Chapter 1
Routing VINES

1

This chapter describes the Cisco Systems implementation of the Banyan VINES routing protocol, and provides the following information:

- Cisco's implementation of VINES
- Configuring routing in a VINES network, including special customizing procedures for such tasks as address resolution and filtering packets through the network
- Maintaining and monitoring the VINES network

Cisco's Implementation of VINES

The Banyan VINES protocol is a networking system for personal computers. The word "VINES" stands for Virtual Network System. This proprietary protocol was developed by Banyan, and is derived from Xerox's XNS protocol. The Cisco implementation of VINES has been designed in conjunction with Banyan. It is a separate effort from Cisco's XNS routing protocol support.

Cisco's implementation of VINES provides routing of VINES packets on all media types. Although software automatically determines a metric value that it uses in routing updates based on the delay set for the interface, the Cisco software implementation allows you to customize the metric.

Cisco's VINES software offers address resolution to respond to address requests, and propagation of broadcast addresses. MAC-level encapsulation is also available for Ethernet, IEEE 802.2, and Token Ring media. Name-to-address binding for VINES host names is also supported, as are access lists to filter packets to or from a specific network. As with all Cisco software, the VINES implementation also includes EXEC-level commands for maintaining, monitoring, and debugging the VINES network.

Configuring VINES

There are only two commands required to enable VINES routing:

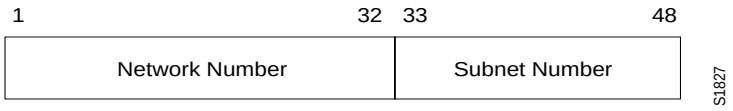
- Use the global configuration command **vines routing** to enable routing.
- Use the interface subcommand **vines metric** to enable VINES processing on the interface.

All other configuration commands provide additional functionality or refinements. Each task is described in the following sections, and are followed by descriptions of the EXEC commands to maintain, monitor and debug the VINES network. Summaries of the global configuration commands and interface subcommands described in this chapter appear at the end of this chapter.

VINES Addressing

All VINES products automatically determine their addresses. The VINES specification guarantees interoperability. The network-level address consists of two numbers: a 32-bit network number, and a 16-bit number that Banyan refers to as a *subnet number* but that actually serves as a host number. Figure 1-1 illustrates the address format.

Figure 1-1 Banyan VINES Network-Level Addresses



Address conflicts are impossible since VINES servers use their Banyan-assigned unique key serial numbers as their network number, and use a subnet number of one. Since the keys are unique, the server addresses are unique. VINES clients do not have addresses, as such. The clients use a modified version of the address of the first file server found on the physical network. The clients assume the servers' network number, and are assigned a subnet number by that server.

Using this address assignment scheme, it is likely that two clients on the same physical LAN will have different addresses. This means that the router must keep a cache of local neighbors as well as of routing entries.

Cisco has been assigned a portion of the overall VINES network number space by Banyan. This portion is the set of all numbers that begin with the first 11 bits (of the 32) of 0110 0000 000. This will appear in all Cisco displays as a hexadecimal number beginning with 0x600 or 0x601. Routers attempt to automatically map themselves into the Cisco number space based upon the first nonzero Ethernet or Token Ring address found. In the unlikely

event that two routers map themselves to the same address, you may use the optional arguments to the **vines routing** command (see the section "Configuring VINES Routing") to override this selection.

Note: Older implementations of the Cisco software mapped themselves to numbers beginning with 0xFC0. This was done before Banyan made the address assignment.

The VINES Routing Table Protocol

Neighboring clients, servers, and routers are found using the Routing Table Protocol (RTP). Every VINES client sends a broadcast HELLO message at periodic intervals. This message indicates that the client is still operating and reachable from the network. For VINES Version 3, this interval is 30 seconds. It has been increased to once every 90 seconds in VINES Version 4 release. VINES servers send both *HELLO* and *update* messages periodically. The *update* message is used to indicate which remote servers and their satellite clients can be reached by routing through that server. These messages are sent by VINES servers every 90 seconds. Cisco routers also broadcast update messages once every 90 seconds; however, they do not send *HELLO* messages, as they are redundant when updates are being sent.

