Introduction

This guide describes the Cisco Systems private, or local, Management Information Base (MIB) for Software Release 9.21. The Cisco MIB is provided with all Cisco software releases and with CiscoWorks router management software. The MIB file contains variables that can be set or read to provide information on network devices and interfaces.

The Cisco MIB is a set of variables that are private extensions to the Internet standard MIB II. The MIB II is documented in RFC 1213, *Management Information Base for Network Management of TCP/IP-based Internets: MIB-II*.

The listings of Cisco MIB variables in the *mib921.txt* file, which can be obtained by FTP from the Cisco server, is identical to the listings of Cisco MIB variables in this guide. Unlike the *mib921.txt* file, however, the Cisco MIB variables are presented alphabetically in this guide for quick reference.

You can obtain the file *mib921.txt* describing the Cisco MIB by using the **ftp ftp.cisco.com** command. Log in with the username **anonymous** and press Return when prompted for the password. Use the **get README** command to display the *readme* file containing a list of available files. To obtain the Software Release 9.21 MIB file *mib921.txt*, use the **get mib921.txt** command.

The Cisco MIB variables are accessible via the Simple Network Management Protocol (SNMP), which is an application-layer protocol designed to facilitate the exchange of management information between network devices. The SNMP system consists of three parts: an SNMP manager, SNMP agent, and Management Information Base (MIB).

Instead of defining a large set of commands, SNMP places all operations in a *get request*, *get-next-request*, and *set request* format. For example, an SNMP manager can get a value from an SNMP agent or store a value into that SNMP agent. The SNMP manager can be part of a Network Management System (NMS), and the SNMP agent can reside on a networking device such as a router. You can compile the Cisco MIB with

your network management software. If SNMP is configured on a router, the SNMP agent can respond to MIB-related queries being sent by the NMS.

An example of an NMS is CiscoWorks, the Cisco network management software. CiscoWorks uses the Cisco MIB variables to set device variables and to poll devices on the internetwork for specific information. The results of a poll can be graphed and analyzed in order to troubleshoot internetwork problems, increase network performance, verify the configuration of devices, monitor traffic loads, and more.

As shown in Figure 1, the SNMP agent gathers data from the MIB, which is the repository for information about device parameters and network data. The agent also can send traps, or notification of certain events, to the manager. The Cisco trap file, *mib.traps921*, which documents the format of the Cisco traps, is available on the Cisco host `ftp.cisco.com`.

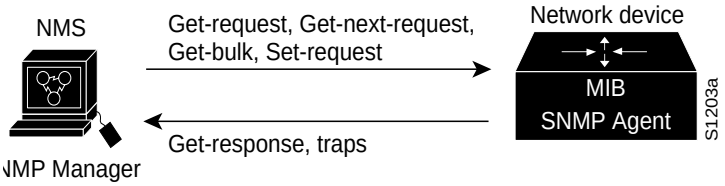

Figure 1 SNMP Network

The SNMP manager uses information in the MIB to perform the operations described in Table 1.

Table 1 SNMP Manager Operations

Operation	Description
get-request	Retrieve a value from a specific variable.
get-next-request	Retrieve a value from a variable within a table.[1]
get-response	The reply to a get-request, get-next-request, and set-request sent by an NMS.
set-request	Store a value in a specific variable.
trap	An unsolicited message sent by an SNMP agent to an SNMP manager indicating that some event has occurred.

1. With this operation, an SNMP manager does not need to know the exact variable name. A sequential search is performed to find the needed variable from within a table.

Internet MIB Hierarchy

The MIB structure is logically represented by a tree hierarchy. (See Figure 2.) The *root* of the tree is unnamed and splits into three main branches: Consultative Committee for International Telegraph and Telephone (CCITT), International Organization for Standardization (ISO), and joint ISO/CCITT.

These branches and those that fall below each category have short text strings and integers to identify them. Text strings describe *object names*, while integers allow computer software to create compact, encoded

representations of the names. For example, the Cisco MIB variable *authAddr* is an object name and is denoted by number 5, which is listed at the end of its object identifier number *1.3.6.1.4.1.9.2.1.5*.

The *object identifier* in the Internet MIB hierarchy is the sequence of numeric labels on the nodes along a path from the root to the object. The Internet standard MIB is represented by the object identifier of 1.3.6.1.2.1. It also can be expressed as *iso.org.dod.internet.mgmt.mib* (See Figure 2.)

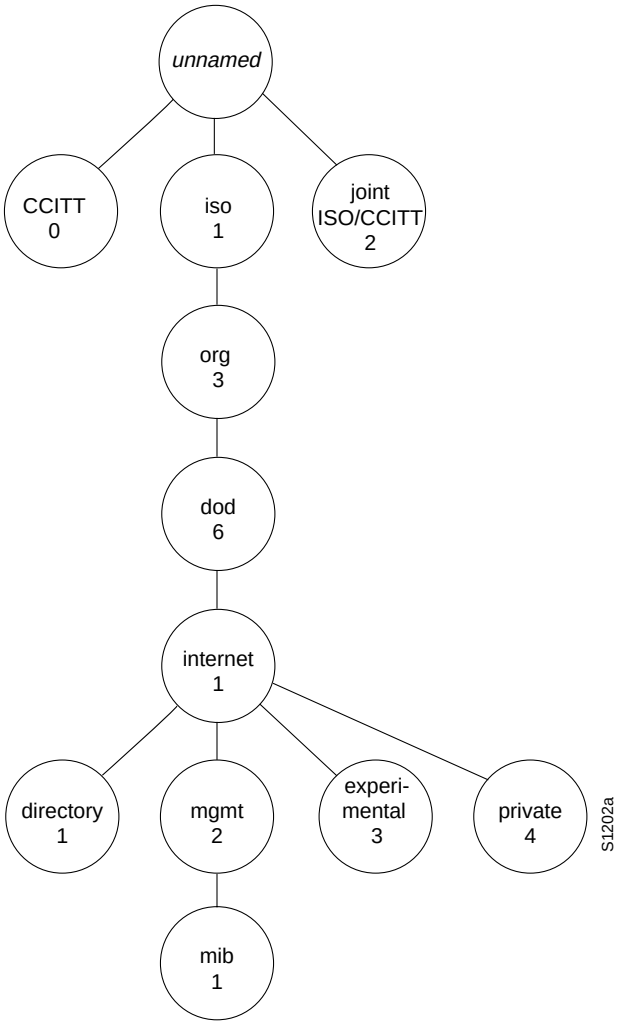

Figure 2 Internet MIB Hierarchy

Cisco MIB

The private Cisco MIB is represented by the object identifier 1.3.6.1.4.1.9, or *iso.org.dod.internet.private.enterprise.cisco*. The Cisco MIB is split into two main areas: local variables and temporary variables.

Note Local variables do not change; they are supported by the Cisco MIB in each subsequent system software release. As indicated by the name, temporary variables might have a limited time span or change with each system software release. (See Figure 3.)

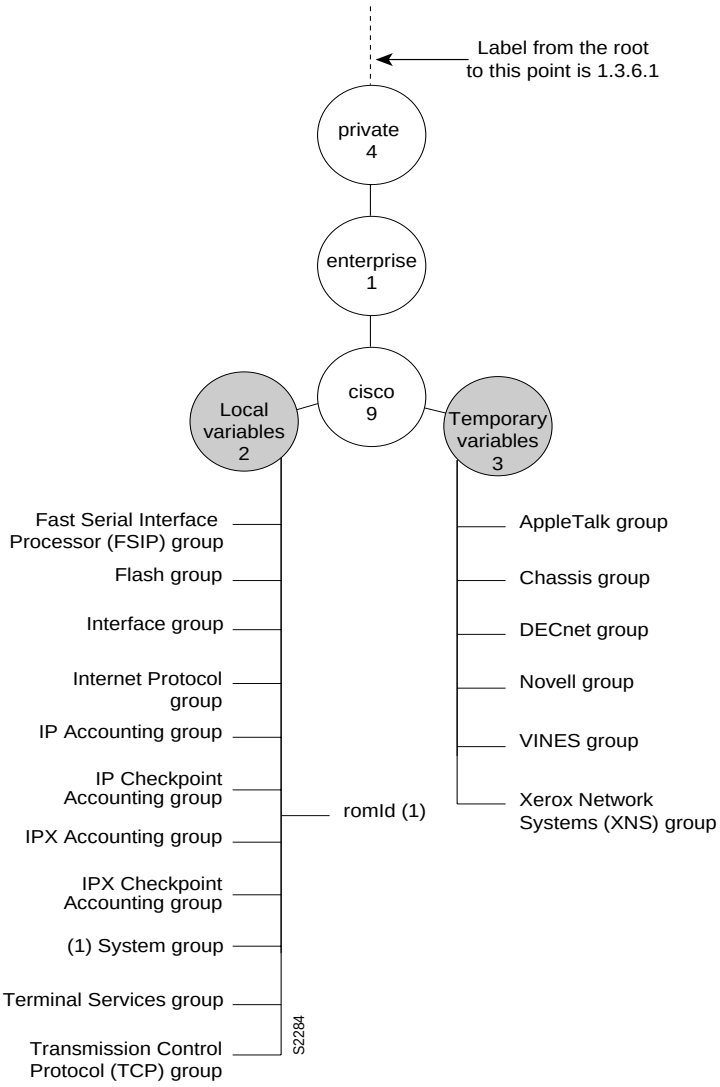

Figure 3 Cisco Private MIB Hierarchy

In Figure 3, the local variables group is identified by 2; its subgroup, called *lsystem*, is identified by 1; and the first variable is *romId* with a value of 1. Therefore, the variable *romId* has a value of 1.3.6.1.4.1.9.2.1.1.0. The appended 0 indicates that 1.3.6.1.4.1.9.2.1.1.0 is the one and only instance of *romId*.

Note Although variables are arranged as shown in Figure 3 and as described in the compilable Cisco MIB file *mib921.txt*, this quick reference guide organizes variable groups and variables within groups alphabetically so you can quickly look up descriptions of MIB variables.

Interpreting the Object Identifier

In this guide, each group of Cisco MIB variables is accompanied by an illustration that indicates the specific *object identifier* for each variable.

For example, in Figure 4 the *object identifier* 1.3.6.1.4.1.9.2.1at the top of the illustration indicates the labeled nodes. The last value is substituted by the number of the Cisco MIB variable. For example, the MIB variable *HostconfigAddr* is indicated by the number 51. The object identifier for *HostconfigAddr* is *iso.org.dod.internet.private.enterprise.cisco.local variables.interface group.HostconfigAddr* or *1.3.6.1.4.1.9.2.1.1.51*.

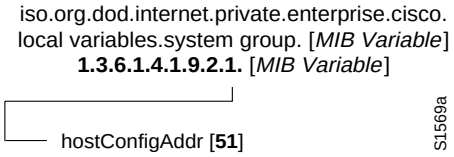

Figure 4 Object Identifier Example for a Cisco MIB Variable

Tables

When network management protocols use names of MIB variables in messages, each name has a suffix appended. For simple variables, the suffix 0 refers to the instance of the variable with that name. A MIB also can contain tables of related variables. A table allows a related set of variables to be applied across several devices or interfaces.

