate LAN ID or tree error:          0
Dropped, STE hop count exceeded:                  0


Counters for port 3, segment 002, interface TKR /1:
SRF frames received:          0    sent:          0
STE frames received:          0    sent:          0
ARE frames received:        825    sent:          0
SR frames sent as TB:                             0
TB frames sent as SR:                          2041
Dropped, input queue overflow:                    0
Dropped, source address filtering:                0
Dropped, invalid RI length:                       0
Dropped, duplicate segment:                   3300
Dropped, segment mismatch:                        0
Dropped, Duplicate LAN ID or tree error:          0
Dropped, STE hop count exceeded:                  0
```

Port	Lists the numbers of ports associated with source routing bridging
Segment	Lists the source-routing segment numbers in hex.
Interface	Lists the name of the network interface.
SRF Frames Received/Sent	Lists the number of Specifically Routed frames received or sent on this bridge.
STE Frames Received/Sent	Lists the number of Spanning Tree Explorer frames received or sent on this bridge.
ARE Frames Received/Sent	Lists the number of All Route Explorer frames received or sent on this bridge.
SR Frames Sent as TB	Lists the number of source routing frames received on this interface that were sent as Transparent Bridge frames.
TB Frames Sent as SR	Lists the number of Transparent Bridge frames received on this interface that were sent as source routing frames.
Dropped, input queue	Lists the number of frames arriving on this interface that were dropped because the input queue to the forwarder overflowed.
Dropped, source address filtering	Lists the number of frames arriving on this interface that were dropped because this source address matched a source address filter in the filtering database.
Dropped, destination address filtering	Lists the number of frames arriving on this interface that were dropped because this destination address matched a destination address filter in the filtering database.

Dropped, protocol filtering	Lists the number of frames arriving on this interface that were dropped because their protocol identifier was one that is being administratively filtered.
Dropped, duplicate RIF length	Lists the number of frames arriving on this interface that were dropped because the RIF length is less than 2 or greater than 30.
Dropped, duplicate segment	Lists the number of frames arriving on this interface that were dropped because of a duplicate segment in the RIF. **Note:** Normal ARE frame behavior cause this counter to increase.
Dropped, segment mismatch	Lists the number of frames arriving on this interface that were dropped because the outgoing segment number does not match any in this bridge.

Example: **list source-routing counters port *port#***

```
   Counters for port 3, segment 002, interface TKR /1:
   SRF frames received:          0    sent:          0
   STE frames received:          0    sent:          0
   ARE frames received:       1140    sent:          0
   SR frames sent as TB:                             0
   TB frames sent as SR:                          2931
   Dropped, input queue overflow:                    0
   Dropped, source address filtering:                0

   Dropped, invalid RIF length:                       0
   Dropped, duplicate segment:                     4560
   Dropped, segment mismatch:                         0
   Dropped, Duplicate LAN ID or tree error:           0
   Dropped, STE hop count exceeded:                   0
   Dropped, ARE hop count exceeded:                   0
   Dropped, no buffer available to copy:              0
   Dropped, MTU exceeded:                             0
```

Example: **list source-routing counters segment 2**

```
Counters for port 3, segment 002, interface TKR /1:
SRF frames received:          0    sent:          0
STE frames received:          0    sent:          0
ARE frames received:       1249    sent:          0
SR frames sent as TB:                             0
TB frames sent as SR:                          3200
Dropped, input queue overflow:                    0
Dropped, source address filtering:                0
Dropped, dest address filtering:                  0
Dropped, protocol filtering:                      0
Dropped, invalid RI length:                       0
Dropped, duplicate segment:                    4996
Dropped, segment mismatch:                        0
Dropped, Duplicate LAN ID or tree error:          0
```

spanning-tree protocol

Displays Spanning Tree protocol information. The Spanning Tree protocol is used by the transparent bridge to form a loop-free topology. There are a number of general datagroup options which may be displayed under the **list spanning-tree-protocol** command. These include the following:

- **Configuration** – Displays information concerning the Spanning Tree protocol.
- **Counters** – Displays the Spanning Tree protocol counters.
- **State** – Displays the current Spanning Tree protocol state information.
- **Tree** – Displays the current spanning tree information including port, interface, and cost information.

The following examples illustrate each of the **list spanning-tree-protocol** display options.

Example: **list spanning-tree-protocol configuration**

```
Bridge ID (prio/add):  32768/08–00–2B–00–84–EA
Bridge state:          Enabled
Maximum age:           20 seconds
Hello time:            2 seconds
Forward delay:         15 seconds
Hold time:             1 seconds
Filtering age:         320 seconds
Filtering resolution:  5 seconds
```

```
   Port  Interface  Priority  Cost   State
      1  FDDI/0     128       10     Enabled
      4  Eth /0     128       100    Enabled
    128  Tunnel     128       65535  Enabled
```

Example: **list spanning-tree-protocol counters**

```
   Time since topology change (seconds)          0
   Topology changes:                             3
   BPDUs received:                               0
   BPDUs sent:                                3866

Port  Interface  BPDUs received  BDPU input overflow  Forward transitions
   1  FDDI/0            0              0                   1
   4  Eth /0            0              0                   1
 128  Tunnel            0              0                   1
```

Example: **list spanning-tree-protocol state**