Configuring VINES Routing

Use the global configuration command **vines routing** to enable VINES routing. The command has this syntax:

vines routing [*address*]

no vines routing

Specify the optional *address* argument only when you are not using an Ethernet or Token Ring interface, since the router automatically maps itself into the VINES address space reserved for Cisco routers. When the optional *address* argument is used, the routers' VINES address is automatically configured as the specified address.

Use the **vines metric** interface subcommand to enable VINES processing on the defined interface. The command has this syntax:

vines metric [*number*]

no vines metric

The system automatically chooses a reasonable metric value that it uses in routing updates based upon the delay value set for the interface. These numbers are chosen to match as closely as possible to the numbers that a Banyan server would choose for the same type and speed of interface. Use the optional *number* argument to force the delay metric for the interface to a specific value. Delay metrics are used by Banyan servers to compute time-outs when communicating with other hosts. Be careful, when forcing metrics, that you do not set this number too high or too low and therefore disrupt the normal function of the Banyan servers. This number is generally inversely proportional to the speed of the interface. Some example numbers are:

Ethernet	2
16M TR	2
4M TR	4
56K Serial	45
9600 Serial	90

Example:

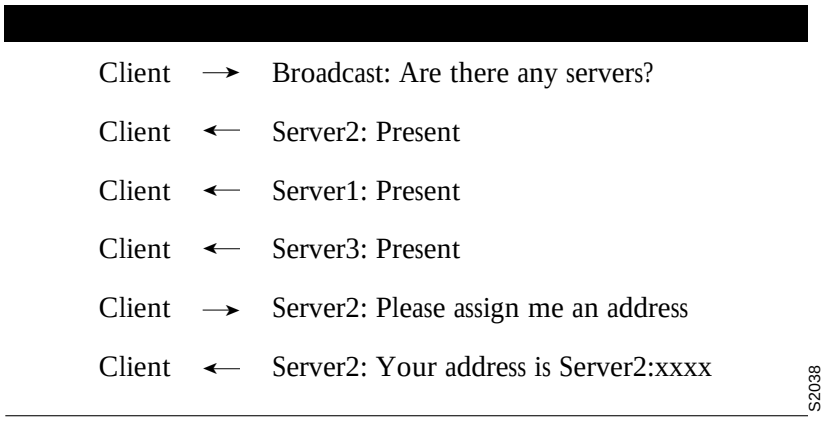
These commands enable VINES routing on the Ethernet 0 interface.

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

Configuring Address Resolution

VINES clients boot without any knowledge of network-level addresses and preferred servers. The first thing that a client does after initializing its hardware interface is send a broadcast message looking for available servers on the network to which it is attached. The client then waits for responses from the network. Once a message is received, the client sends an address assignment request to the first server that replied to the previous message. That server computes a new, unique address based on its own network number, and assigns the address to that client.

The address assignment interaction is accomplished by sending responses directly to the MAC addresses of the client, since it does not have a network-level address yet. The following illustration depicts these events.



Cisco routers normally do not respond to any of these messages. There is only one situation where a Cisco response would be necessary, and this is when a *stub* network (an Ethernet network with only one connection to a router) has only clients and no server. The Cisco router must be explicitly configured to provide address resolution functionality. By turning on the Cisco address resolution feature, the router will begin responding to address requests and assigning addresses. The Cisco router then acts as a network communications service provider for the client. There must still be a VINES file server somewhere on the network for the client to connect to for all other services. Use the command **vines arp-enable** to enable this feature. The command has this syntax:

vines arp-enable

no vines arp-enable

The Cisco router generates a unique network number in a range assigned by Banyan, based on its VINES address.