Following is an excerpt of the information on IP Routing Table (known as *lipRoutingTable*) from the *mib921.txt* file:

```
lipRoutingTable OBJECT-TYPE
SYNTAX  SEQUENCE OF LIpRouteEntry
ACCESS  not-accessible
STATUS  mandatory
DESCRIPTION
"A list of IP routing entries."
::= { lip 2 }

lipRouteEntry OBJECT-TYPE
SYNTAX LIpRouteEntry
ACCESS  not-accessible
STATUS  mandatory
DESCRIPTION
"A collection of additional objects in the
cisco IP routing implementation.”

LIpRouteEntry ::=
SEQUENCE {
locRtMask
IpAddress,
locRtCount
INTEGER,
locRtUses
INTEGER
                  }
```

The local IP routing table, *lipRoutingTable*, is illustrated in Table 7. The *lipRoutingTable* contains the following three variables: *locRtMask*, *locRtCount*, and *locRtUses*. The index for this table is the destination address of the IP route, or *ipRouteDest*. If there are *n* number of routes available to a device, there will be *n* rows in the IP routing table.

In Table 2, for the route with the destination IP address of 131.104.111.1 the routing table network mask is 255.255.255.0. The number of parallel routes within the routing table is 3, and the route was used in a forwarding operation two times.

Table 2 IP Routing

ipRouteDest	locRtMask	locRtCount	locRtUses
131.104.111.1	255.255.255.0	3	2
133.45.244.245	255.255.255.0	1	1

Typically, an instance identifier can be a unique interface number or a 0, as described earlier with the *romId* example. An instance identifier can also be an Internet Protocol (IP) address. For example, to find the network mask for the route with a destination address of *131.104.211.243*, use the variable *locRtMask* with an instance identifier of *131.104.211.243*. The format is *locRtMask.131.104.211.243*.

Note In this guide, when variables belong to a table, they are listed in the section describing the table. The following tag is used to indicate the end of a table.

End of Table

All variables before this tag are part of the table.

Local Variables

The local variables section pertains to all Cisco devices and contains the following groups.

Note Although variables are arranged as shown in Figure 3 and as described in the compilable Cisco MIB file *mib921.txt*, this quick reference guide organizes variable groups and variables within groups alphabetically, so you can quickly look up descriptions of MIB variables.

- Flash group

  Pertains to the Flash memory used to store, boot, and write system software images. Includes information such as Flash memory size and erases Flash memory and writes Flash memory to a Trivial File Transfer Protocol (TFTP) server.

- Fast Serial Interface Processor (FSIP)

  Provides information about the fast serial interface processor card.

- Interface group

  Provides information on Cisco device interfaces, such as traffic statistics, line status, average speed of input and output packets, error checking, and more.

- Internet Protocol (IP) group

  Provides information about devices running IP. Includes information such as how and from whom an interface obtained its address, Internet Control Message Protocol (ICMP) messages, and number of packets lost.

- IP Accounting group

  Provides access to the active database that is created and maintained if IP accounting is enabled on a router.

- IP Checkpoint Accounting group

  Records the number of bytes and packets switched through the system on an IP address basis, if IP checkpoint accounting is enabled on a router.

- System group

  Provides information on system-wide parameters for Cisco devices, such as software version, host name, domain name, buffer size, configuration files, and environmental statistics.

- Terminal Server group

  Provides information about the terminal server products, such as number of physical lines, line status, line type, line speed, type of flow control, and type of modem.

- Transmission Control Protocol (TCP) group

  Provides statistics on the number of input and output bytes and packets for TCP connections.

Temporary Variables

This section is equivalent to the experimental space defined by the Structure of Management Information (SMI). These variables are subject to change for each Cisco Systems software release.

Temporary variables consists of the following groups, which are presented in alphabetical order. (See Figure 3.)

- AppleTalk group

  Pertains to devices running the AppleTalk protocol. Includes information such as total number of input and output packets, number of packets with errors, and number of packets with Address Resolution Protocol (ARP) requests and replies.

- Chassis group

  Pertains to hardware information about Cisco devices. Includes information such as the types of cards used by the device, the hardware version of the cards, and the number of slots in the chassis.

- DECnet group

  Pertains to devices running the DECnet protocol. Includes information such as hop count, host name, total packets received and sent, and number of packets with header errors.

- Novell group

  Pertains to devices running the Novell protocol. Includes information such as total number of input and output packets, number of packets with errors, and number of packets with Service Access Point (SAP) requests and replies.

- Virtual Network System (VINES) group

  Pertains to devices running the VINES protocol. Includes information such as total number of input and output packets, number of packets with errors, and number of packets with Internet Control Protocol (ICP) requests and replies.

- Xerox Network Systems (XNS) group

  Pertains to devices running the XNS protocol. Includes information such as number of packets forwarded, total number of input packets, and total number of packets with errors.

Terminology

This section presents the syntax and access type categories used to describe each variable. For details on syntax, refer to RFC 1155.

Syntax

The syntax describes the format of the information, or value, that is returned upon monitoring or setting information in a device with a MIB variable. The syntax can be any one of the following categories:

- Counter

  A counter is a nonnegative integer that increases until it reaches some maximum value. After reaching the maximum value, it rolls back to zero. For example, the variable *locIfipInPkts* counts the number of IP protocol input packets on an interface.

- Display string

  A display string is a printable ASCII string. It is typically a name or description. For example, the variable *netConfigName* provides the name of the network configuration file for a device.

- Integer

  An integer is a numeric value. It can be an actual number, for example, the number of lost IP packets on an interface. It also can be an arbitrary number that represents a nonnumeric value. For example, the variable *tsLineType* returns the type of terminal server line to the SNMP manager. A 2 indicates a console line; a 3 indicates a terminal line; and so on.

- Network address

  A network address represents an address for an interface or device. Currently, only the Internet Protocol (IP) is supported. For example, the variable *hostConfigaddr* indicates the IP address of the host that provided the host configuration file for a device.

- Timeticks

  Timeticks is a nonnegative integer that counts the hundredths of a second since an event. For example, the variable *loctcpConnElapsed* provides the length of time that a TCP connection has been established.

Access

The access type describes whether a MIB variable can be used under one of the following circumstances:

- Read-only

  This variable can be used to monitor information only. For example, the *locIPUnreach* variable whose access is read-only, indicates whether Internet Control Message Protocol (ICMP) packets concerning an unreachable address will be sent.

- Read-write

  This variable can be used to monitor information and to set a new value for the variable. For example, the *tsMsgSend* variable whose access is read-write, determines what action to take after a message has been sent.

The possible integer values for this variable follow:

1 = nothing
2 = reload
3 = message done
4 = abort

- Write-only

  This variable can be used to set a new value for the variable only. For example, the *writeMem* variable whose access is write-only, writes the current (running) router configuration into nonvolatile memory where it can be stored and retained even if the router is reloaded. If the value is set to 0, the *writeMem* variable erases the configuration memory.

Internetwork Management

The International Organization for Standards (ISO) Network Management Forum defined five areas of network management: fault, configuration, security, performance, and accounting. Cisco MIB variables can be mapped to each of these areas (as described in this section) and used to manage your internetwork.

- Fault Management

  Fault management involves running diagnostic tests on the internetwork, analyzing the results, and isolating and resolving problems.

  Example:

  Several of the variables described in the section “Basic” on page 82 provide resources for troubleshooting. For example, the variables *freemem*, *ping*, and *whyreload*, provide information on why a router was reloaded, whether a device is functioning, and how much memory is currently available in a device.

  The variables described in the section “Environmental Monitor Card” on page 97 provide feedback on the physical status of the AGS+ router.

Statistics from variables in the section “Interface Table” on page 48 record the number of packets dropped on particular interfaces so that they can be identified as potential trouble spots, and so on.

- Configuration Management

  Configuration management involves monitoring and controlling the configuration of devices on the internetwork.

  Example:

  The *locIPhow* and *locIPwho* variables described in the section “Internet Protocol (IP) Group” starting on page 73 provide information on how a device received its IP address and the device that provided it with its address.

  The variables described in the section “Host Configuration File” on page 106 and “Network Configuration File” on page 107 provide configuration file names and addresses of hosts supplying network configuration files.

  The variables described in the section “System Configuration” on page 108 provide information such as the name of the host that supplied the system boot image for a device and the name of the boot image.

- Security Management

  Security management deals with controlling access to network resources through the use of authentication techniques and authorization policies.

  Example:

  The variable *authAddr* contains the address of the last SNMP manager that failed the authorization check. The *locIPSecurity* variable provides the IP security level assigned to an interface.

- Performance Management

  Performance management measures traffic flow across the internet, calculates the number of packets that are successfully transmitted against those that are dropped, and so on, in order to optimize efficiency.

Example:

The variables described in the section “CPU Utilization” on page 96 provide feedback on CPU performance. The variables described in “Interface Group” on page 47 provide statistics on time between packets sent, number of packets transmitted successfully, and so on.

- Accounting Management

  Accounting management involves collecting and processing data related to resource consumption on the internet.

  Example:

  The variables described in the section “IP Checkpoint Accounting Table” on page 80 provide numerous statistics such as packets and bytes sent successfully or dropped.

Cisco-Supported MIBs

Cisco supports several MIBs, which are described in the following Requests for Comments (RFCs). Also listed are RFCs describing the Internet standards that Cisco Systems follows with regard to its MIB format and the SNMP protocol.

- RFC 1155, *Structure and Identification of Management Information for TCP/IP-based Internets*, May 1990

  Describes the common structures and identification scheme for the definition of management information for use with TCP/IP-based internets. Formal descriptions of the structure are given using Abstract Syntax Notation One (ASN.1)

- RFC 1156, *Management Information Base for Network Management of TCP/IP-based Internets*, May 1990

  Describes the initial version of the standard Internet Management Information Base, MIB I. MIB I is superceded by MIB II, as described in RFC 1213

- RFC 1157, *A Simple Network Management Protocol (SNMP)*, May 1990

  Describes the SNMP architecture and supported operations.

- RFC 1212, *Concise MIB Definitions*, March 1991

Describes the format for producing concise, yet descriptive, MIB modules.

- RFC 1213, *Management Information Base for Network Management of TCP/IP-based Internets: MIB-II*, March 1991

  Describes the Internet standard MIB II for use with network management protocols in TCP/IP-based internets.

- RFC 1215, *A Convention for Defining Traps for use with the SNMP*, March 1991

  Describes the SNMP standardized traps and provides a means for defining enterprise-specific traps.

- RFC 1231, *IEEE 802.5 Token Ring MIB*, May 1991

  Describes the managed objects used for managing subnetworks that use the IEEE 802.5 Token Ring technology.

  Cisco implements the mandatory tables (Interface table and Statistics table), but not the optional table (Timer table) of this MIB. Support is provided for only the CSC-R16, STR (dual-port Token Ring) and CTR (Token Ring card for the cBus controller) cards.