```
   Designated root (prio/add):    32768/08–00–2B–00–84–EA
   Root cost:                     0
   Root port:                     Self
   Current (root) maximum age:    20 seconds
   Current (root) hello time:     2 seconds
   Current (root) Forward delay:  15 seconds
   Topology change detected:      FALSE
   Topology change:               FALSE

   Port  Interface   State
      1  FDDI/0      Forwarding
      4  Eth /0      Forwarding
    128  Tunnel      Forwarding
```

Example: **list spanning-tree-protocol tree**

```
Port                      Designated   Desig.                Designated Des.No.
Interface                       Root   Cost                       Bridge Port
  1  FDDI/0  32768/00–00–93–00–84–EA   0 32768/00–00–93–00–84–EA     80–01
  4  Eth /0  32768/00–00–93–00–84–EA   0 32768/00–00–93–00–84–EA     80–04
128  Tunnel  32768/00–00–93–00–84–EA   0 32768/00–00–93–00–84–EA     80–80
```

transparent

Displays transparent bridge configuration information. There are a number of general datagroup options which may be displayed under the **list transparent** command. These include the following:

- **Configuration** – Displays information concerning the transparent bridge.

- **Counters** – Displays the transparent bridge counters. You may use **all-ports** after the command to display the counters for all ports or enter the specific port number after the command to display counters for one port.

- **State** – Displays the transparent state information.

The following examples illustrate each of the **list transparent** display options.

Example: **`list transparent configuration`**

```
Filtering database size:    5141
Aging time:                 300 seconds
Aging granularity           5 seconds

Port  Interface   State     MTU
   1  FDDI/0      Enabled   0
   4  Eth /0       Enabled   0
 128  Tunnel      Enabled   0
```

Example: **`list transparent counters all-ports`**

```
Counters for port 4, interface Eth /0:
Total frames received by interface:           25885
Frames submitted to bridging:                 13732
Frames submitted to routing:                   6101
Dropped, source address filtering:                0
Dropped, dest address filtering:              12677
Dropped, protocol filtering:                      0
Counters for port 128, interface Tunnel:
Total frames received by interface:               0
Frames submitted to bridging:                     0
Frames submitted to routing:                      0
Dropped, source address filtering:                0
Dropped, dest address filtering:                  0
Dropped, protocol filtering:                      0
Dropped, no buffer available to copy:             0
Dropped, input queue overflow:                    0
Dropped, source port blocked:                     0
Frames sent by bridging:                       5327
Dropped, dest port blocked:                       0
Dropped, transmit error:                          0
Dropped, too big to send on port:                 0
```

Example: **list transparent counters port 4**

```
Counters for port 4, interface Eth /0:
Total frames received by interface:          25885
Frames submitted to bridging:                13732
Frames submitted to routing:                  6101
Dropped, source address filtering:               0
Dropped, dest address filtering:             12677
Dropped, protocol filtering:                     0
Dropped, no buffer available to copy:         6073
Dropped, input queue overflow:                 122
Dropped, source port blocked:                   31
Frames sent by bridging:                       388
Dropped, dest port blocked:                      0
Dropped, transmit error:                         0
Dropped, too big to send on port:                0
```

Example: **list transparent state**

```
Filtering database size:                      5141
Number of static entries:                        0
Number of dynamic entries:                      10
Hash collision count:                            1
Filtering database overflow count:               0
```

tunnel

Displays tunnel configuration information. There are general datagroup options which may be displayed under the **list tunnel** command. These include:

- **Bridges** – Displays tunnel bridge information.
- **Config** – Displays information concerning the tunnel configuration.

The following examples illustrate each of the **list tunnel** display options.

```
list tunnel bridges
list tunnel config
```

Netbios-filter

Accesses the `NETBIOS Filter>` prompt. NETBIOS filtering console commands may be entered at the `NETBIOS Filter>` prompt.

Syntax: netbios-filter

Example: **netbios-filter**

Note: If the NETBIOS filtering feature has not been purchased for your bridging router software load, you receive the following message when you try to use this command:

```
NETBIOS Filtering is not available in this load.
```

Exit

Exit the ASRT console process and return to the GWCON environment.

Syntax: exit

`Example:` **`exit`**

NETBIOS Filtering Console Commands

This section summarizes and then explains the NETBIOS Filtering console commands. These commands let you monitor and display NETBIOS Filter information as an added feature to ASRT bridging. Console commands are entered at the `NETBIOS Filter>` prompt. This prompt is accessed by entering the **netbios-filter** command at the `ASRT>` prompt.

Table 6–2 NETBIOS Filtering Console Commands Summary

Command	Function
List	Displays all information concerning created filters.
Exit	Exits the NETBIOS Filtering console process and returns to the ASRT console environment.

? (Help)

List the commands that are available from the current prompt level. You can also enter a **?** after a specific command name to list its options.

Syntax: ?

`Example:` **`?`**

List

Display all information concerning created filters.

Syntax: list byte-filter-lists
filters
name-filter-lists

byte-filter-lists

Displays information related to filter items in the specified byte-filter-list.

Example: **list byte-filter-lists**