A Banyan VINES network without a server needs to be configured with the **vines serverless** interface subcommand. This command provides special processing for certain broadcast packets and certain packets directed at the router. It is necessary for proper functioning of the clients on a network without a server. This is especially important when two networks, one with a server and one without a server, are not connected to the same router; in this situation you must use this command to build a path between the two networks. The command has this syntax:

vines serverless

no vines serverless

The default is **no vines serverless**.

Example:

Use the following example commands when interface Ethernet 1 is a stub Ethernet that does *not* contain any VINES servers.

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

See the section "VINES Configuration Examples," later in this chapter, for additional examples of use of this command.

Configuring VINES Encapsulation

Use the interface subcommand **vines encapsulation** to set the MAC-level encapsulation used for VINES broadcasts on the defined interface. The command has this syntax:

vines encapsulation [**arpa**|**snap**|**vines-tr**]

You can choose from these encapsulation types:

- **arpa** for Ethernet lines
- **snap** for IEEE 802.2 media
- **vines-tr** for Token Rings

The default encapsulation for Ethernet interfaces is **arpa**. The default encapsulation for Token Ring interfaces is **vines-tr.** All other media defaults to **snap**. As Banyan migrates to IEEE 802.2 encapsulation in future releases, the default will become **snap** for all media types.

Example:

This example configures the IEEE 802.2 SNAP encapsulation.

```
vines routing
!
interface ethernet 0
vines metric 2
vines encapsulation snap
```

Please note that the **vines encapsulation** command only affects broadcasts from the router. The router keeps track of which encapsulation is used by each of its neighbors, and will use the same style of encapsulation when talking directly to a neighbor.

Configuring Name-to-Address Mappings

Cisco provides only static name-to-address bindings for the VINES protocol. (This is completely separate from Banyan's Streettalk implementation.) Once entered, a VINES host name can be used anywhere that a VINES address can be used. This makes typing commands easier as it is much easier to remember and type the name of a system than it is to remember and type its address. Names are entered by using the **vines host** configuration command. The command has this syntax:

vines host *name address*

no vines host *name*

The argument *name* can be any length and sequence of characters separated by white space. The argument *address* consists of a VINES address in the form of network:subnet.

Example:

These commands define four host names.

```
!
! cisco names
vines host FARSLAYER 60002A2D:0001
vines host DOOMGIVER 60000A83:0001
! VINES PS/2 server
vines host COINSPINNER 0027AF92:0001
! PC clone client
vines host STUFF 0027AF92:8001
!
```

To examine the cache of VINES name-to-address mappings, use the EXEC command **show vines host.**

The output appears as follows.

```
VINES Name Table
            FARSLAYER                    60002A2D:0001
            DOOMGIVER                    60000A83:0001
            COINSPINNER                  0027AF92:0001
            STUFF                        0027AF92:8001
```

See the section "Monitoring the VINES Network" for more information about this command.

Configuring VINES Access Lists

An access list is a list of VINES network numbers kept by the Cisco router to control access to or from a specific network for a number of services.

Cisco's VINES access list provides network security by permitting or denying certain packets onto a network interface.

Use the global configuration command **vines access-list** to create or delete a VINES access list. The command has this syntax:

vines access-list *number* {**permit**|**deny**} **IP** *source-address source-mask dest-address dest-mask*

vines access-list *number* {**permit**|**deny**} *protocol source-address source-mask source-port dest-address dest-mask dest-port*

no vines access-list *number*

The argument *number* must be an integer between 1 and 100 for both variations of the command. The argument *protocol* may be an integer between 1 and 255, or one of these protocols:

- **SPP**—Sequence Packets Protocol
- **RTP**—Routing Table Protocol
- **ARP**—Address Resolution Protocol
- **IPC**—Interprocess Communications
- **ICP**—Internet Control Protocol

Source and destination addresses (*source-address* and *dest-address*) are VINES addresses written in the standard form.

