CHAPTER 13

Configuring Banyan VINES

The Banyan VINES protocol is a networking system for personal computers. “VINES” is an acronym for Virtual Network System. This proprietary protocol was developed by Banyan Systems, Inc. and is derived from Xerox’s XNS protocol. Our implementation of VINES has been designed in conjunction with Banyan.

This chapter describes how to configure VINES and provides configuration examples. For a complete description of the commands mentioned in this chapter, refer to the ”Banyan VINES Commands“ chapter of the *Router Products Command Reference* publication. For historical background and a technical overview of VINES, see the *Internetworking Technology Overview* publication.

Cisco’s Implementation of VINES

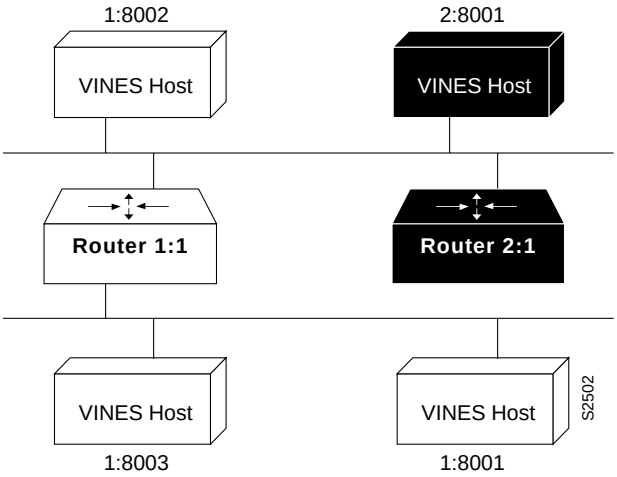
Cisco’s implementation of Banyan VINES provides routing of VINES packets on all media types. Although the software automatically determines a metric value that it uses for routing updates based on the delay set for the interface, this software implementation allows you to customize the metric. Cisco’s implementation also offers address resolution to respond to address requests and broadcast address propagation. MAC-level echo support is also available for Ethernet, IEEE 802.2, Token Ring, and FDDI media. Name-to-address mapping for VINES host names also is supported, as are access lists to filter packets to or from a specific network.

VINES Addresses

VINES network-layer addresses are 48-bit addresses that consist of a network number (better described as a server number) and a subnetwork number (better described as a host number). In this manual, VINES addresses are expressed in the format *network:host*.

The network number identifies a VINES logical network, which consists of a single server and a group of client nodes. The network number is 32 bits (4 bytes) long and is the serial number of the service node. Figure 1-1 shows two logical networks, network 1, which is shaded, and network 2, which is black.

Figure 1-1 VINES Logical Network



The subnetwork number is 16 bits (2 bytes) long. For service nodes, this the subnetwork number is always 1. For client nodes, it can have a value from 0x8001 through 0xFFFE.

The following is an example of a VINES network address:

```
3000577A:0001
```

Here, the server number, or more specifically, the serial number of the service node, is 3000577A and the host number is 0001, indicating that this is a service node. Both portions of the address are expressed in hexadecimal.

VINES Configuration Task List

To configure VINES routing, complete one or more of the following tasks. At a minimum, you must complete the tasks described in the section “Configure VINES Routing.”

- Configure VINES routing (page 13-2)
- Control access to the VINES network (page 13-4)
- Configure other VINES network parameters (page 13-6)
- Configure VINES over WANs (page 13-10)
- Monitor and maintain the VINES network (page 13-11)

This chapter describes how to perform these tasks. Configuration examples are provided at the end of the chapter.

Configure VINES Routing

To configure VINES routing, first enable it on the router. Then enable it on each interface. These are the only two tasks you must perform.

If you are configuring a VINES serverless network, you also must configure the router to respond to ARP address requests. You probably also will want to configure it for serverless support.

Enable VINES Routing on the Router

To enable VINES routing on the router, perform the following task in global configuration mode:

Task	Command
Enable VINES routing on the router.	**vines routing** [*address*]

For an example of how to enable VINES routing, see the section "Typical VINES Network Configuration Example" later in this chapter.