```
BYTE Filter–List Name:  Engineering
BYTE Filter–List Default:  Exclusive



Filter Item #  Inc/Ex    Byte Offset  Pattern      Mask

   1          Inclusive     14        0x123456    0xFFFFFF
   2          Exclusive      0        0x9876      0xFFFF
   3          Exclusive     28        0x1000000   0xFF00FF00
```

Filter Item#	Specifies filter item number. Filter items are evaluated in numerical order when determining the Inclusive/Exclusive status of the filter list.
Inc/Ex	Specifies the default status of the filter item.
Byte-offset	Specifies the byte offset (in decimal) into the packet being filtered starting from the beginning of the NETBIOS header of the packet.

Pattern	The hexadecimal number used to compare with the bytes starting at the byte-offset offset of the NETBIOS header after the optional masking takes place. Syntax rules for hex-pattern include no 0x in front, a maximum of 32 digits, and an even number of hex numbers.
Mask	If present, must be the same length as hex-pattern and is logically ANDed with the bytes in the packet starting at byte-offset before the result is compared for equality with hex_pattern. If the hex-mask argument is omitted, it is considered to be all binary 1's.

filters

Displays information related to all configured filters.

Example: **list filters**

```
NETBIOS Filtering: Disabled

    Port #  Direction  Filter List Handle(s) Pkts Filtered

      3     Output     nlist                      0
```

name-filter-lists

Displays information related to filter items in the specified name-filter-list.

Example: **list name-filter-lists**

```
NAME Filter List Name: nlist
NAME Filter List Default: Exclusive

 Filter Item #  Type    Inc/Ex     Hostname   Last Char

      1         ASCII   Inclusive  EROS        <0x03>
      2         ASCII   Inclusive  ATHENA
      3         ASCII   Exclusive  FOOBAR
```

Filter Item#	Specifies filter item number. Filter items are evaluated in numerical order when determining the Inclusive/Exclusive status of the filter list.
Inc/Ex	Specifies the status of the filter item.

Type	ASCII indicates a host-name filter item added as ASCII characters. Hex indicates a host-name filter item added as hexadecimal numbers.
Host-name	ASCII string up to 16 characters long. A “?” can be used in host-name to indicate a single character wildcard. An “*” can be used as the final character of host-name to indicate a wildcard for the remainder of the host-name. If host-name contains fewer than 15 characters, it is padded to the 15th character with ASCII spaces. Host-name cannot contain the following characters: `. / \ [ ] : |< > + = ; ,` <space>
Last char	The offset (in hexadecimal) of the last character to be used from host-name for a comparison if host-name contains fewer than 16 characters. It is a hexadecimal number (with no 0x in front of it) indicating the value to be used for the last character. If the LAST argument is not specified on a host-name containing fewer than 16 characters, then a “?” wildcard is supplied for the 16th character.

Exit

Return to the previous `ASRT>` prompt level.

Syntax: exit

`Example:` **`exit`**

7

Configuring TCP/IP Host Services

This chapter describes how to configure the TCP/IP Host Services (TCP/IP Host) protocol and how to use the TCP/IP Host configuration commands.

Accessing the TCP/IP Host Configuration Environment

For information on how to access the TCP/IP Host configuration environment, see the chapter entitled “Getting Started” in the *System Software Guide*.

Basic Configuration Procedures

The following sections describe the basic configuration procedures for enabling TCP/IP Host Services on your bridging router.

Setting the IP Address

To minimally configure TCP/IP Host services, assign the bridging router an IP address by using the **set ip-host address** command. This IP address is associated with the bridging router as a whole, instead of being associated with a single interface.

Adding a Default Gateway

To configure a default gateway for your bridging router, use the **add default gateway** command. After the bridging router is designated as the default gateway, it exists as the static (permanent) authoritative router that receives all packets with destinations that cannot be found in the routing table.

This action generally occurs dynamically through resident routing protocols. The **add default gateway** command is used for instances when static routing is desired (for example, as a precaution against power failures) or in cases when routing information cannot be obtained dynamically.

The assigned IP address and default gateway information is saved immediately.

Enabling TCP/IP Host Services

After assigning the bridging router IP address and default gateway information, use the **enable services** command to enable TCP/IP Host Services.

TCP/IP Host Configuration Commands

This section summarizes and explains all the TCP/IP Host configuration commands. The TCP/IP Host configuration commands allow you to specify network parameters for the TCP/IP Host bridge. Restart the router to activate the configuration commands. Enter the TCP/IP Host configuration commands at the `TCP/IP-Host config>` prompt.

Table 7–1 TCP/IP Host Configuration Commands Summary

Command	Function
? (Help)	Lists all of the TCP/IP Host configuration commands, or lists the options associated with specific commands.
Add	Adds a default-gateway.
Delete	Deletes a default-gateway.
Disable	Disables TCP/IP Host Services, router-discovery processes, and RIP listening.
Enable	Enables TCP/IP Host Services, router-discovery processes, and RIP listening.
List	Lists the current TCP/IP Host configuration.
Set	Sets the bridging router’s IP address.
Exit	Exits the TCP/IP Host configuration process and returns to the CONFIG environment.

? (Help)

List the commands that are available from the current prompt level. You can also enter a **?** after a specific command name to list its options.

Syntax: ?

`Example:` **`?`**

Add

Add default gateways (routers) to your configuration. When your bridging router has been designated as the default gateway, it exists as the static authoritative router that receives all packets with destinations not found in the routing table.

Note: The **add** command configures your router to be the default gateway for another router. This command is issued from the other router.

Default gateways are used when trying to send packets to IP destinations that are off the local connection. The routing table is then built up through redirect processing. An attempt is made to detect routers that disappear.

Syntax: add default-gateway *def-gateway-IP-address*

`Example:` **`add default–gateway 123.45.67.89`**