Source and destination masks (*source-mask* and *dest-mask*) are also VINES addresses in the standard form, but they indicate the bits in the corresponding address that should be ignored.

Port numbers (*source-port* and *dest-port*) are integers in the range of 0 to 65,535.

Use the interface subcommand **vines access-group** to assign an outgoing VINES access list to the defined interface. The command has this syntax:

vines access-group *list*

no vines access-group

The argument *list* is the number of an access list defined with the **vines access-list** command.

Examples:

In this example, the first line permits any two servers (subnet of 1) to talk to each other, regardless of their network numbers, and the second line denies everything else.

```
!
vines routing
vines access-list 1 permit IP 0:1 ffffffff:0    0:1 ffffffff:0
vines access-list 1 deny   IP 0:0 ffffffff:ffff 0:0 ffffffff:ffff
!
```

In this example, the first line prohibits any communication on IPC port number 15 (hex 0F); the second line permits all other communication:

```
vines access-list 1 deny   IPC 0:0 ffffffff:ffff 0xf 0:0 ffffffff:ffff 0xf
vines access-list 1 permit IP 0:0 ffffffff:ffff     0:0 ffffffff:ffff
```

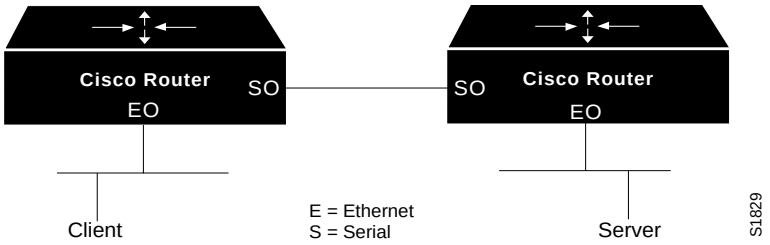
VINES Configuration Examples

Use the following configurations examples to help you configure your VINES routing network.

Propagating Broadcasts

The following examples illustrate how to configure a VINES network where the clients are all behind one router, and the servers are behind another router, as depicted in the following illustration:

Figure 1-2 VINES Client/Server Configuration



Example for Client:

```
!
vines routing
!
interface ethernet 0
vines metric 2
vines serverless
!
interface serial 0
vines metric 15
vines serverless
!
```

Example for Server:

```
!
vines routing
!
interface ethernet 0
vines metric 2
!
interface serial 0
vines metric 15
vines serverless
!
```

Filtering Packets

The following example illustrates how to configure an access list that allows packets across two VINES networks and filters all packets across a third VINES network.

Example:

```
!
vines routing
vines access-list 1 permit IP 60000A83:0 0:FFFF 60003753:0 0:FFFF
vines access-list 1 deny   IP 0:0 FFFFFFFF:FFFF 0:0 FFFFFFFF:FFFF
!
```

Maintaining the VINES Network

Use these EXEC commands to maintain the VINES routing protocol tables.

Use the privileged EXEC command **clear vines neighbor** to delete the specified address from the neighbor table maintained by the router. This also forces the deletion of all routing table entries that have the given neighbor as the first hop in their path.

clear vines neighbor *{address | *}*

The argument *address* is the neighbor address, or may be an asterisk (*), which deletes all entries from the neighbor routing table except for the entry for the router itself.

Example:

This command clears all entries from the neighbor routing table.

```
gateway#clear vines neighbor *
```

Use the privileged EXEC command **clear vines route** to delete the specified network address from the routing table maintained by the router.

clear vines route *{address | *}*

The argument *address* is the routing address, or may be an asterisk (*), which deletes all entries from the routing table except for the entry for the router itself.

Example:

This command clears all entries from the routing table.

```
gateway#clear vines route *
```

Monitoring the VINES Network

Use these EXEC commands to monitor networks configured for VINES routing.