- RFC 1285, *FDDI Management Information Base*, January 1992

  Describes the managed objects for Fiber Distributed Data Interface (FDDI) devices that are accessible via the Simple Network Management Protocol (SNMP).

  Cisco Systems supports only some of the variables in the Station Management (SMT) and Media Access Control (MAC) groups of this MIB. Refer to the Cisco publication *FDDI MIB Variables in 9.0 Product Update Bulletin No. 181*.

- RFC 1286, *Definitions of Managed Objects for Bridges*, December 1991

  Cisco supports all of the groups described in this MIB, including the following groups: dotldBase, dotldSr, dot1dStp, and dotIdTp.

- RFC 1315, *Management Information Base for Frame Relay DTEs*, April 1992

  Cisco supports the following tables in this MIB:

  — Data Link Connection Management Interface Table

  — Circuit Table

  — Frame Relay Globals

  — Data Link Connection Management Interface Related Traps

  The Error Table is not supported in this MIB.

- RFC 1381, *SNMP MIB Extension for X.25 LAPB*, November 1992

  Cisco supports the following tables in this MIB:

  — LAPB Admn Table (read-only)

  — LAPB Operating Parameters Table

  — LAPB Flow Table

  The LAPB XID Table is not supported in this MIB.

- RFC 1382, *SNMP MIB Extension for the X.25 Packet Layer*, November 1992

  The X.25 packet layer MIB is available under the ifType node rfc887-x25 (5) registered under the MIB-II transmission Object Identifier. This applies to all X.25 interfaces, including any DDN-X.25 encapsulation interfaces. Cisco supports the following tables in this MIB:

  — X.25 Administration Table (read-only)

  — X.25 Operational Table

  — X.25 Statistics Table

  — X.25 Channel Table (read-only)

  — X.25 Circuits Information Table (read-only)

— X.25 Traps (Both must be configured)

The following tables are not supported in this MIB:

— X.25 Cleared Circuit Table

— X.25 Call Parameter Table

To obtain copies of RFCs, use the **ftp nic.ddn.mil** command. Log in as **anonymous** and press Return when prompted for the password. Enter the **cd rfc** command to change to the correct directory. Use the **get rfc-index.txt** command to retrieve a list of all available RFCs. To obtain a copy of any specific RFC, enter **get rfcnnnn.txt**, where *nnnn* is the RFC number.

Related Cisco Publications

For detailed information on configuration and troubleshooting commands, refer to the following Cisco publications:

- *Router Products Configuration Guide*, System Software Release 9.21
- *Router Products Command Reference*, System Software Release 9.21
- *Communication Server Configuration Guide*, System Software Release 9.25(2)
- *Communication Server Command Reference*, System Software Release 9.25(2)

Users of CiscoWorks, the Cisco router management software, can refer to the *CiscoWorks User Guide* for information on Cisco router management software features and its use of MIB variables for the purposes of graphing and analyzing network performance, ensuring configuration consistency, troubleshooting, and more.

Suggested Reading

Following are suggested reading materials:

- Leinwand, A. and K. Fang, *Network Management: A Practical Perspective.* Reading, Massachusetts: Addison-Wesley Publishing Company, Inc.; 1993.

- Rose, M. T. *The Simple Book: An Introduction Management of TCP/IP-based Internets*. Englewood Cliffs, New Jersey: Prentice-Hall; 1991.
- Rose, M. T. *The Simple Book: An Introduction to Internet Management, 2nd edition.* Englewood Cliffs, New Jersey: Prentice-Hall; 1993.
- Stallings, W. *SNMP, SNMPv2, and CMIP: The Practical Guide to Network Management Standards*. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc.; 1993.

Object Identifier Numbers for Variables

The figures in this section provide a visual overview of the Cisco MIB variables along with the object identifier numbers for each MIB variable. The MIB variables are arranged alphabetically within each figure (in the same order in which they appear in the sections of this guide).

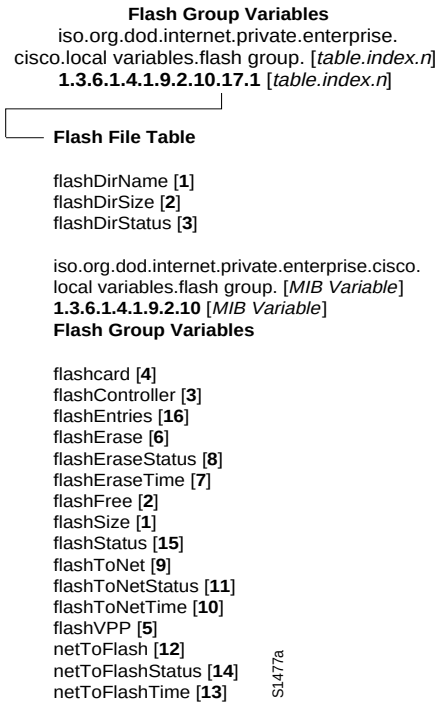

Figure 5 Local Variables: Flash File Table and Flash Group

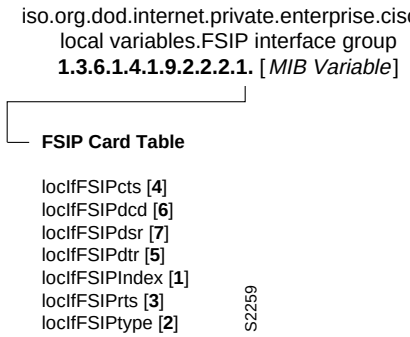

Figure 6 FSIP Group Variables

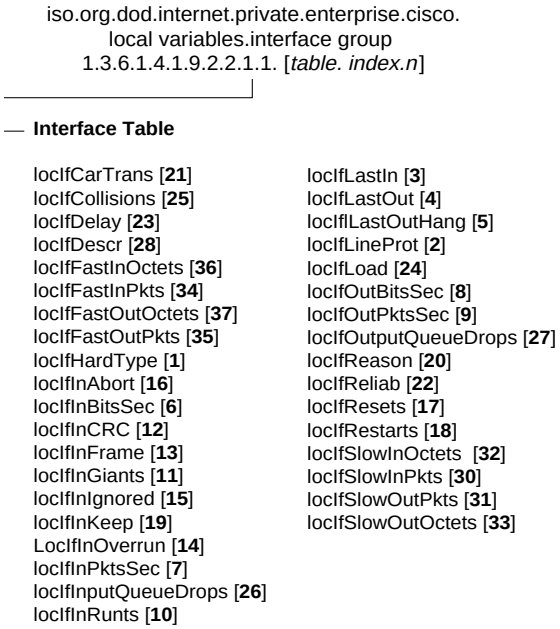

Figure 7 Local Variables: Interface Group Table

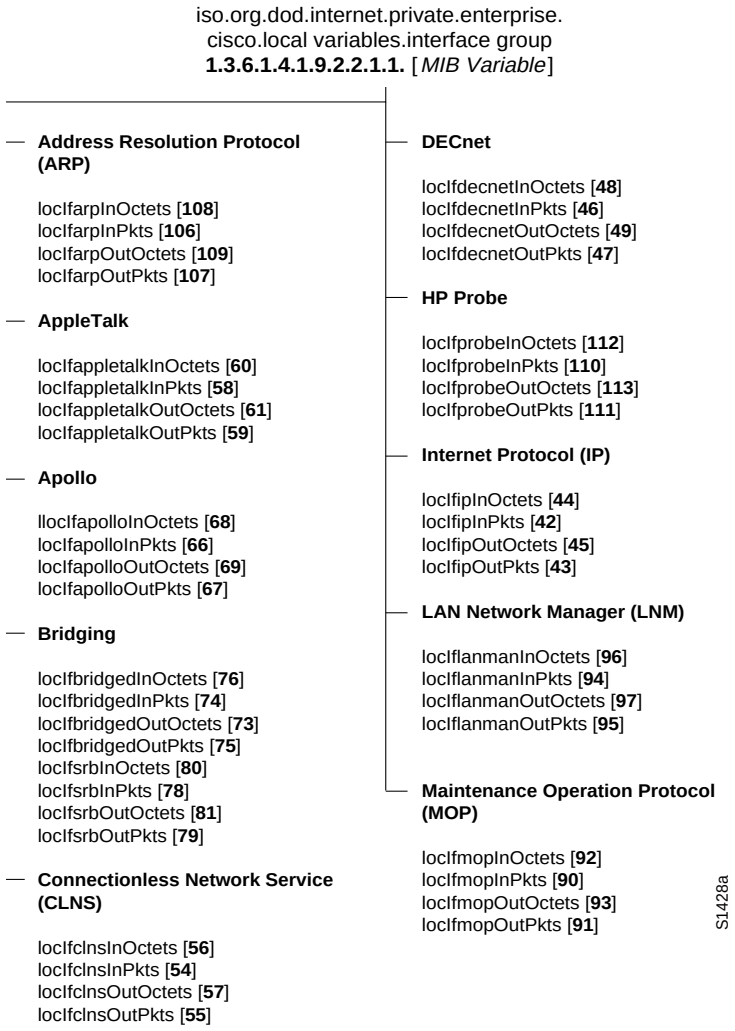

Figure 8 Local Variables: Interface Group—ARP, AppleTalk, Apollo, Bridging, CLNS, DECnet, HP Probe, IP, LNM, and MOP

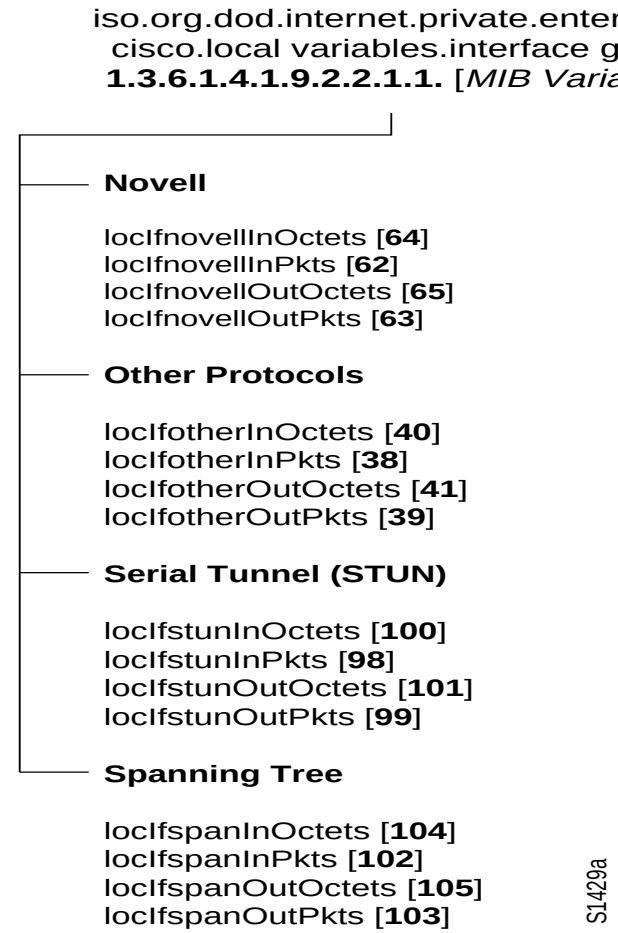

Figure 9 Local Variables: InterfaceGroup—Novell, Other Protocols, STUN, Spanning Tree