If a router contains only Token Ring interfaces (or Token Ring and serial interfaces), either the Token Ring interface must be fully initialized before you enable VINES routing on the router, or you must specify an address in the **vines routing** command.

Enable VINES Routing on an Interface

After you have enabled VINES on the router, enable it on each interface that will handle VINES traffic. When you enable VINES processing on a specified interface, you can optionally set the metric for that interface. The metric sets the distance to another router or client accessible through that interface. The routing table uses metrics to determine which interface provides the best routing path. If you do not specify a metric, the system automatically chooses a reasonable value that is based on the interface type. The metrics are chosen to match as closely as possible the numbers that a Banyan server would choose for the same type and speed of interface.

To enable VINES routing on an interface other than a serial interface, perform the following task in interface configuration mode:

Task	Command
Enable VINES routing on an interface.	**vines metric** [*number*]

To enable VINES routing on a serial interface, perform the following tasks in interface configuration mode:

Task		Command
Step 1	Determine the bandwidth of the interface.	**show interface**
Step 2	Enable VINES routing, explicitly setting the metric.	**vines metric** *number*

For an example of how to enable VINES routing, see the section "Typical VINES Network Configuration Example" later in this chapter.

For a list of metric values, refer to the discussion of the **vines metric** command in the "Banyan VINES Commands" chapter of the *Router Products Command Reference* manual.

Enable VINES on Serverless Networks

When you have a serverless network configuration, such as separate networks of clients and servers connected by routers, you must perform two additional steps when configuring VINES routing:

- Configure the router to respond to ARP address requests
- Announce that the network is serverless

On the serverless Banyan VINES network, you must configure the router to provide special processing for certain broadcast packets and certain packets directed at the router. This is necessary for proper functioning of the clients on the serverless network: It allows a client to find the services that are provided by a server on another network. Configuring this special processing is especially important when two networks, one with a server and one without a server, are connected to the same router.

Client systems on VINES networks are assigned network addresses dynamically. When VINES clients boot, they have no knowledge of their addresses and preferred servers. Immediately after it initializes its hardware interface, the client sends a broadcast request asking a server to provide it with a network-layer address. Our routers normally do not respond to these broadcast requests. However, on a network that has only clients and no servers (a serverless network), the router needs to respond to the broadcast requests so that all the clients on that serverless network can acquire network addresses. You do this by configuring the router to provide address resolution (ARP) functionality. When ARP is enabled, the router responds to address requests and assigns addresses to network clients. The router then acts as a network communication service provider for the client. The router generates a unique network number for the client based on its own VINES address. A VINES file server must still be present somewhere on the network in order for the client to connect to all other network services.

To enable VINES on serverless networks, perform the following tasks in interface configuration mode *in addition to* the tasks for enabling VINES routing on the network and on specific interfaces.

Task		Command
Step 1	Respond to ARP address requests and assign network addresses to clients.	**vines arp-enable**
Step 2	Announce that the VINES network has no server.	**vines serverless**

For an example of how to enable VINES routing on a serverless network, see the section “Serverless Network Configuration Example” later in this chapter.

Control Access to the VINES Network

To control access to VINES networks, you create access lists and then apply them to filters on individual interfaces. An *access list* is a list of VINES network numbers that is maintained by the router. The list controls access to or from a particular interface. Access lists are useful for providing network security.

There are two types of VINES access lists that you can use to filter routed traffic:

- Standard access list—Restricts traffic based on the protocol, source address and mask, and destination address and mask. You can further restrict traffic by specifying a source and a destination port. Standard VINES access lists have numbers from 1 to 100.
- Extended access list—Restricts traffic in the same way as the standard access list, except that you can also specify masks for the source and destination ports. Extended VINES access lists have numbers from 101 to 200.

VINES has a third type of access list, called a simple access list, that restricts traffic based on source address and source address mask. This type of access list is used to decide which stations to accept time updates from, not to filter traffic.

There are two types of filters you can define on VINES networks:

- Filters on a packet's protocol, source and destination addresses, address masks, and explicit port numbers
- Filters on a packet's protocol, source and destination addresses, address masks, port numbers, and port masks

You perform the following tasks to control access to VINES networks. These tasks are described in the sections that follow.