Displaying the VINES Name Table

Use the EXEC command **show vines host** to display the contents of the VINES name table.

show vines host [*name*]

If the optional *name* argument is included, then only that entry is printed from the table. If no name is present, then the entire table is printed.

Displaying VINES Interface Settings

Use the EXEC command **show vines interface** to display all VINES-related interface settings.

show vines interface [*interface unit*]

If no interface name is supplied, then the values for all interfaces are printed, along with the VINES global parameters.

Sample output from this command is shown below.

```
 VINES address is 60000A83:1
Next client will be 60000A83:8000
–More–
Ethernet 0 is up, line protocol is up
 VINES protocol processing disabled
–More–
Serial 0 is up, line protocol is up
 Interface metric is 27 (5.4 seconds)
 Outgoing access list is not set
–More–
Ethernet 1 is up, line protocol is up
 VINES unknown host encapsulation is ARPA
 Interface metric is 2 (0.4 seconds)
 Outgoing access list is not set
 -More–
```

```
Serial 1 is up, line protocol is up
 VINES protocol processing disabled
```

Displaying the VINES Neighbor Table

Use the EXEC command **show vines neighbors** to display the entire contents of the VINES neighbor table maintained by this router.

show vines neighbors [*address*]

If the optional *address* argument is supplied, then only that entry from the table is displayed.

Sample output from this command is shown below.

```
Network   Subnet Age   Metric Hardware Address Type      Interface
0027AF92   0001    16       2    0260.8C3C.141C   ARPA      Ethernet0
60000A83   0001    12       2    0000.0C00.0A84   ARPA      Ethernet1
0027AF92   8001    12       2    0000.C05A.0511   ARPA      Ethernet0
60002A2D  0001     -      0    -               -        -
```

Displaying the VINES Routing Table

Use the EXEC command **show vines route** to display the entire contents of the VINES routing table.

show vines route [*address*]

If the optional *address* argument is supplied, then only the routing entry for that network is displayed.

Sample output of this command is shown below.

```
Codes: R - RTP derived, C - connected, S - static, 4 routes
R Net 0027AF92 [2] via 0027AF92:1, 21 sec, 0 uses, Ethernet0
R Net 60000A83 [2.H] via 60000A83:1, 503 sec, 0 uses, Ethernet1
C Net 60002A2D is this router's network, 0 uses
```

A .H after the metric indicates that the route is currently in a hold down state.

Displaying VINES Traffic

Use the EXEC command **show vines traffic** to display the statistics kept on VINES protocol traffic.

show vines traffic

Sample output of this command is shown below.

```
     Rcvd: 4218 total, 0 format errors, 0 checksum errors, 12 bad hop count,
           3994 local destination
   Bcast: 3994 received, 2 sent, 1 forwarded
          0 not lan, 0 not gt 4800, 0 no charge
    Sent: 3512 generated, 147 forwarded
          0 encapsulation failed, 65 no route
     IPC: 0 errors received, 12 errors sent, 0 metric sent
    Echo: Received 0, Sent 2
Unknown: 0 packets
```

VINES Ping Command

The EXEC command **ping** allows the network administrator to determine network connectivity by sending datagrams to the host in question. A sample session is shown below.

```
Doomgiver# ping coinspinner
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Verbose [n]:
Type escape sequence to abort.
Sending 5, 100-byte VINES Echos to 27AF92:1,
timeout is 2 seconds:
!!!!!
Success rate is 100 percent, round-trip min/avg/max = 4/7/8 ms
```

VINES Trace Command

The EXEC command **trace** allows the administrator to determine the path that a packet takes when traversing a VINES network. This command does not produce the names of any VINES servers that are traversed. Sample output of tracing through Cisco routers to reach a Banyan file server is shown below.

```
Doomgiver#trace coinspinner
Type escape sequence to abort.
Tracing the route to COINSPINNER (27AF92:1)
 0 FARSLAYER (60002A2D:1) 0 msec 4 msec 4 msec
 1 COINSPINNER (27AF92:1) 4 msec 4 msec 8 msec
```

Debugging the VINES Network

Use these **debug** commands to obtain reports of VINES' routing functionality.