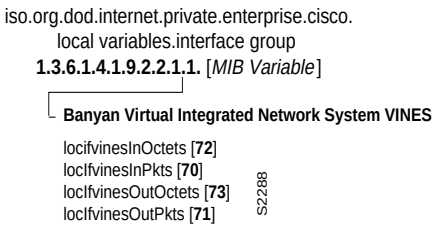

Figure 10 Local Variables: Interface Group—Vines

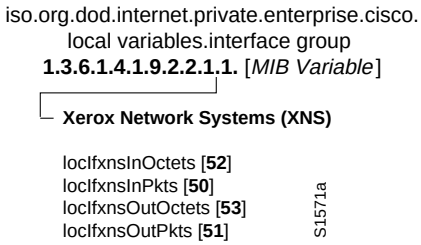

Figure 11 Local Variables: Interface Group—XNS

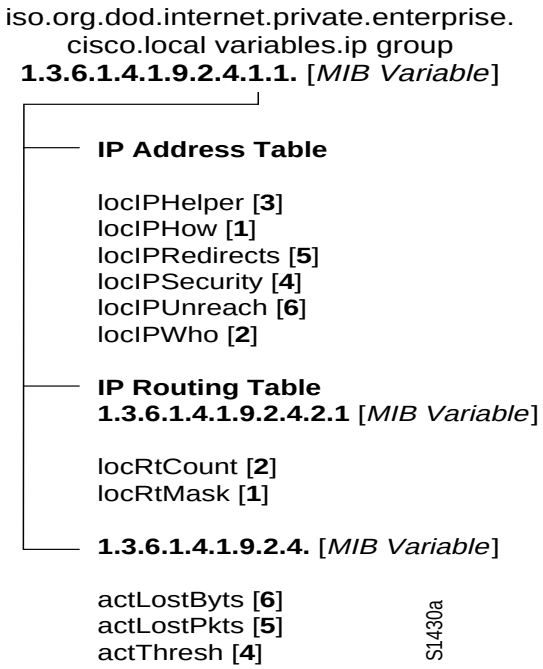

Figure 12 Local Variables: Internet Protocol (IP) Group

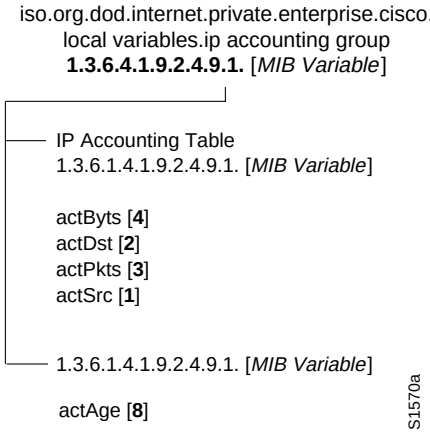

Figure 13 Local Variables: IP Accounting Table

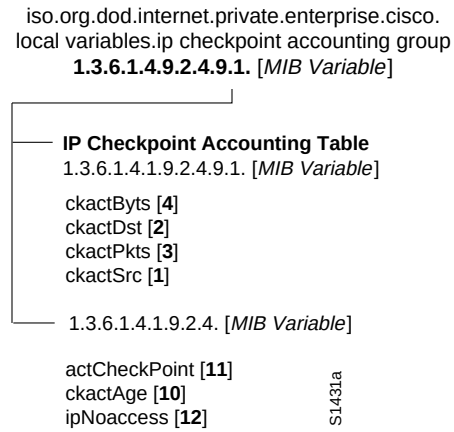

Figure 14 Local Variables: IP Checkpoint Accounting Table

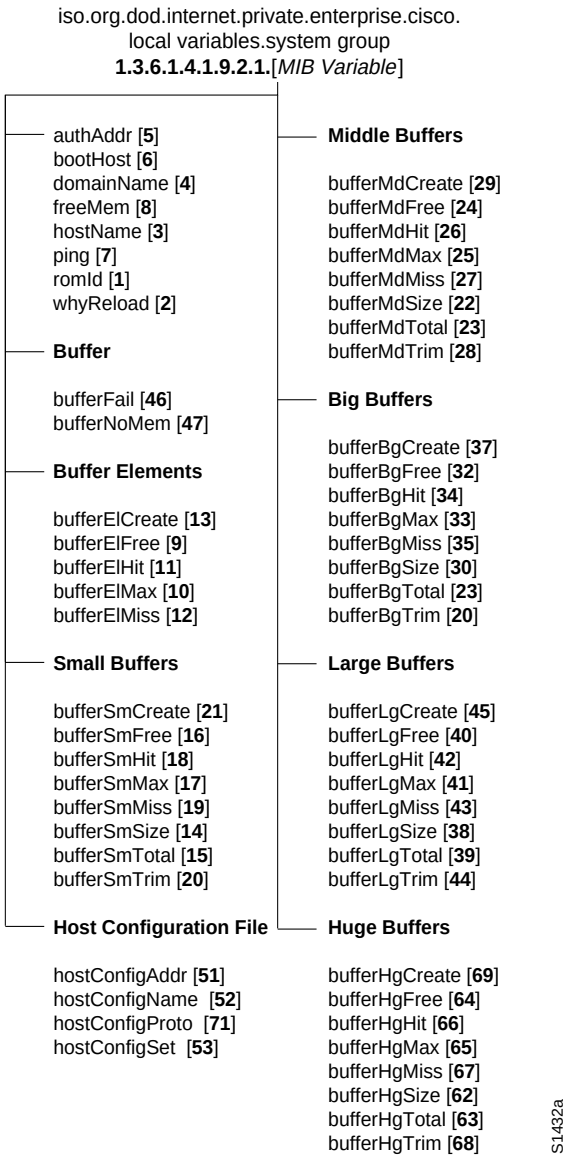

Figure 15 Local Variables: System Group—Buffers

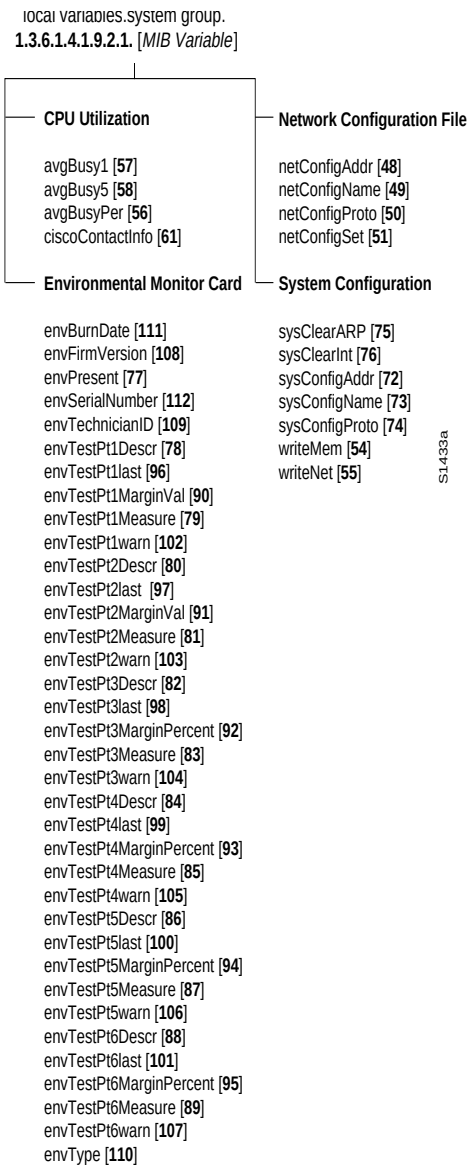

Figure 16 Local Variables: System Group—CPU Utilization and Environmental Monitor Card