```
Default-Gateway address [0.0.0.0]?
```

Delete

Delete default gateways from your bridging router configuration. Enter the IP address of the default gateway you want to remove after the **delete** command.

Syntax: delete default-gateway *def-gateway-IP-address*

`Example:` **`delete default–gateway 123.45.67.89`**

```
Enter address to be deleted [0.0.0.0]?
```

Disable

Disable the following TCP/IP functions:

- TCP/IP Host Services
- Router-discovery processes
- RIP listening

Syntax: disable rip-listening
router-discovery
services

rip-listening

Disables the building of routing table entries that were gathered by listening to the RIP protocol. By default, RIP-listening is disabled.

```
Example: disable rip–listening
```

router-discovery

Disables the ability to learn default gateways by receiving ICMP Router Discovery messages. By default, router discovery is enabled.

```
Example: disable router-discovery
```

services

Disables the TCP/IP Host Services protocol entirely. By default, TCP/IP Host Services are enabled.

```
Example: disable services
```

Enable

Enable the following TCP/IP functions:

- TCP/IP Host Services
- Router discovery processes
- RIP listening

```
Syntax:   enable   rip-listening
                   router-discovery
                   services
```

rip-listening

Enables the building of routing table entries that were gathered by the bridge listening to the RIP protocol.   RIP-listening is disabled by default.

Example: **`enable rip–listening`**

router-discovery

Enables the learning of default gateways through reception of ICMP Router Discovery messages. By default, router discovery is enabled.

Example: **`enable router–discovery`**

services

Enables the TCP/IP Host Services protocol. By default, TCP/IP Host Services are enabled.

Example: **`enable services`**

List

Display information about the current TCP/IP Host configuration.

```
Syntax:   list   all
```

Example: **`list all`**

```
IP-Host IP address : 128.185.142.1
Address mask : 255.255.255.0

Default Gateway IP-address(es)
128.185.142.47

TCP/IP-Host Services Enabled.

RIP-LISTENING Disabled.

Router Discovery Enabled.
```

IP-Host IP address	Displays the current IP-Host IP address.
Address mask	Displays the current IP-Host IP subnet address mask.
Default Gateway IP-address(es)	Displays the current default gateway IP address.
TCP/IP Host Services	Displays whether TCP/IP Host Services is enabled or disabled.
RIP-LISTENING	Displays whether RIP-LISTENING is enabled or disabled.
Router Discovery	Displays whether Router Discovery is enabled or disabled.

Set

Set the bridging router’s IP address. You must assign the bridging router an IP address before enabling TCP/IP Host Services.

Syntax: set IP-Host address *IP-host-address*

Example: **set ip-host address 123.45.67.89**

```
   IP-Host address [0.0.0.0]?
   Address mask [255.255.0.0]?
```

Exit

Return to the previous prompt level.

Syntax: exit

Example: **exit**

8

Monitoring TCP/IP Host Services

This chapter describes how to monitor the TCP/IP Host Services on the bridging router.

Accessing the TCP/IP Host Console Environment

For information about accessing the TCP/IP Host console environment, see the chapter entitled “Getting Started” in the *System Software Guide*.

TCP/IP Host Console Commands

This section explains the TCP/IP Host console commands. These commands allow you to view parameters and enter information requests from the active console. Enter these commands at the `HST>` prompt.

Table 8–1 TCP/IP Host Console Commands Summary

Command	Function
? (Help)	Lists all of the TCP/IP Host console commands, or lists the options associated with specific commands.
Dump	Displays the current IP routing table. One line is printed for each destination.
Interface	Displays the IP addresses configured for each interface.
Ping	Continuously pings a given destination, printing a line for each response received.
Traceroute	Displays the hop-by-hop route to a given destination.
Routers	Displays the list of all IP routers known to the bridging router.
Exit	Exits the TCP/IP Host console process and returns to the GWCON environment.

? (Help)

List the commands that are available from the current prompt level. You can also enter a **?** after a specific command name to list its options.

Syntax: ?

```
Example: ?

   dump
   interfaces
   ping
   traceroute
   routers
   exit
```

Dump

Display the current IP routing table. One line is printed for each destination. Many of the entries that displayed are the result of ICMP redirects.

Syntax: dump

Example: **dump**

```
   Type   Dest net         Mask     Cost Age  Next hop(s)

   Stat   0.0.0.0          00000000  0    0    128.185.142.47
   Dir*  128.185.142.0     FFFFFF00  1    0    BDG/0

   Default gateway in use.
   Type Cost Age  Next hop
   Stat 0    0    128.185.142.47