- Create access lists
- Apply one access list to each interface on which you want to restrict traffic

Keep the following points in mind when configuring VINES network access control:

- You can assign only one access list to an interface.
- The conditions in the access list are applied to all outgoing packets not sourced by the router.
- Access lists entries are scanned in the order you enter them. The first matching entry is used.
- An implicit *deny everything* entry is defined at the end of an access list unless you include an explicit *permit everything* entry at the end of the list.
- All new entries to an existing list are placed at the end of the list. You cannot add an entry to the middle of a list. This means that if you have previously included an explicit *permit everything* entry, new entries will never be scanned. The solution is to delete the access list and retype it with the new entries.

To control access to VINES network, perform the following tasks:

Step 1 Create an access list.

Step 2 Apply an access list to an interface.

To create a VINES access list, perform one or more of the following tasks in global configuration mode:

Task	Command
Create a standard access list.	**vines access-list** *access-list-number* {**deny** | **permit**} *protocol source-address source-address-mask* [*source-port*] *destination-address destination-address-mask* [*destination-port*]
Create an extended access list.	**vines access-list** *access-list-number* {**deny** | **permit**} *protocol source-address source-address-mask* [*source-port source-port-mask*] *destination-address destination-address-mask* [*destination-port destination-port-mask*]
Create a simple access list.	**vines access-list** *access-list-number* {**deny** | **permit**} *source-address source-mask*

Standard access lists have numbers from 1 to 100. Extended access lists have numbers from 101 to 199. Simple access lists have numbers from 201 to 300.

To apply an access list to an interface, perform the following task in interface configuration mode. Remember that you can apply only one access list to each interface.

Task	Command
Apply a VINES access list to an interface.	**vines access-group** *access-list-number*

The access list number must be a number from 1 to 200.

For an example of how to create a VINES access list, see the section “Access List Example” later in this chapter.

Configure Other VINES Network Parameters

To configure other VINES network parameters, perform one or more of the following tasks:

- Select an encapsulation type
- Control the display of host addresses
- Control the radix of host addresses
- Control routing updates
- Disable fast switching
- Set the time
- Configure static routes
- Configure static paths
- Control the forwarding of broadcast packets

Select an Encapsulation Type

You can choose a MAC-level encapsulation type for each Ethernet, Token Ring, and IEEE 802.2 interface. This controls the type of encapsulation used by the router when sending broadcast packets.

To select an encapsulation type, perform the following task in interface configuration mode:

Task	Command
Set the MAC-level encapsulation type.	**vines encapsulation** [**arpa** | **snap** | **vines-tr**]

Note You should not use the **vines encapsulation** command with the current versions of VINES software. This command is provided for future interoperability when Banyan begins using encapsulation types other than the current default ones.

Control the Display of Host Addresses

By default, you enter VINES addresses as numerical values. Also, addresses are displayed numerically in the output of the **show**, **ping**, and **trace** commands. You can assign a host name to each VINES address. Names are easier to remember and type. Assigning a host name allows you to enter the name instead of the address, and it means that the name instead of the numeric address is displayed in output.

To assign a host name to a VINES network address, perform the following task in global configuration mode:

Task	Command
Assign a host name to an address.	**vines host** *name address*

Control the Base of Host Addresses

By default, VINES addresses are represented as hexadecimal numbers. This applies to both the input of addresses and the representation of addresses in output from the router. You can configure the router to display addresses in decimal for consistency with Banyan network management displays.

Names are always preferred when printing addresses. If a name is not available, the address will be printed as a number in the base specified.

To display VINES addresses as decimal numbers, perform the following task in global configuration mode:

Task	Command
Interpret VINES addresses in decimal.	**vines decimal**

Control Routing Updates

There are several tasks you can perform to control the routing updates sent by the router.

The **vines update interval** command controls the interval at which the router sends routing updates. The default interval is 90 seconds. The routing update interval should be the same on all VINES-speaking entities on the same physical network.

The **vines update deltas** command modifies the way that routing information is propagated across the network. On LAN media, using this command causes the router to stop transmitting and to stop expecting periodic full routing updates. Instead, the router transmits and expects a periodic empty routing update, also known as a hello message. On WAN media, using this command causes the router to transmit three normally spaced full routing updates and then cease transmission. The router does *not* send periodic hello messages.