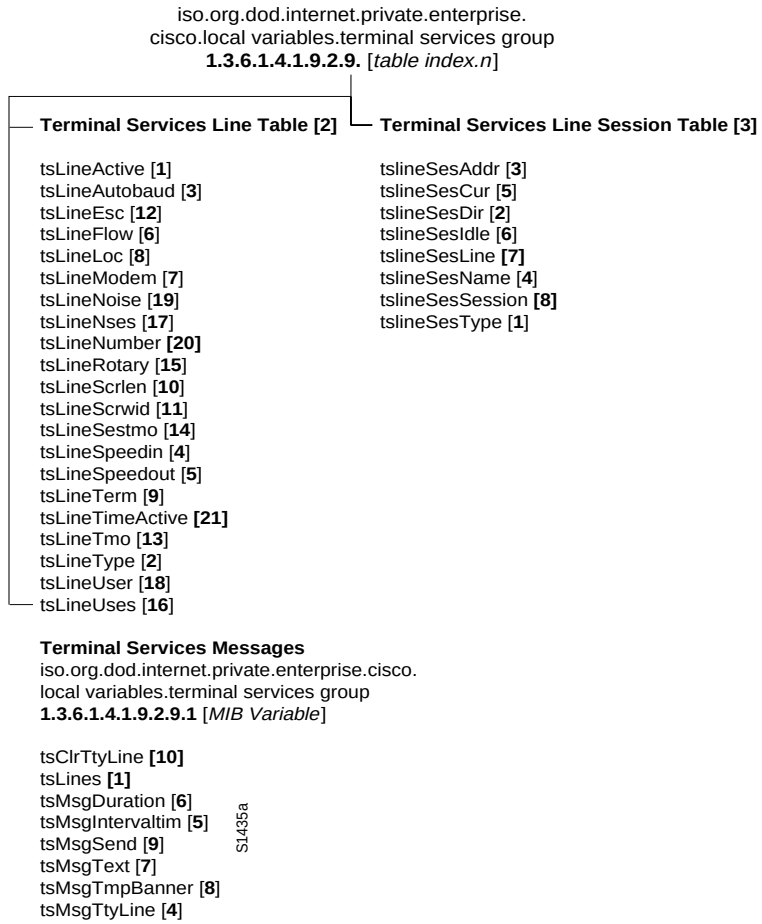

Figure 17 Local Variables: Terminal Server Group

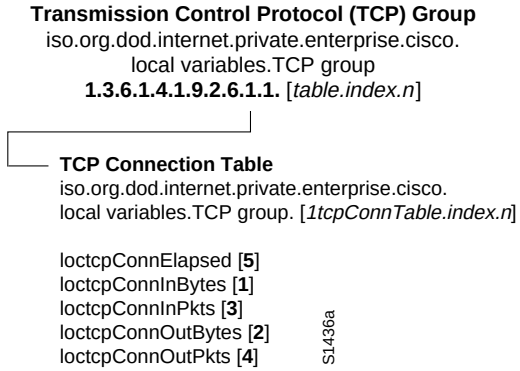

Figure 18 Local Variables: Transmission Control Protocol (TCP) Connection Table

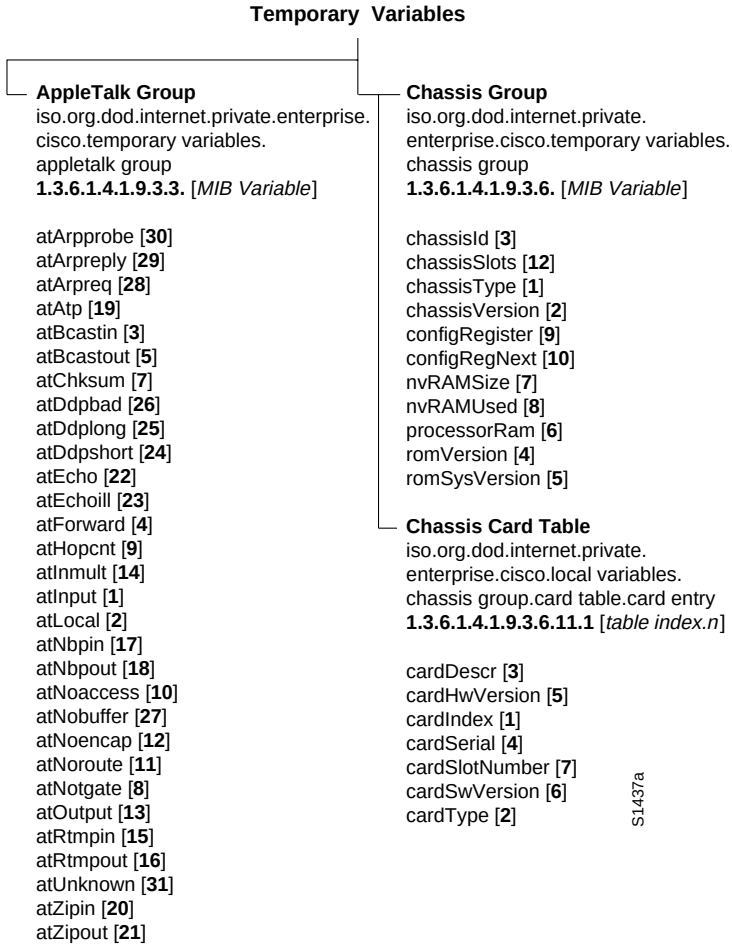

Figure 19 Temporary Variables: AppleTalk and Chassis

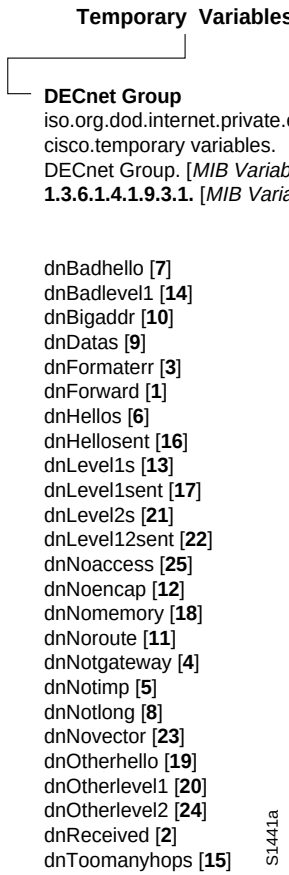

Figure 20 Temporary Variables: DECnet

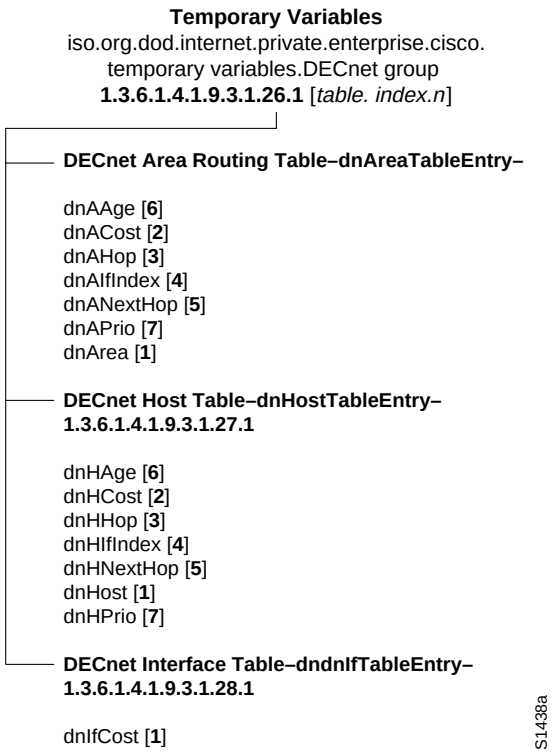

Figure 21 Temporary Variables: DECNet Tables

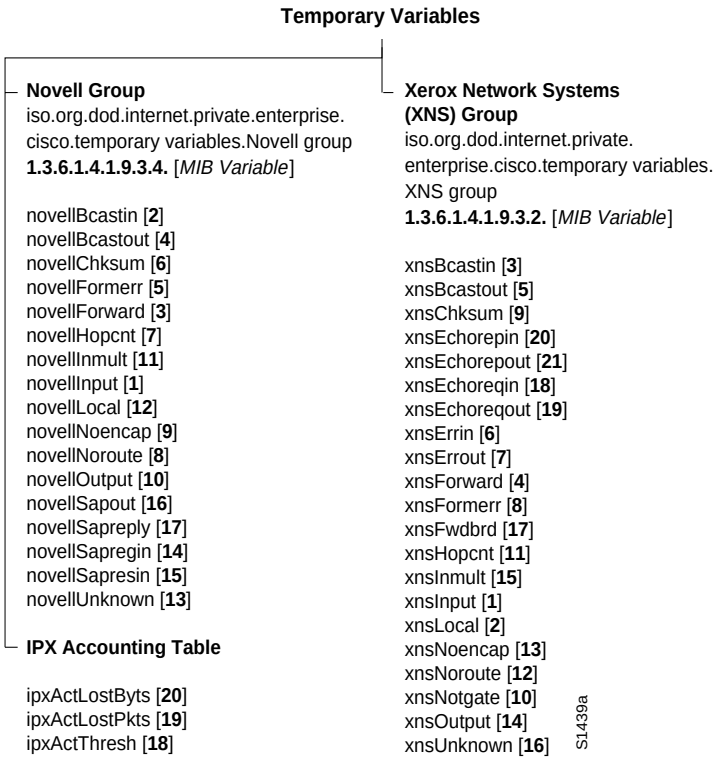

Figure 22 Temporary Variables: Novell and XNS

Virtual Integrated Network Service (VINES) Group
iso.org.dod.internet.private.enterprise.cisco.
temporary variables.vines group
1.3.6.1.4.1.9.3.5. [*MIB Variable*]

vinesBcastfwd [**7**]
vinesBcasttin [**5**]
vinesBcastout [**6**]
vinesCksumerr [**12**]
vinesClient [**28**]
vinesEchoIn [**22**]
vinesEchoOut [**23**]
vinesEncapsfailed [**15**]
vinesFormaterror [**11**]
vinesForwarded [**4**]
vinesHopcount [**13**]
vinesIcpIn [**17**]
vinesIcpOut [**18**]
vinesInput [**1**]
vinesLocaldest [**3**]
vinesMacEchoIn [**20**]
vinesMacEchoOut [**21**]
vinesMetricOut [**19**]
vinesNet [**26**]
vinesNocharges [**10**]
vinesNoroute [**14**]
vinesNotgt4800 [**9**]
vinesNotlan [**8**]
vinesOutput [**2**]
vinesProxy [**24**]
vinesProxyReply [**25**]
vinesSubnet [**27**]
vinesUnknown [**16**]

S2584

Figure 23 Temporary Variables: VINES I

Virtual Integrated Network Service (VINES) Interface Table
iso.org.dod.internet.private.enterprise.cisco.
temporary variables.Vines group
1.3.6.1.4.1.9.3.5.29. [*If*].[*Variable*]

vinesIfAccesslist [**3**]
vinesIfArpEnabled [**5**]
vinesIfEnctype [**2**]
vinesIfFastOkay [**11**]
vinesIfInputRouterFilter [**81**]
vinesIfLineup [**10**]
vinesIfMetric [**1**]
vinesIfInputNetworkFilter [**82**]
vinesIfOutputNetworkFilter [**83**]
vinesIfPropagate [**4**]
vinesIfRedirectInterval [**8**]
vinesIfRouteCache [**12**]
vinesIfRxArp0 [**25**]
vinesIfRxArp1 [**26**]
vinesIfRxArp2 [**27**]
vinesIfRxArp3 [**28**]
vinesIfRxArpIllegal [**29**]
vinesIfRxBcastDuplicate [**47**]
vinesIfRxBcastForwarded [**46**]
vinesIfRxBcastHelpered [**45**]
vinesIfRxBcastin [**20**]
vinesIfRxChecksumError [**24**]
vinesIfRxEcho [**48**]
vinesIfRxFormatError [**18**]
vinesIfRxForwarded [**21**]
vinesIfRxIcpError [**30**]
vinesIfRxIcpIllegal [**32**]
vinesIfRxIcpMetric [**31**]
vinesIfRxIpc [**33**]
vinesIfRxLocalDest [**19**]
vinesIfRxMacEcho [**49**]
vinesIfRxNoRoute [**22**]
vinesIfRxNotEnabled [**17**]
vinesIfRxProxyReply [**50**]
vinesIfRxRtp0 [**34**]
vinesIfRxRtp1 [**35**]
vinesIfRxRtp2 [**36**]
vinesIfRxRtp3 [**37**]
vinesIfRxRtp4 [**38**]
vinesIfRxRtp5 [**39**]
vinesIfRxRtp6 [**40**]
vinesIfRxRtpIllegal [**41**]
vinesIfRxIpUnknownCnt [**43**]
vinesIfRxIpcUnknownCnt [**44**]
vinesIfRxSpp [**42**]
vinesIfRxZeroHopCount [**23**]
vinesIfServerless [**6**]
vinesIfSplitDisabled [**9**]
vinesIfTxArp0 [**63**]
vinesIfTxArp1 [**64**]
vinesIfTxArp2 [**65**]
vinesIfTxArp3 [**66**]
vinesIfTxBcast [**52**]
vinesIfTxBcastForwarded [**61**]
vinesIfTxBcastHelpered [**62**]
vinesIfTxEcho [**78**]
vinesIfTxFailedAccess [**55**]
vinesIfTxFailedDown [**56**]
vinesIfTxFailedEncaps [**54**]
vinesIfTxForwarded [**53]**
vinesIfTxIcpError [**67**]
vinesIfTxIcpMetric [**68**]
vinesIfTxIpc [**69**]
vinesIfTxMacEcho [**29**]
vinesIfTxNotBcastNotgt4800 [**54**]
vinesIfTxNotBcastNotlan [**58**]
vinesIfTxNotBcastPpcharge [**60**]
vinesIfTxNotBcastToSource [**57**]
vinesIfTxProxy [**80**]
vinesIfTxRtp0 [**70**]
vinesIfTxRtp1 [**71**]
vinesIfTxRtp2 [**72**]
vinesIfTxRtp3 [**73**]
vinesIfTxRtp4 [**74**]
vinesIfTxRtp5 [**75**]
vinesIfTxRtp6 [**76**]
vinesIfTxSpp [**77**]
vinesIfTxUnicast [**51**]

S2585

Figure 24 Temporary Variables: VINES II

Local Variables

This section describes the local variables within the Cisco product line. Certain groups of variables may or may not be present depending upon the software options and configuration in the managed device.