   Routing table size: 768 nets (43008 bytes), 2 nets known
```

Type (route type) Indicates how the route was derived.

- **Sbnt** – Indicates that the network is subnetted; such an entry is a placeholder only.
- **Dir** – Indicates a directly connected network or subnet.
- **RIP** – Indicates the route was learned through the RIP protocol.
- **Del** – Indicates the route was deleted.
- **Stat** – Indicates a statically configured route.
- **EGP** – Indicates routes learned through the EGP protocol.
- **EGPR** – Indicates routes learned through the EGP protocol that are readvertised by OSPF and RIP.
- **Fltr** – Indicates a routing filter.
- **SPF** – Indicates that the route is an OSPF intra-area route.
- **SPIA** – Indicates that it is an OSPF inter-area routes.
- **SPE1**, **SPE2** – Indicates OSPF external routes (type 1 and 2 respectively).

	• **Rnge** – Indicates a route type that is an active OSPF area address range and is not used in forwarding packets.
Dest net	Displays the IP address of the destination network/subnet.
Mask	Displays the IP address mask.
Cost	Displays the Route Cost.
Age	Displays the time that has elapsed since the routing table entry was last refreshed for RIP and EGP routes.
Next Hop	Displays the IP address of the next router on the path toward the destination host. Also displayed is the interface type used by the sending router to forward the packet.
Default gateway	Displays the IP address of the default gateway along with the route type, cost, age, and next hop information associated with that entry.
Routing table size	Displays the current size (in networks and kilobytes) of the current table. Also identifies the number of networks (nets) known to the host.

Interface

List the IP addresses configured for each interface. When TCP/IP Host Services are running over the bridge, a single address is displayed on the console as Bridge/0.

Syntax: interface

Example: **interface**

```
Interface   IP Address(es)        Mask

  BDG/0     128.185.142.16        255.255.255.0
```

`Interface`	Displays a single address as Bridge/0 when TCP/IP Host Services are running over the bridge. When services are disabled, interfaces with their corresponding numbers are displayed.
`IP Address`	Displays the IP address of the TCP/IP Host Services interface.
`Mask`	Displays the IP address subnet mask.

Ping

Use the **ping** command to have the router send ICMP Echo Requests to a given destination once a second (pinging) and watch for a response. This command can be used to isolate trouble in an internetwork environment.

This process is done continuously, incrementing the ICMP sequence number with each additional packet. Matching received ICMP Echo responses are reported with their sequence number and the round trip time. The granularity (time resolution) of the round trip time calculation is usually (depending on platform) around 20 milliseconds. The pinging process stops when a character is typed at the console. At that time, a summary of packet loss, round trip time, and number of ICMP destination unreachables received is displayed.

When a multicast address is given as destination, there may be multiple responses printed for each packet sent, one for each group member. Each returned response is displayed with the source address of the responder.

Note: The size of the ping (number of data bytes in the ICMP message, excluding the ICMP header) is 56 bytes, and the TTL used is 60.

Syntax: ping *interface-address*

Example: **ping 128.185.142.11**

```
PING 128.185.142.11: 56 data bytes
64 bytes from 128.185.142.11: icmp_seq=0. time=0. ms
64 bytes from 128.185.142.11: icmp_seq=1. time=0. ms
64 bytes from 128.185.142.11: icmp_seq=2. time=0. ms
64 bytes from 128.185.142.11: icmp_seq=3. time=0. ms
64 bytes from 128.185.142.11: icmp_seq=4. time=0. ms
64 bytes from 128.185.142.11: icmp_seq=5. time=0. ms

----128.185.142.11 PING Statistics----
6 packets transmitted, 6 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 0/0/0
```

Traceroute

Display the entire path to a given destination, hop by hop. For each successive hop, the **traceroute** command sends out three probes and prints the IP address of the responder along with the round trip time associated with the response. If a particular probe receives no response, an asterisk is printed. Each line in the display relates to this set of three probes, with the left most number indicating the distance from the router executing the command (in router hops).

The traceroute is complete when either the destination is reached, an ICMP Destination Unreachable message is received, or the path length reaches 32 router hops.

Syntax: traceroute *interface-address*

Example: **traceroute 128.185.142.239**