To control routing update frequency and propagation, perform one or both of the following tasks in global configuration mode:

Task	Command
Change the frequency of sending routing updates.	**vines update interval** [*time*]
Change how routing information is propagated.	**vines update deltas**

The VINES Routing Update Protocol (RTP) sends several types of messages, including *redirect* messages If the router detects that a suboptimal path between two nodes is being used, it sends redirect messages to the nodes to indicate the better path. To control the frequency of redirect messages on a specified interface, perform the following task in interface configuration mode:

Task	Command
Set the frequency of RTP redirect messages.	**vines redirect** [*time*]

Normally, the router sends routing updates that list only routes that it learned via other interfaces. This eliminates information that is normally redundant and will be ignored by all routers receiving the update. When split horizon is disabled, routing updates sent out on a given interface will include all routers known by the router. This is useful on X.25 and Frame Relay networks on which there is not a full-mesh topology.

To disable split horizon, perform the following task in interface configuration mode:

Task	Command
Disable split horizon when sending routing updates.	**no vines split-horizon**

Disable Fast Switching

Fast switching allows higher throughput by switching packets using a cache created by previous packets. Fast switching also provides load sharing on a per-packet basis. Fast switching is enabled by default on all interfaces on which it is supported. Fast switching is not supported on older Cisco interface cards, including the CSC-E, CSC-R, CSC-S, and CSC-T cards. Fast switching also is not supported on serial interfaces using encapsulations other than HDLC.

Packet transfer performance is generally better when fast switching is enabled. However, you may want to disable fast switching in order to save memory space on interface cards and to help avoid congestion when high-bandwidth interfaces are writing large amounts of information to low-bandwidth interfaces.

To disable fast switching on an interface, perform the following task in interface configuration mode:

Task	Command
Disable fast switching.	**no vines route-cache**

Set the Time

Banyan VINES servers synchronize time across the entire network by sending zero-hop and two-hop broadcast messages. Previously, in a network where there were three or more routers between servers, time would never get synchronized between the servers. This was because the time synchronization messages could not reach all the routers. The router software now can process and generate time-synchronization messages. It also can retrieve the local time and place it into the VINES time system (which is most useful when running NTP locally) and can use the VINES time system to set a local clock. It also is possible to provide the router with a list of up to 20 destinations for time messages. If you enter specific destination addresses, the router will no longer send broadcast time messages.

To set the VINES network time, perform one or more of the following tasks in interface configuration mode:

Task	Command
Enable the sending of time messages by the local router.	**vines time participate**
Periodically synchronize the router's time with the VINES network time.	**vines time set-system**
Periodically synchronize the VINES network time with the router's time.	**vines time use-system**
Accept time updates from the stations permitted by the specified simple access list.	**vines time access-group** *access-list-number*
Send time updates only to the specified station.	**vines time destination** *address*

The access list number must be from 201 to 300.

For an example of how to set VINES time, see the section "Time-of-Day Service Example" later in this chapter.

Configure Static Routes

VINES uses the Routing Information Protocol (RTP) to determine the best path when several paths to a destination exist. RTP then dynamically updates the routing table. However, you may want to add static routes to the routing table to explicitly specify paths to certain destinations. The decision to use a static route or a dynamic route is always determined by the relative metric numbers.

Be careful when assigning static routes. If a static route is assigned with a better metric than the dynamic routes and the links associated with the static routes are lost, traffic may stop being forwarded, even though an alternative route might be available.

To add a static route to the routing table, perform the following task in global configuration mode:

Task	Command
Add a static route to the routing table.	**vines route** *number address metric*

Configure Static Paths

You can specify static paths to neighbor stations on the network. This is useful for testing VINES networks with test equipment that does not generate hello packets.

To add a static path to a neighbor station, perform the following task in interface configuration mode:

Task	Command
Add a static path to the neighbor station.	**vines neighbor** *address mac-address encapsulation* [*metric*]

Control the Forwarding of Broadcast Packets

Normally, the router decides whether to forward a broadcast packet on an interface based on the settings of both the “hop count” and “class” fields of the VINES IP header. You can have the router ignore the “class” field and make the routing decision solely based on hop count. This is useful on a serverless network that is connected via a serial line.