The local variables section contains the following groups of variables:

- Flash, page 41
  - Flash File Table
- Fast Serial Interface Processor (FSIP), page 45
- Interface, page 47
  - Interface Table
  - Across All Interfaces
  - Address Resolution Protocol (ARP)
  - AppleTalk
  - Apollo
  - Bridging
  - Connections Network Service (CLNS)
  - DECnet
  - HP Probe
  - Internet Protocol (IP)
  - LAN Network Manager (LNM)
  - Maintenance Operation Protocol (MOP)
  - Novell
  - Other Protocols
  - Serial Tunnel (STUN)
  - Spanning Tree
  - Banyan VINES
  - Xerox Network Systems (XNS)

- Internet Protocol (IP), page 73
  - IP Address Table
  - IP Routing Table
- IP Accounting, page 77
  - IP Accounting Table
- IP Checkpoint Accounting, page 79
  - IP Checkpoint Accounting Table
- System, page 82
  - Basic
  - Buffer
  - CPU Utilization
  - Environmental Monitor Card
  - Host Configuration File
  - Network Configuration File
  - System Configuration
- Terminal Server, page 110
  - Terminal Server Line Table
  - Terminal Server Line Session Table
  - Terminal Server Messages
- Transmission Control Protocol (TCP), page 121
  - TCP Connection Table

Flash Group

The Flash memory card is an add-in card of Flash EPROM (erasable programmable read-only memory) storage onto which system software images can be stored, booted, and rewritten.

Flash File Table

The local Flash File table, *lflashFileDirTable*, contains information on a per file basis and includes the following three variables: *flashDirName*, *flashDirSize*, and *flashDirStatus.* The index to this table is *flashEntries*, or the number of Flash files. If the device has *n* number of Flash files, the table will contain *n* number of rows.

For example, in Table 3, the *flash1* file has a directory size of 50 octets and its status is valid, represented by the integer 1.

Table 3 Flash File Table

flashEntries	flashDirName	flashDirSize	flashDirStatus
1	flash1	50	1
2	flash2	100	1
3	flash3	200	2

flashDirName

Provides the name associated with a Flash directory entry.

Syntax: Display string

Access: Read-only

flashDirSize

Provides the size (in octets) of a Flash directory entry.

Syntax: Integer

Access: Read-only

flashDirStatus

Indicates the status of the Flash directory entry.

Syntax: Integer (1 = valid, 2 = deleted)

Access: Read-only

End of Table

flashcard

Provides the type of card connected to the Flash card installed in the router. For example, the type of cards connected to the Flash card could be either CSC-MS or CSC-MC+.

Syntax: Display string

Access: Read-only

flashController

Provides the type of Flash controller (either CCTL or CCTL2) installed in the router.

Syntax: Display string

Access: Read-only

flashEntries

Provides the number of directory entries, or files, that exist in the Flash memory directory.

Syntax: Integer

Access: Read-only

flashErase

This variable sets a request to erase Flash memory, freeing up all available memory space. All of the Flash memory is erased out. Individual files cannot be erased from Flash memory.

Syntax: Integer

Access: Write-only

flashEraseTime

Indicates the last time the Flash memory was erased.

Syntax: Timeticks

Access: Read-only

flashFree

Provides the amount of available Flash memory in octets.

Syntax: Integer

Access: Read-only

flashSize

Provides the amount of total Flash memory in octets.

Syntax: Syntax:Integer

Syntax: Access:Read-only

flashToNet

Requests to write the Flash memory to a Trivial File Transfer Protocol (TFTP) server. The value (display string) is the name of the Flash file being sent, or written, to the server. The instance ID is the IP address of the TFTP host.

This copy of the system image can serve as a backup copy and can also be used to verify that the copy in the Flash memory is the same as the original file.

The Flash memory card can be used as a TFTP file server for other routers on the network. This feature allows you to boot a remote router with an image that resides in the Flash server memory.

Syntax: Display string

Access: Write-only

flashToNetTime

Indicates the last time a file was copied from the Flash memory in the router to the TFTP host.

Syntax: Timeticks

Access: Read-only

flashVPP

Provides the status of the VPP DIP jumper on the Flash memory card. Files can be written to the Flash memory card only if the VPP DIP jumper is turned on.

Syntax: Integer (1 = VPP enabled/Flash write enabled, 2 = VPP disabled/Flash write disabled)

Access: Read-only

netToFlash

Copies a software image from Trivial File Transfer Protocol (TFTP) server to the Flash memory on the router. The value (display string) is the name of the file being sent, or written, to the Flash memory. The instance ID is the IP address of the TFTP host.

The TFTP image copied to the Flash memory must be at least System Software Release 9.0 or later. If earlier system software is copied into the Flash memory, the host processor card will not recognize the CSC-MC+ card upon the next reboot.

If free Flash memory space is unavailable, or if the Flash memory has never been written to, the erase routine is required before new files can be copied.

Syntax: Display string

Access: Write-only

netToFlashTime

Indicates the last time a file was copied from a Trivial File Transfer Protocol (TFTP) server to the Flash memory on the router.

Syntax: Timeticks

Access: Read-only

Fast Serial Interface Processor (FSIP) Group

The local Fast Serial Interface Processor (FSIP) Card table, *lfsipTable*, contains information about FSIP cards used by the Cisco 7000 and includes the following six variables that provide information about the processor: *locIfFSIPtype*, *locifFSIPrts*, *locIfFSIPcts*, *locIfFSIPdtr*,

locIfFSIPdcd, and *locIfFSIPdsr*. The index to this table is *locIfSIPIndex* which indicates the interface index of the card corresponding to its *IfIndex*.

Table 4 FSIP Card Table

locIFSIPIndex	locIfSIPtype	locIFSIPrts	locIFSIPcts	and so on
1	DCE	1	2	
2	DTE	1	3	
and so on				

locIfFSIPcts

Indicates whether the CTS signal is up or down.

Syntax: Integer (1 = not available, 1 = up, 2 = down)

Access: Read-only

locIfFSIPdcd

Indicates whether the DCD signal is up or down.

Syntax: Integer (1 = not available, 2 = up, 3 = down)

Access: Read-only

locIfFSIPdsr

Indicates whether the DSR signal is up or down.

Syntax: Integer (1 = not available, 2 = up, 3 = down)

Access: Read-only

locIfFSIPdtr

Indicates whether the DTR signal is up or down.

Syntax: Integer (1 = not available, 2 = up, 3 = down)

Access: Read-only

locIfFSIPrts

Indicates whether the RTS signal is up or down.

Syntax: Integer (1 = not available, 2 = up, 3 = down)

Access: Read-only

locIfFSIPtype

Indicates whether the FSIP line uses DCE or DTE.

Syntax: Integer (1 = not available, 2 = DTE, 3 = DCE)

Access: Read-only

End of Table

Interface Group

The following variables apply to interfaces attached to Cisco devices. These variables can be used to monitor the performance of the network in terms of the number of packets dropped, time allocations for input and output packets, and so on. These variables also can be used for fault management. For example, variable values indicate which interfaces are dropping packets or have had to be restarted several times.

Interface Table

The Interface table, *lifTable*, contains all of the variables in the Interface group. The index to the table is *ifIndex* which indicates the number of the interface. If the device has *n* number of interfaces, the Interface table will contain *n* rows.

In the Interface table shown in Table 5, the first column indicates the number of interfaces on the device. Each of the variables in the interface table occupies one column; for example, *locIfHardType* is shown in a column, followed by *locIfLineProt* in the next column, and so on.

Table 5 Interface Table

Interface Numer	locIfHardType	locIfLineProt	and so on
1	The Interface Table, *lifTable*, contains all of the variables described in the Interface group.		
2			
3			
and so on			

Across All Interfaces

This section contains basic interface variables that apply to all interfaces and are not protocol-specific.

locIfCarTrans

Provides the number of times the serial interface received the carrier detect (CD) signal. If the carrier detect line is changing state often, it might indicate modem or line problems.

Syntax: Integer

Access: Read-only

locIfCollisions

Provides the number of output collisions detected on this interface.

Syntax: Integer

Access: Read-only

locIfDelay

Provides the media-dependent delay in transferring a packet to another interface on the media. The delay is indicated in microseconds. Used by Interior Gateway Routing Protocol (IGRP).

Syntax: Integer

Access: Read-only

locIfDescr

Provides a description of the interface (whether it is Ethernet, serial, and so on) if one was provided by the user previously.

Syntax: Display string

Access: Read-write

locIfFastInOctets

Provides the octet count for inbound traffic routed with fast and autonomous switching.

Syntax: Counter

Access: Read-only

locIfFastOutOctets

Provides the octet count for outbound traffic routed with fast and autonomous switching

Syntax: Counter

Access: Read-only

locIfFastInPkts

Provides the packet count for inbound traffic routed with fast and autonomous switching

Syntax: Counter

Access: Read-only

locIfFastOutPkts

Provides the packet count for outbound traffic routed with fast and autonomous switching.

Syntax: Counter

Access: Read-only

locIfHardType

Provides the type of interface, such as Ethernet, serial, FDDI, and so on.

Syntax: Display string

Access: Read-only

locIfInAbort

Provides the number of input packets that were aborted. Aborted input packets usually indicate a clocking problem between the serial interface and the data link equipment.

Syntax: Integer

Access: Read-only

locIfInBitsSec

Provides a 5-minute exponentially delayed average of CPU input bits per second.

Syntax: Integer

Access: Read-only

locIfInCRC

Provides the number of input packets that had cyclic redundancy checksum (CRC) errors. The CRC generated by the originating station or far-end device does not match the checksum calculated from the data received. On a serial link, CRCs usually indicate noise, gain hits, or other transmission problems on the data link.

Syntax: Integer

Access: Read-only

locIfInFrame

Provides the number of input packet that were received incorrectly with a cyclic redundancy check (CRC) error and a noninteger number of octets. On a serial line, this is usually the result of noise or other transmission problems.

Syntax: Integer

Access: Read-only

locIfInGiants

Provides the number of input packets that were discarded because they exceeded the maximum packet size allowed by the physical media.

Syntax: Integer

Access: Read-only

locIfInIgnored

Provides the number of input packets that were ignored by this interface because the interface hardware ran low on internal buffers. Broadcast storms and bursts of noise can cause the ignored count to be increased.

Syntax: Integer

Access: Read-only

locIfInOverrun

Provides the number of times the serial receiver hardware was unable to hand received data to a hardware buffer because the input rate exceeded the ability of the receiver to handle the data.

Syntax: Integer

Access: Read-only

locIfInPktsSec

Provides a weighted average of input bits and packets transmitted per second in the last 5 minutes.

Syntax: Integer

Access: Read-only

locIfInputQueueDrops

Provides the number of packets dropped because the input queue was full.