```
TRACEROUTE 128.185.142.239: 56 data bytes
   1 128.185.142.7 16 ms 0 ms 0 ms
   2 128.185.123.22 16 ms 0 ms 16 ms
   3 * * *
   4 * * *
   5 128.185.124.110 16 ms ! 0 ms ! 0 ms !
```

TRACEROUTE	Displays the destination area address and the size of the packet being sent to that address.
1	The first trace showing the destination's NSAP and the amount of time it took the packet to arrive at the destination. The packet is traced three times.

Destination unreachable	Indicates that no route to destination is available.
*1 * * * 2 * * **	Indicates that the router is expecting some form of response from the destination, but the destination is not responding.

When a probe receives an unexpected result (see the above output example), several indicators can be printed. These indicators are explained in the following table.

IN	Indicates that an ICMP Destination Unreachable (net unreachable) was received.
!H	Indicates that an ICMP Destination Unreachable (host unreachable) was received.
!P	Indicates that an ICMP Destination Unreachable (protocol unreachable) was received.
!	Indicates that the destination is reached, but the reply sent by the destination was received with a TTL of 1. This usually indicates an error in the destination, prevalent in some versions of UNIX, whereby the destination is inserting the probe's TTL in its replies. This unfortunately leads to a number of lines consisting solely of asterisks before the destination is finally reached.

Routers

Display the list of all IP routers that are known to the bridging router. Routers can be learned through:

- Static configuration (using the above **add default-gateway** command)
- Received ICMP redirects
- ICMP Router Discovery messages (if configured)
- RIP updates (if configured)

Each router is listed with its origin, its priority (used when selecting the default route), and its lifetime (the number of seconds before the router is declared invalid unless it is heard from again).

Syntax: routers

`Example:` **`routers`**

Exit

Exit the TCP/IP Host console process and return to the GWCON environment.

Syntax: exit

`Example:` **`exit`**

Index

Symbols

?(Help)
  ASRT Bridge configuration command, 4–3
  ASRT Bridge console command, 6–2
  LNM configuration command, 4–39
  NETBIOS Filtering configuration command, 4–43
  NETBIOS Filtering console command, 6–25
  TCP/IP Host Services configuration command, 7–3
  TCP/IP Host Services console command, 8–2
  Tunnel configuration command, 4–57

Numbers

8209 bridges, 3–10

A

Adaptive Source Routing Transparent Bridge (ASRT), 2–1, 3–1
  basic configuration procedures, 5–1
  bit ordering in STB and SRB bridges, 2–40
  bridge–only management, 3–3
  bridging basics, 1–1
  bridging tunnel, 3–1
    encapsulation and OSPF, 3–3
  configuration matrix, 5–4
  configuring, 2–41, 4–1, 5–1
  description of, 2–39
  eliminating packet size problems, 2–40
  Ethernet packet format translation, 2–7
  MIB support, 3–4
  monitoring, 6–1
  overview, 1–1
    complex bridges, 1–5
    CSMA/CD MAC frames, 1–9
    local bridges, 1–5
    MAC bridge frame formats, 1–8
    operation and protocol architecture, 1–6
    point–to–point links, 1–6
    remote bridges, 1–5
    simple bridge, 1–4, 1–6
    token–ring MAC frames, 1–10
  protocol filtering, 2–21
  pseudo serial ethernet, 3–8

source routing bridge (SRB), 2–13
  operation, 2–14
  source routing frames, 2–15
  spanning tree explore option, 2–19
spanning tree bridges, 2–7
spanning tree explore option
  balancing traffic loads, 2–20
  simulating a network, 2–20
SR–TB bridging, examples, 2–34
SR–TB Conversion, operation, 2–29, 2–30
SR–TB conversion, description of, 2–28
SRB terminology and concepts, 2–22
  bridge instance, 2–22
  bridge number, 2–23
  explorer frames, 2–23
  interface number, 2–23
  route, 2–24
  route discovery, 2–24
  segment number, 2–24
  source routing, 2–24
TCP/IP host services, 3–3
terminology and concepts, 2–8, 2–37
  aging time, 2–8
  all routes broadcast, 2–37
  all stations broadcast, 2–37
  bridge, 2–8, 2–37
  bridge address, 2–8
  bridge hello time, 2–8
  bridge identifier, 2–9
  bridge maximum age, 2–9
  bridge number, 2–37
  bridge priority, 2–9
  desiganted port, 2–10
  destination bridge, 2–10
  explorer frames, 2–37
  filtering and permanent databases, 2–10
  parallel bridges, 2–11
  path cost, 2–11
  port, 2–11
  port ID, 2–12
  port number, 2–12
  port priority, 2–12
  resolution, 2–12
  ring number, 2–37
  root bridge, 2–12
  root port, 2–12
  route, 2–38
  route designator, 2–38
  route discovery, 2–38
  segment number, 2–38
  single route broadcasting, 2–38
  source routing bridging, 2–38
  spanning tree, 2–12, 2–38
  transparent bridging, 2–13, 2–38
transparent bridge (STB), 2–1
  network requirements, 2–2
  operation of, 2–3
  routers and transparent bridges, 2–2
  shaping the spanning tree, 2–4
transparent–source routing compatibility, 2–39
Add