To have the router modify how it forwards broadcast packets, perform the following task in interface configuration mode:

Task	Command
Have the router ignore the “class” field when forwarding broadcast packets.	**vines propagate**

Configure VINES over WANs

You can configure VINES over X.25, Frame Relay, and SMDS networks. To do this, configure the appropriate address mappings as described in the chapters ”X.25 and LAPB Commands,“ ”Frame Relay Commands,“ and ”SMDS Commands,“ respectively.

Monitor and Maintain the VINES Network

To monitor a VINES network, perform one or more of the following tasks at the EXEC prompt:

Task	Command
Delete entries from the VINES fast-switching cache table.	**clear vines cache** [**interface** *interface* | **neighbor** *address* | **server** *number*]
Delete entries from the neighbor table.	**clear vines neighbor** {*address* | ***}**
Delete network addresses from the routing table.	**clear vines route** {*address* | ***}**
Zero the VINES-related traffic statistics displayed by the **show vines traffic** command.	**clear vines traffic**
Send datagrams to a host to determine network connectivity.	**ping** [**vines** *hostname* | [**vines**] *address*]
Display the VINES access lists currently defined.	**show vines access**
Display the contents of the VINES fast-switching cache table.	**show vines cache** [*address* | **interface** *interface* | **neighbor** *address* | **server** *number*]
Display the entries in the VINES host name table.	**show vines host** [*name*]
Display VINES-related interface settings.	**show vines interface** [*interface unit*]
Display information about any currently active IPC connections.	**show vines ipc**
Display the contents of the VINES neighbor table.	**show vines neighbors** [*address* | **interface** *interface* |**server** *number*]
Display the contents of the VINES routing table.	**show vines route** [*number* | **neighbor** *address*]
Display information about the router's current time.	**show vines services**
Display the statistics about VINES protocol traffic.	**show vines traffic** [*interface unit*]
Determine the path a packet takes when traversing a VINES network.	**trace**

If you find that two routers have the same VINES network addresses, you can have the routers dynamically recompute their addresses. To do this, perform the following task in global configuration mode on each of the two routers:

Task	Command
Dynamically redetermine the router's address.	**vines routing recompute**

VINES Configuration Examples

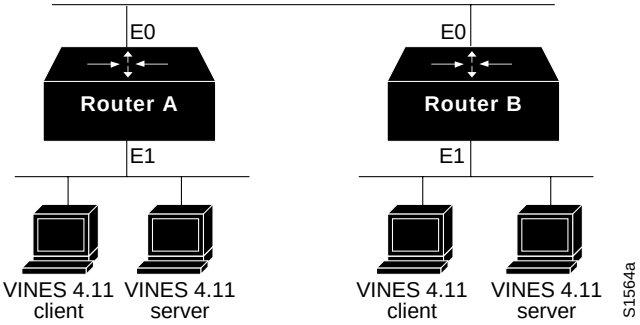
Use the following configuration examples to help in configuring VINES routing on your network:

- Typical VINES network configuration example (page 13-12)
- Serverless network configuration example (page 13-12)
- Access list example (page 13-12)
- Time-of-day service example (page 13-17)

Typical VINES Network Configuration Example

Figure 1-2 illustrates how to configure a simple VINES network.

Figure 1-2 VINES Simple Configuration



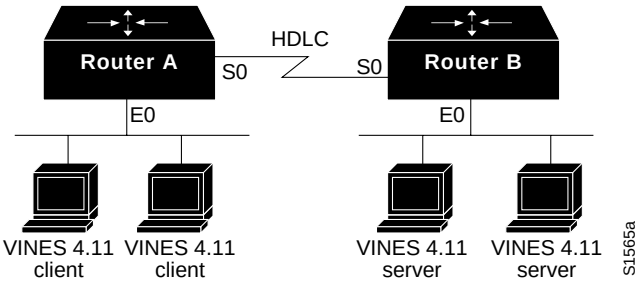
The following is an example configuration for Routers A and B:

```
!
vines routing
!
interface ethernet 0
vines metric 2
!
interface ethernet 1
vines metric 2
!
```

Serverless Network Configuration Example

The following examples illustrate how to configure VINES routing for various network topologies that include serverless networks. The first example illustrates how to configure a simple serverless network (see Figure 1-3).