Syntax: Integer

Access: Read-only

locIfInRunts

Provides the number of input packets that were discarded because they were smaller than the minimum packet size allowed by the physical media.

Syntax: Integer

Access: Read-only

locIfKeep

Indicates whether keepalives are enabled on this interface.

Syntax: Integer (1 = enabled, 2 = disabled)

Access: Read-only

locIfLastIn

Provides the elapsed time in milliseconds since the last line protocol input packet was successfully received by an interface. Useful for knowing when a dead interface failed.

Syntax: Integer

Access: Read-only

locIfLastOut

Provides the elapsed time in milliseconds since the last line protocol output packet was successfully transmitted by an interface. Useful for knowing when a dead interface failed.

Syntax: Integer

Access: Read-only

locIfLastOutHang

Provides the elapsed time in milliseconds since the last line protocol output packet could not be successfully transmitted.

OR

Provides the elapsed time (in milliseconds) since the interface was last reset because of a transmission that took too long.

Syntax: Integer

Access: Read-only

locIfLineProt

Indicates whether the interface is up or down.

Syntax: Integer (1 = up, 2 = down)

Access: Read-only

locIfLoad

Provides the loading factor of the interface. The load on the interface is calculated as an exponential average over 5 minutes and expressed as a fraction of 255 (255/255 is completely saturated). Used by Interior Gateway Routing Protocol (IGRP).

Syntax: Integer

Access: Read-only

locIfOutBitsSec

Provides a 5-minute weighted average of output bits per second for the specific protocol.

Syntax: Integer

Access: Read-only

locIfOutPktsSec

Provides a 5-minute weighted average of output packets per second for the specific protocol.

Syntax: Integer

Access: Read-only

locIfOutputQueueDrops

Provides the number of packets dropped because the output queue was full.

Syntax: Integer

Access: Read-only

locIfReason

Provides the reason for the most recent status change of the interface.

Syntax: Display string

Access: Read-only

locIfReliab

Provides the level of reliability for the interface. The reliability of the interface is calculated as an exponential average over 5 minutes and expressed as a fraction of 255 (255/255 is 100 percent). Used by Interior Gateway Routing Protocol (IGRP).

Syntax: Integer

Access: Read-only

locIfResets

Provides the number of times the interface was reset internally. An interface can be reset if packets queued for transmission were not sent within several seconds. On a serial line, this can be caused by a malfunctioning modem that is not supplying the transmit clock signal or by a cable problem. If the system notices that the carrier detect line of a serial interface is up, but the line protocol is down, it periodically resets the interface in an effort to restart it. Interface resets also can occur when an interface is looped back or shut down.

Syntax: Integer

Access: Read-only

locIfRestarts

Provides the number of times the interface needed to be completely restarted because of errors.

Syntax: Integer

Access: Read-only

locIfSlowInOctets

Provides the octet count for inbound traffic routed with process switching.

Syntax: Counter

Access: Read-only

locIfSlowInPkts

Provides the packet count for inbound traffic routed with process switching.

Syntax: Counter

Access: Read-only

locIfSlowOutPkts

Provides the packet count for outbound traffic routed with process switching.

Syntax: Counter

Access: Read-only

locIfSlowOutOctets

Provides the octet count for outbound traffic routed with process switching.

Syntax: Counter

Access: Read-only

End of Table

Address Resolution Protocol (ARP)

The following variables in the Interface group apply to interfaces running the Address Resolution Protocol (ARP). ARP provides dynamic addressing between 32-bit IP addresses and Ethernet addresses. For detailed information on ARP, refer to the *Router Products Configuration and Reference* publication.

locIfarpInOctets

Provides the ARP input octet count.

Syntax: Counter

Access: Read-only

locIfarpInPkts

Provides the ARP input packet count. It indicates the number of ARP Reply packets received by this router from other hosts.

Syntax: Counter

Access: Read-only

locIfarpOutOctets

Provides the ARP output octet count.

Syntax: Counter

Access: Read-only

locIfarpOutPkts

Provides the ARP output packet count. It indicates the number of ARP Request packets sent by this router to other hosts on the network.

Syntax: Counter

Access: Read-only

AppleTalk

The following variables in the Interface group apply to interfaces running AppleTalk:

locIfappletalkInOctets

Provides the AppleTalk protocol input octet count.

Syntax: Counter

Access: Read-only

locIfappletalkInPkts

Provides the AppleTalk protocol input packet count.

Syntax: Counter

Access: Read-only

locIfappletalkOutOctets

Provides the AppleTalk protocol output octet count.

Syntax: Counter

Access: Read-only

locIfappletalkOutPkts

Provides the AppleTalk protocol output packet count.

Syntax: Counter

Access: Read-only

Apollo

The following variables in the Interface group apply to interfaces running Apollo:

locIfapolloInOctets

Provides the Apollo protocol input octet count.

Syntax: Counter

Access: Read-only

locIfapolloInPkts

Provides the Apollo protocol input packet count.

Syntax: Counter

Access: Read-only

locIfapolloOutOctets

Provides the Apollo protocol output octet count.

Syntax: Counter

Access: Read-only

locIfapolloOutPkts

Provides the Apollo protocol output packet count.

Syntax: Counter

Access: Read-only

Bridging

The following variables in the Interface group apply to interfaces running bridging protocols:

locIfbridgedInOctets

Provides the bridged protocol input octet count.

Syntax: Counter

Access: Read-only

locIfbridgedInPkts

Provides the bridged protocol input packet count.

Syntax: Counter

Access: Read-only

locIfbridgedOutOctets

Provides the bridged protocol output octet count.

Syntax: Counter

Access: Read-only

locIfbridgedOutPkts

Provides the bridged protocol output packet count.

Syntax: Counter

Access: Read-only

locIfsrbInOctets

Provides the Source-Route Bridging (SRB) protocol input octet count.

Syntax: Counter

Access: Read-only

locIfsrbInPkts

Provides the SRB protocol input packet count.

Syntax: Counter

Access: Read-only

locIfsrbOutOctets

Provides the SRB protocol output octet count.

Syntax: Counter

Access: Read-only

locIfsrbOutPkts

Provides the SRB protocol output packet count.

Syntax: Counter

Access: Read-only

Connectionless Network Service (CLNS)

The following variables in the Interface group apply to interfaces running Connectionless Network Service (CLNS):

locIfclnsInOctets

Provides the CLNS protocol input byte count.

Syntax: Counter

Access: Read-only

locIfclnsInPkts

Provides the CLNS protocol input packet count.

Syntax: Counter

Access: Read-only

locIfclnsOutOctets

Provides the CLNS protocol output byte count.

Syntax: Counter

Access: Read-only

locIfclnsOutPkts

Provides the CLNS protocol output packet count.

Syntax: Counter

Access: Read-only

DECnet

The following variables in the Interface group apply to interfaces running DECnet.

locIfdecnetInOctets

Provides the DECnet protocol input octet count.

Syntax: Counter

Access: Read-only

locIfdecnetInPkts

Provides the DECnet protocol input packet count.

Syntax: Counter

Access: Read-only

locIfdecnetOutOctets

Provides the DECnet protocol output octet count.

Syntax: Counter

Access: Read-only

locIfdecnetOutPkts

Provides the DECnet protocol output packet count.

Syntax: Counter

Access: Read-only

HP Probe

The following variables in the Interface group apply to interfaces running HP Probe, an address resolution protocol developed by Hewlett-Packard:

locIfprobeInOctets

Provides the HP Probe protocol input octet count.

Syntax: Counter

Access: Read-only

locIfprobeInPkts

Provides the HP Probe protocol input packet count.

Syntax: Counter

Access: Read-only

locIfprobeOutOctets

Provides the HP Probe protocol output octet count.

Syntax: Counter

Access: Read-only

locIfprobeOutPkts

Provides the HP Probe protocol output packet count.

Syntax: Counter

Access: Read-only

Internet Protocol (IP)

The following variables in the Interface group apply to interfaces running the Internet Protocol (IP):

locIfipInOctets

Provides the IP input octet count.

Syntax: Counter

Access: Read-only

locIfipInPkts

Provides the IP input packet count.

Syntax: Counter

Access: Read-only

locIfipOutOctets

Provides the IP output octet count.

Syntax: Counter

Access: Read-only

locIfipOutPkts

Provides the IP output packet count.

Syntax: Counter

Access: Read-only

LAN Network Manager (LNM)

The following variables in the Interface group apply to interfaces running the LAN Network Manager (LNM) protocol. This protocol manages source-route bridging (SRB) networks.

locIflanmanInOctets

Provides the LAN Network Manager protocol input octet count.

Syntax: Counter

Access: Read-only

locIflanmanInPkts

Provides the LAN Network Manager protocol input packet count.

Syntax: Counter

Access: Read-only

locIflanmanOutOctets

Provides the LAN Network Manager protocol output octet count.

Syntax: Counter

Access: Read-only

locIflanmanOutPkts

Provides the LAN Network Manager protocol output packet count.

Syntax: Counter

Access: Read-only

Maintenance Operation Protocol (MOP)

The following variables in the Interface group apply to interfaces running the Maintenance Operation Protocol (MOP):

locIfmopInOctets

Provides the MOP input octet count.

Syntax: Counter

Access: Read-only

locIfmopInPkts

Provides the MOP input packet count.

Syntax: Counter

Access: Read-only

locIfmopOutOctets

Provides the MOP output octet count.

Syntax: Counter

Access: Read-only

locIfmopOutPkts

Provides the MOP output packet count.

Syntax: Counter

Access: Read-only

Novell

The following variables in the Interface group apply to interfaces running Novell:

locIfnovellInOctets

Provides the Novell protocol input octet count.

Syntax: Counter

Access: Read-only

locIfnovellInPkts

Provides the Novell protocol input packet count.

Syntax: Counter

Access: Read-only

locIfnovellOutOctets

Provides the Novell protocol output octet count.

Syntax: Counter

Access: Read-only

locIfnovellOutPkts

Provides the Novell protocol output packet count.

Syntax: Counter

Access: Read-only

Other Protocols

The following variables in the Interface group record the number of input and output packets and octets for interfaces running protocols other than those listed in the Interface group.

locIfotherInOctets

Provides the input octet count for protocols other than those listed in the Interface group.

Syntax: Counter

Access: Read-only

locIfotherInPkts

Provides the input packet count for protocols other than those listed in the Interface group.

Syntax: Counter

Access: Read-only

locIfotherOutOctets

Provides the output octet count for protocols other than those listed in the Interface group.

Syntax: Counter

Access: Read-only

locIfotherOutPkts

Provides the output packet count for protocols other than those listed in the Interface group.

Syntax: Counter

Access: Read-only

Serial Tunnel (STUN)

The following variables in the Interface group apply to interfaces using the Serial Tunnel (STUN) protocol. STUN allows devices that use Synchronous Data Link Control (SDLC) or High-Level Data Link Control (HDLC) to be connected through one or more Cisco routers across different network topologies.

locIfstunInOctets

Provides the STUN protocol input octet count.

Syntax: Counter

Access: Read-only

locIfstunInPkts

Provides the STUN protocol input packet count.