  ASRT Bridge configuration command, 4–3
  ASRT Bridge console command, 6–3
  TCP/IP Host Services configuration command, 7–3
  Tunnel configuration command, 4–57
Address entries
  dynamic, 4–32, 6–4, 6–14
  free, 4–33, 6–4

permanent, 4–32, 6–4, 6–14
registered, 4–32, 6–4, 6–14
reserved, 4–32
static, 4–32
ARP
with AppleTalk threading, 3–15
with IP threading, 3–12
ASRT. *See* Adaptive Source Routing Transparent Bridge
ASRT Bridge configuration commands
?(Help), 4–3
add, 4–3
and IP tunnel, 4–38
change, 4–14
delete, 4–14
disable, 4–17
enable, 4–20
exit, 4–38
functional address to group address mapping, 4–9
IP tunnel commands, 4–55
list, 4–24
LNM, 4–33
LNM commands, 4–38
LNM configuration commands
?(Help), 4–39
disable, 4–41
enable, 4–39
exit, 4–42
list, 4–42
set, 4–42
NETBIOS filtering commands, 4–43
NETBIOS filtering concepts, 3–4
NETBIOS filtering configuration commands
?(Help), 4–43
create, 4–44
delete, 4–44
disable, 4–45
enable, 4–45
exit, 4–55
filter–on, 4–46
list, 4–47
update, 4–48
netbios–filtering, 4–33
port maps explained, 4–6
set, 4–33
summary of, 4–1
tunnel, 4–38
tunnel configuration commands
?(Help), 4–57
add, 4–57
delete, 4–57
exit, 4–59
list, 4–58
ASRT Bridge console commands
?(Help), 6–2
add, 6–3
cache, 6–3
delete, 6–5
exit, 6–25
flip, 6–5
list, 6–6
NETBIOS filtering console commands, 6–25
?(Help), 6–25
exit, 6–28
list, 6–26
netbios–filter, 6–24
summary of, 6–1
ASRT Bridge LNM feature, prompt, 4–2
ASRT Bridge NETBIOS Filtering feature, prompt, 4–2
ASRT Bridge tunnel feature, prompt, 4–1

B

Bridge
  MAC frame formats, 1–8
  point–to–point links, 1–6
Bridging tunnel
  description of, 3–1
  encapsulation and OSPF, 3–3
Brouter, 2–2

C

Cache
  ASRT Bridge console command, 6–3
  TCP/IP Host Services onsole command, 8–4
Change, ASRT Bridge configuration command, 4–14
Create, NETBIOS filtering configuration command, 4–44

D

Database, permanent, 6–4, 6–14
Delete
  ASRT Bridge configuration command, 4–14
  ASRT Bridge console command, 6–5
  NETBIOS filtering configuration command, 4–44
  TCP/IP Host Services configuration command, 7–3
  Tunnel configuration command, 4–57
Disable
  ASRT Bridge configuration command, 4–17
  NETBIOS filtering configuration command, 4–45
  TCP/IP Host Services configuration command, 7–4
disable, LNM configuration command, 4–41
Dump, TCP/IP Host Services console command, 8–2

E

Enable
  ASRT Bridge configuration command, 4–20
  NETBIOS filtering configuration command, 4–45
  TCP/IP Host Services configuration command, 7–5
enable, LNM configuration command, 4–39
Exit
  ASRT Bridge configuration command, 4–38
  ASRT Bridge console command, 6–25
  LNM configuration command, 4–42
  NETBIOS Filtering configuration command, 4–55
  NETBIOS Filtering console command, 6–28
  TCP/IP Host Services configuration command, 7–7
  TCP/IP Host Services console command, 8–8
  Tunnel configuration command, 4–59

F

Filter–on, NETBIOS filtering configuration command, 4–46
Flip, ASRT Bridge console command, 6–5

Functional address to group address mapping, ASRT Bridge configuration command, 4–9

I

IP tunnel configuration commands, summary of, 4–55
IP tunnel feature and ASRT bridge, description of, 4–1

L

List
  ASRT Bridge configuration command, 4–24
  ASRT Bridge console command, 6–6
  NETBIOS Filtering configuration command, 4–47
  NETBIOS Filtering console command, 6–26
  TCP/IP Host Services configuration command, 7–5
  Tunnel configuration command, 4–58
list, LNM configuration command, 4–42
LLC2 support, 3–11
LNM, accessing the configuration prompt, 4–2
LNM configuration, basic, 5–14
LNM (LAN Network Manager), ASRT Bridge configuration command, 4–33
LNM configuration commands
  ?(Help), 4–39
  disable, 4–41
  enable, 4–39
  exit, 4–42
  list, 4–42
  set, 4–42
  summary of, 4–38

M

MAC addresses, 4–34
MAC frames
  CSMA/CD, 1–9
  token–ring, 1–10
Multiple spanning trees, 3–9
  problems, 3–9

N

NETBIOS Filtering
  and ASRT bridge, 4–1
  basic configuration procedures, 5–6
  building a filter, 3–7
  concepts, 3–4
  simple and complex filters, 3–8
  using bytes, 3–6
  using host names, 3–5
NETBIOS filtering, prompt, 4–2
NETBIOS filtering configuration commands
  ?(Help), 4–43
  create, 4–44
  delete, 4–44
  disable, 4–45
  enable, 4–45
  exit, 4–55
  filter–on, 4–46
  list, 4–47
  summary of, 4–43
  update, 4–48
NETBIOS filtering console commands
  ?(Help), 6–25
  exit, 6–28
  list, 6–26

summary of, 6–25
Netbios–filter, ASRT Bridge console command, 6–24
Netbios–filtering, ASRT Bridge configuration command, 4–33

P

Ping, TCP/IP Host Services console command, 8–5
Port map, 4–33, 6–4
Port maps, ASRT Bridge configuration command, 4–6
Protocol filters
  Ethernet Type, 4–11, 4–16
  SNAP packets, 4–11, 4–16
Protocols
  Adaptive Source Routing Transparent Bridge (ASRT), 4–1, 5–1, 6–1
  TCP/IP Host Services, 7–1, 8–1

R

Routers, TCP/IP Host Services console command, 8–7

S

Set
  ASRT Bridge configuration command, 4–33