Figure 1-3 VINES Serverless Configuration



Configuration for Router A

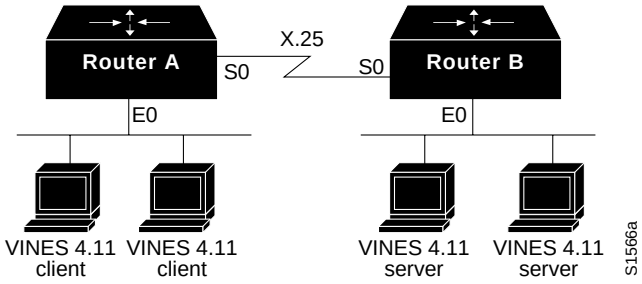
```
!
vines routing
!
interface ethernet 0
vines metric 2
vines arp-enable
vines serverless
!
interface serial 0
vines metric 45
!
```

Configuration for Router B

```
!
vines routing
!
interface ethernet 0
vines metric 2
!
interface serial 0
vines metric 45
vines serverless
!
```

The following example (see Figure 1-4) has an X.25 interface instead of an HDLC serial line, and it also has multiple versions of VINES software running at the same time. The configuration differences to support this are very minor.

Figure 1-4 VINES Serverless X.25 Configuration



Configuration for Router A

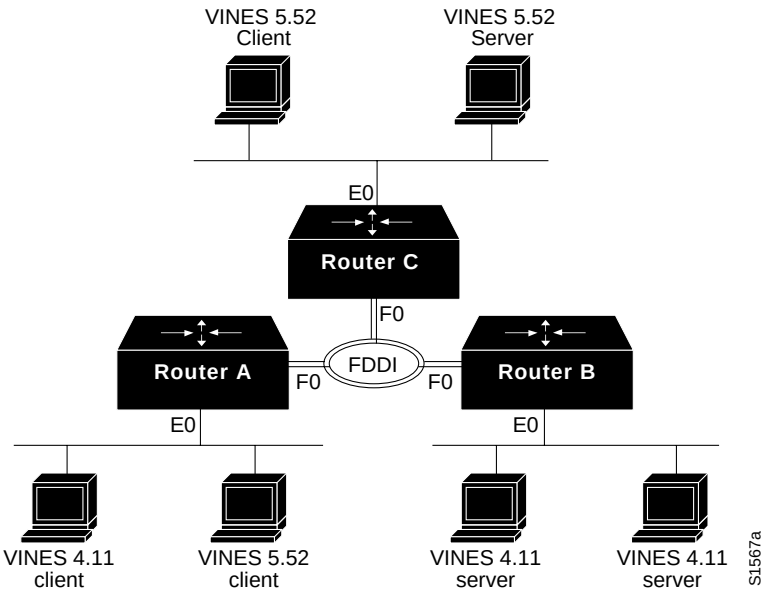
```
!
vines routing
!
interface ethernet 0
vines metric 2
vines arp-enable
vines serverless
!
interface serial 0
vines metric 55
vines propagate
!
```

Configuration for Router B

```
!
vines routing
!
interface ethernet 0
vines metric 2
!
interface serial 0
vines metric 55
vines serverless
!
```

The following example has an FDDI interface instead of a serial line (see Figure 1-5). It also has the servers for the different VINES versions on different physical networks and has a requirement that the clients to be able to run any VINES version. The best way to configure this topology would be the following configuration.

Figure 1-5 VINES Complex Serverless Configuration



Configuration for Router A

```
!
vines routing
!
interface ethernet 0
vines metric 2
vines arp-enable
vines serverless broadcast
!
interface fddi 0
vines metric 1
!
```