The following **debug** commands are disabled by entering the **undebug** version of the command once it is enabled.

debug vines-arp

The EXEC command **debug vines-arp** enables logging of all ARP protocol processing that occurs in the router.

debug vines-echo

The EXEC command **debug vines-echo** enables logging of all MAC-level echo processing that occurs in the router. This echo functionality is used by the Banyan interface testing programs.

debug vines-packet

The EXEC command **debug vines-packet** enables logging of general VINES debugging information. This includes packets received, generated, and forwarded, as well as failed access checks and other items.

debug vines-routing

The EXEC command **debug vines-routing** enables logging of all RTP update messages sent or received, and all routing table activities that occur in the router.

debug vines-table

The EXEC command **debug vines-table** enables logging of all modifications to the VINES routing table. This command provides a subset of the information provided by the **debug vines-routing** command.

VINES Global Configuration Command Summary

Following is an alphabetical list of the VINES global configuration commands that specify system-wide parameters for VINES support.

[**no**] **vines access-list** *number* {**permit**|**deny**} **IP** *source-address source-mask dest-address dest-mask*

[**no**] **vines access-list** *number* {**permit**|**deny**} *protocol source-address source-mask source-port dest-address dest-mask dest-port*

Creates or deletes a VINES access list. The argument *number* must be an integer between 1 and 100 for both variations of the command. The argument *protocol* is an integer between 1 and 255, or one of these protocols.

- **SPP**—Sequence Packets Protocol
- **RTP**—Routing Table Protocol
- **ARP**—Address Resolution Protocol
- **IPC**—Interprocess Communications
- **ICP**—Internet Control Protocol

Source and destination addresses (*source-address* and *dest-address*) are VINES addresses written in the standard form.

Source and destination masks (*source-mask* and *dest-mask*) are also VINES addresses in the standard form, but they indicate the bits in the corresponding address that should be ignored.

Port numbers (*source-port and dest-port*) are integers in the range of 0 to 65,535.

The **no** variation with the access list number deletes the access list.

[**no**] **vines host** *name address*

Adds or deletes an entry to the VINES name-to-address mapping table. Once entered, a VINES name can be used anywhere that a VINES addresses is requested.

[**no**] **vines routing** [*address*]

Enables or disables VINES routing. The argument *address* is not necessary if you have Ethernet interfaces, as the router will automatically map itself into the VINES address space that is reserved for Cisco. If the optional *address* argument is given, then the routers' VINES address will be forced to be the specified address.

VINES Interface Subcommand Summary

Following is an alphabetical list of the VINES interface subcommands that specify parameters for interfaces configured for VINES routing. These commands must follow an **interface** command.

[**no**] **vines access-group** *list*

Assigns or removes an outgoing VINES access list to the defined interface. The argument *list* is the number of an access list defined with the **vines access-list** command.

[**no**] **vines arp-enable**

Enables or disables the processing of ARP packets received on the defined interface. When enabled, the router responds to ARP packets and assigns network addresses to clients.

vines encapsulation [**arpa**|**snap**|**vines-tr**]

Sets the MAC-level encapsulation used for VINES broadcasts on the defined interface. The current defaults are: **arpa** for Ethernet lines, **snap** for IEEE 802.2 media, and **vines-tr** for Token Rings.

[**no**] **vines metric** [*number*]

Enables or disables the processing of VINES processing on this interface. The system automatically chooses a reasonable metric value, which is used in routing updates, and is based upon the delay value set for the defined interface. When the optional *number* argument is supplied, the system forces the VINES delay metric for this interface to the supplied number.

[**no**] **vines serverless**

Enter to configure a Banyan VINES network without a server. This command can be used on several routes to build a path to a network that contains servers. The default is **no vines serverless**.