  TCP/IP Host Services configuration command, 7–6
set, LNM configuration command, 4–42
Source Routing, threading, 3–12
Source routing, terminology and concepts
  all routes broadcast, 2–37
  all stations broadcast, 2–37
  bridge, 2–37
  bridge number, 2–37
  explorer frames, 2–37
  ring number, 2–37
  route, 2–38
  route designator, 2–38
  route discovery, 2–38
  segment number, 2–38
  single route broadcasting, 2–38
  source routing bridging, 2–38
  spanning tree, 2–38
  transparent bridging, 2–38
Source Routing Bridge
  description of, 2–13
  frame types, 2–15, 2–19
  operation of, 2–14
  Routing Information Field, 2–17
  spanning tree explorer frame, 2–18
  terminology and concepts, 2–37
    bridge instance, 2–22
    bridge number, 2–23
    explorer frames, 2–23
    interface number, 2–23
    route, 2–24
    route discovery, 2–24
    segment number, 2–24
    source routing, 2–24
Source Routing Transparent Bridge
  architecture, 2–26
  description of, 2–24
  operation of, 2–26
  terminology, 2–27
    explorer frames, 2–27
    routing information field (RIF), 2–27

routing information indicator (RII), 2–27
source routing, 2–27
spanning tree, 2–28
transparent bridging, 2–28
Spanning Tree Bridge, 2–2, 2–7
explore option, 2–19
Spanning tree network
balancing traffic loads, 2–20
simulation of, 2–20
Spanning Tree protocol, with 8209 bridges, 3–10

T

TCP/IP Host Services
basic configuration procedures, 7–1
configuring, 7–1
monitoring, 8–1
TCP/IP Host Services configuration commands
?(Help), 7–3
add, 7–3
delete, 7–3
disable, 7–4
enable, 7–5
exit, 7–7
list, 7–5
set, 7–6
summary of, 7–2
TCP/IP Host Services console commands
?(Help), 8–2
dump, 8–2
exit, 8–8
interface, 8–4
ping, 8–5
routers, 8–7
summary of, 8–1
traceroute, 8–6
Threading
by AppleTalk end stations, 3–14
by DNA end–stations, 3–13
by IP end–stations, 3–12
by IPX end stations, 3–14
Traceroute, TCP/IP Host Services console command, 8–6
Transparent Bridge (STB)
bridge ID, 2–3
description of, 2–1
Ethernet packet format translation, 2–7
network requirements, 2–2
operation of, 2–3
port ID, 2–3
root bridge ID, 2–3
routers and bridges, 2–2
shaping the spanning tree, 2–4
spanning tree bridges, 2–7
terminology and concepts, 2–8
aging time, 2–8
bridge, 2–8
bridge address, 2–8
bridge hello time, 2–8
bridge identifier, 2–9
bridge maximum age, 2–9
bridge priority, 2–9
designated bridge, 2–10
designated port, 2–10
filtering and permanent databases, 2–10
parallel bridges, 2–11
path cost, 2–11
port, 2–11
port ID, 2–12
port number, 2–12

port priority, 2–12
resolution, 2–12
root bridge, 2–12
root port, 2–12
spanning tree, 2–12
transparent bridging, 2–13
Tunnel, ASRT Bridge configuration command, 4–38
Tunnel configuration commands
?(Help), 4–57
add, 4–57
delete, 4–57
exit, 4–59
list, 4–58
Tunnel feature, prompt, 4–1
Tunneling, bridge tunnel, 2–14

U

Update, NETBIOS filtering configuration command, 4–48

HOW TO ORDER ADDITIONAL DOCUMENTATION

DIRECT TELEPHONE ORDERS

In Continental USA
call 800–DIGITAL

In Canada
call 800–267–6215

In New Hampshire
Alaska or Hawaii
call 603–884–6660

In Puerto Rico
call 809–754–7575 x2012

ELECTRONIC ORDERS (U.S. ONLY)

Dial 800–234–1998 with any VT100 or VT200
compatible terminal and a 1200 baud modem.
If you need assistance, call 1–800–DIGITAL.

DIRECT MAIL ORDERS (U.S. AND PUERTO RICO*)

U. S. SOFTWARE SUPPLY BUSINESS
DIGITAL EQUIPMENT CORPORATION
10 Cotton Road
Nashua, New Hampshire 03063–1260

DIRECT MAIL ORDERS (Canada)

DIGITAL EQUIPMENT OF CANADA LTD.
940 Belfast Road
Ottawa, Ontario, Canada K1G 4C2
Attn: A&SG Business Manager

INTERNATIONAL

DIGITAL
EQUIPMENT CORPORATION
A&SG Business Manager
c/o Digital's local subsidiary
or approved distributor

Internal orders should be placed through the Software Services Business (SSB)
Digital Equipment Corporation, Westminster, Massachusetts 01473

*Any prepaid order from Puerto Rico must be placed
with the Local Digital Subsidiary:
809–754–7575 x2012

Distributed Routing Software
Bridging Configuration Guide
AA–QL29B–TE

READER'S COMMENTS

What do you think of this manual? Your comments and suggestions will help us to improve the quality and usefulness of our publications.

Please rate this manual:

	Poor				Excellent
Accuracy	1	2	3	4	5
Readability	1	2	3	4	5
Examples	1	2	3	4	5
Organization	1	2	3	4	5
Completeness	1	2	3	4	5

Did you find errors in this manual? If so, please specify the error(s) and page number(s).

General comments:

Suggestions for improvement:

Name ______________________ Date ______________

Title ______________________ Department ______________

Company ______________________ Street ______________

City ______________ State/Country ______________ Zip Code ________

DO NOT CUT – FOLD HERE AND TAPE

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY LABEL
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

digital™

Shared Engineering Services
550 King Street
Littleton, MA 01460–1289

DO NOT CUT – FOLD HERE