Configuration for Routers B and C

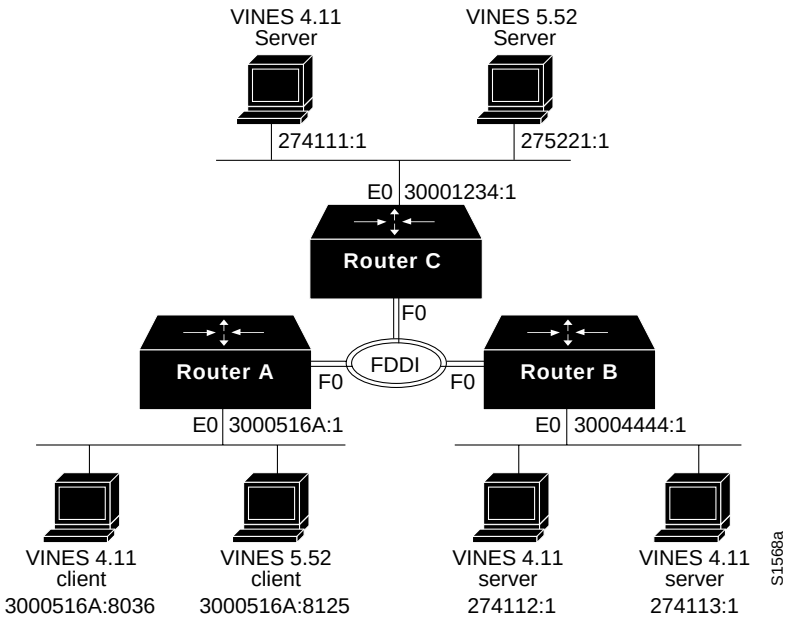
```
!
vines routing
!
interface ethernet 0
vines metric 2
!
interface fddi 0
vines metric 1
vines serverless
!
```

The **broadcast** keyword on the **vines serverless** command on server A causes it to forward packets onto the FDDI ring as broadcasts instead of sending them to either Router B or Router C. This allows the **vines serverless** commands on both routers to forward the frame from the FDDI ring to the Ethernet network.

Access List Example

The following example illustrates how to configure an access list that filters all packets between two VINES servers (see Figure 1-6). For this example, the servers in the upper left and lower right corners are configured.

Figure 1-6 VINES Access List Configuration



On Router B, you would set up the following configuration:

```
vines routing
vines access-list 1 deny IP 274113:1 0:0 274111:1 0:0
vines access-list 1 permit IP 0:0 FFFFFFFF:FFFF 0:0 FFFFFFFF:FFFF
!
interface ethernet 0
vines metric 2
vines access-group 1
!
interface fddi 0
vines metric 1
!
```

The first line in the access list prohibits any communication between the two servers, while the second line allows all other communication to pass through the router.

If you wanted to allow only mail traffic between these two servers, you would need the following configuration. Port 4 is the VINES Mail port.

```
vines routing
vines access-list 101 permit IPC 274113:1 0:0 0 FFFF 274111:1 0:0 4 0
vines access-list 101 permit IPC 274111:1 0:0 4 0 274113:1 0:0 0 FFFF
vines access-list 101 deny IP 274111:1 0:0 274113:1 0:0
vines access-list 101 permit IP 0:0 FFFFFFFF:FFFF 0:0 FFFFFFFF:FFFF
!
interface ethernet 0
vines metric 2
vines access-group 101
!
interface fddi 0
vines metric 1
!
```

The first line in the access list allows mail messages being sent from the server in the lower right to the server in the upper left. The second line allows mail messages in the other direction. The third line prohibits all other communication between these two servers. The last line allows all other communication to pass through the router.

Time-of-Day Service Example

The following example, using the configuration shown in Figure 1-6, illustrates how to configure the "Time of Day" support in a VINES network. Router C also is configured as a NTP server and will provide time to the VINES network.

Configuration for Routers A and B

```
vines routing
!
interface ethernet 0
vines metric 2
!
interface fddi 0
vines metric 1
!
vines access-list 201 permit 3000516A:1 0:0
vines access-list 201 deny 0:0 FFFFFFFF:FFFF
vines time access-group 201
vines time participate
vines time set-system
!
```

Configuration for Router C

```
vines routing
!
interface ethernet 0
vines metric 2
!
interface fddi 0
vines metric 1
!
ntp peer 128.9.2.129
vines time participate
vines time use-system
!
```

The access list on Routers A and B is not absolutely necessary. It prevents the routers from learning the time from anyone other than Router C. The reason this is not very important is that each time message from Router C will override any time that has been previously learned (because of the **vines time use-system** command).