Syntax: Counter

Access: Read-only

locIfstunOutOctets

Provides the STUN protocol output octet count.

Syntax: Counter

Access: Read-only

locIfstunOutPkts

Provides the STUN protocol output packet count.

Syntax: Counter

Access: Read-only

Spanning Tree

The following variables in the Interface group apply to interfaces running the Spanning Tree protocol. Used in bridging, spanning trees provide root and designated bridges to notify all other bridges in the network when an address change has occurred, thereby eliminating loops.

locIfspanInOctets

Provides the spanning-tree input octet packet count.

Syntax: Counter

Access: Read-only

locIfspanInPkts

Provides the spanning-tree input protocol packet count.

Syntax: Counter

Access: Read-only

locIfspanOutOctets

Provides the spanning-tree output octet packet count.

Syntax: Counter

Access: Read-only

locIfspanOutPkts

Provides the spanning-tree output protocol packet count.

Syntax: Counter

Access: Read-only

Banyan Virtual Network System (VINES)

The following variables in the Interface group apply to interfaces running the Banyan Virtual Network System (VINES) protocol. This proprietary protocol is derived from the Xerox Network Systems (XNS) protocol. The VINES variables provide the number of input and output packets and octets on a per interface basis.

locIfvinesInOctets

Provides the VINES protocol input octet count.

Syntax: Counter

Access: Read-only

locIfvinesInPkts

Provides the VINES protocol input packet count.

Syntax: Counter

Access: Read-only

locIfvinesOutOctets

Provides the VINES protocol output octet count.

Syntax: Counter

Access: Read-only

locIfvinesOutPkts

Provides the VINES protocol output packet count.

Syntax: Counter

Access: Read-only

Xerox Network Systems (XNS)

The following variables in the Interface group apply to interfaces running XNS.

locIfxnsInOctets

Provides the XNS protocol input octet count.

Syntax: Counter

Access: Read-only

locIfxnsInPkts

Provides the XNS input packet count.

Syntax: Counter

Access: Read-only

locIfxnsOutOctets

Provides the XNS protocol output octet count.

Syntax: Counter

Access: Read-only

locIfxnsOutPkts

Provides the XNS protocol output packet count.

Syntax: Counter

Access: Read-only

Internet Protocol (IP) Group

The Internet Protocol (IP) group provides variables pertaining to the IP, such as the determination of how an interface obtained its IP address, who supplied the address, and Internet Control Message Protocol (ICMP) messages about IP packet processing.

IP Address Table

The Cisco IP address table, *lipAddrTable,* contains the following six variable entries, or rows:*locIPHelper*, *locIPHow*, *locIPRedirects*, *locIPSecurity,* and *locIPUnreach*, *locIPWho.* The index to this table is the IP address of the device, or *ipAdEntAddr.* If a device has *n* number of IP addresses, there will be *n* rows in the table.

For simplification, Table 6 shows only the *locIpHow* and *locIPWho* variables. The *locIPHow* variable value shows that the device at 131.108.201.245 obtained its address through nonvolatile memory. The *locIPWho* variable value indicates the device was assigned its current address by the device at 131.101.200.248.

Table 6 IP Address

IP Address	locIPHow	locIPWho	and so on
131.108.201.245	nonvolatile	131.101.200.248	
142.111.202.244	nonvolatile	131.56.70.249	
and so on			

locIPHelper

Provides the IP address for broadcast forwarding support. Provides the destination broadcast or IP address that the router should use when forwarding User Datagram Protocol (UDP) broadcast datagrams, including BootP, received on the interface.

Syntax: Network address

Access: Read-only

locIPHow

Describes how this interface obtained its IP address. Typically, the address is determined by nonvolatile memory.

Syntax: Display string

Access: Read-only

locIPRedirects

Indicates whether Internet Control Message Protocol (ICMP) redirects will be sent. A router sends an ICMP Redirect message to the originator of any datagram that it is forced to resend through the same interface on which it was received. It does so because the originating host presumably could have sent that datagram to the ultimate destination without involving the router at all. ICMP Redirect messages are sent only if the router is configured with the **ip redirects** command.

Syntax: Integer (1 = sent, 2 = not sent)

Access: Read-only

locIPSecurity

Indicates whether IP security is enabled on the interface. For details on IP security levels, see RFC 1108, *U.S. Department of Defense Security Options for the Internet Protocol*.

Syntax: Integer (0 = false, 1 = true)

Access: Read-only

locIPUnreach

Indicates whether Internet Control Message Protocol (ICMP) packets indicating unreachable addresses will be sent for a specific route.

If this variable is set, and the router receives a datagram that it cannot deliver to its ultimate datagram (because it knows of no route to the destination address), it replies to the originator of that datagram with an ICMP Host Unreachable message.

Syntax: Integer (0 = false, 1 = true)

Access: Read-only

locIPWho

Provides the IP address of the device from which this interface received its IP address. If the interface does not use an IP address from another device, a value of 0.0.0.0 is displayed.

Syntax: Network address

Access: Read-only

End of Table

IP Routing Table

The local IP routing table, *lipRoutingTable*, contains the following three variables: *locRtCount, locRtMask*, and *locRtUses*. The index for this table is the destination address of the IP route, or *ipRouteDest.* If there are *n* number of routes available to a device, there will be *n* rows in the IP routing table.

In Table 7, for the route with the destination IP address of 131.104.111.1, the routing table network mask is 255.255.255.0. The number of parallel routes within the routing table is 3, and the route was used in a forwarding operation two times.

Table 7 IP Routing Table

ipRouteDest	locRtMask	locRtCount	locRtUses
131.104.111.1	255.255.255.0	3	2
133.45.244.245	255.255.255.0	1	1

locRtCount

Provides the number of parallel routes within the routing table.

Syntax: Integer

Access: Read-only

locRtMask

Provides the routing table network mask. For example, 255.255.255.0.

Syntax: Network address

Access: Read-only

locRtUses

Provides the number of times the route was used in a forward operation.

Syntax: Integer

Access: Read-only

End of Table

actLostByts

Provides the total number of bytes of lost IP packets as a result of accounting failure.

Syntax: Integer

Access: Read-only

actLostPkts

Provides the number of IP packets that were lost due to memory limitations and accounting failure.

Syntax: Integer

Access: Read-only

actThresh

Provides the threshold of IP accounting records in use before IP traffic will be discarded.

Syntax: Integer

Access: Read-only

IP Accounting Group

The Cisco router maintains two accounting databases: an active database and a checkpoint database. The router takes a snapshot of the running, or active database, and copies it into the checkpoint database. For detailed information on active and checkpoint databases, refer to the *Router Products Configuration and Reference* and *Router Products Command Reference* publications.

This group provides access to the active database that is created and maintained if IP accounting is enabled on a router. The active database contains information about the number of bytes and packets switched through a system on a source and destination IP address basis. Only

transit IP traffic is measured and only on an outbound basis; traffic generated by the router or terminating in the router is not included in the accounting statistics. Internetwork statistics obtained through these variables can be analyzed to improve network performance.

The local IP accounting table, *lipAccountingTable*, includes the following four related variables: *actByts, actDst, actPkts,* and *actSrc*. The index for this table is *actSrc* and *actDst*. For example, in the first row in Table 8, the source host address is 131.24.35.248, and the destination host address is 138.32.28.245. Fifty IP packets and 400 bytes of data have been sent between the source and destination address.

Table 8 Local IP Accounting Table

actByts	actDst	actPkts	actSrc
400	138.32.28.245	50	131.24.35.248
1259	128.52.33.101	110	128.52.33.96

actByts

Provides the total number of bytes in IP packets from the source to destination host.

Syntax: Integer

Access: Read-only

actDst

Provides the IP destination address for the host traffic matrix.

Syntax: Network Address

Access: Read-only

actPkts

Provides the number of IP packets sent from the source to destination host.

Syntax: Integer

Access: Read-only

actSrc

Provides the IP address for the host traffic matrix.

Syntax: Network address

Access: Read-only

End of Table

actAge

Provides the age of the accounting data in the current data matrix of the active database.

Syntax: Timeticks

Access: Read-only

IP Checkpoint Accounting Group

The Cisco router maintains two accounting databases: an active database and a checkpoint database. The router takes a snapshot of the running, or active database, and copies it into the checkpoint database. For detailed information on active and checkpoint databases, refer to the *Router Products Configuration and Reference* publication.

If the checkpoint database already had data obtained previously from the active database, the router appends the latest copy of the active database to the existing data in the checkpoint database. The checkpoint database stores data retrieved from the active database until you delete the contents of this database by using the **clear ip accounting [checkpoint]** command.

A Network Management System (NMS) can use checkpoint MIB variables to analyze stable data in the checkpoint database.

IP Checkpoint Accounting Table

The local IP checkpoint accounting table, *lipCkAccountingTable*, includes the following four related variables: *ckactByts*, *ckactDst*, *ckactPkts*, and *ckactSrc*. The index for this table is *ckacSrc* and *ckactDst*. For example, on page 80, the source host address is 131.24.35.248. The destination host address is 138.32.28.245. Fifty IP packets and 400 bytes of data have been sent between the source and destination address.

Table 9 IP Checkpoint Accounting

ckactByts	ckactDst	ckactPkts	ckacSrc
400	138.32.28.245	50	131.24.35.248
480	124.45.222.246	60	123.34.216.244

ckactByts

Provides the total number of bytes in IP packets from source to destination in the checkpoint matrix.

Syntax: Integer

Access: Read-only

ckactDst

Provides the IP destination address of the host receiving the IP packets. The address is listed in the checkpoint traffic matrix.

Syntax: Network address

Access: Read-only

ckactPkts

Provides the number of IP packets sent from the source to the destination address in the checkpoint matrix.

Syntax: Integer

Access: Read-only

ckactSrc

Provides the IP source address of the host sending the IP packets. The address is listed in the checkpoint traffic matrix.

Syntax: IP address

Access: Read-only

End of Table

actCheckPoint

Activates a checkpoint database. This variable must be read and then set to the same value that was read. The value read and then set will be incremented after a successful set request.

For detailed information on active and checkpoint databases, refer to the *Router Products Command Reference* and *Router Products Configuration and Reference* publications.

Syntax: Integer

Access: Read-write

ckactAge

Provides information on how long ago the data was first stored in the checkpoint matrix.

Syntax: Timeticks

Access: Read-only

ipNoaccess

Provides the total number of packets dropped due to access control failure.

Syntax: Counter

Access: Read-only

System Group

The variables described in this section are system-wide and apply to all Cisco Systems products.

Basic

The following variables pertain to basic information such as system software description and version number, host and domain names, and number of bytes of free memory in the managed device.

authAddr

Provides the IP address of the device causing the last SNMP authorization failure. The device did not use a configured community string.

Syntax: IP address

Access: Read-only

bootHost

Provides the IP address of the host that supplied the software currently running on the managed device.

Syntax: IP address

Access: Read-only

domainName

Provides the domain portion of the domain name of the host.

Syntax: Display string

Access: Read-only

freeMem

Provides the number of bytes of free memory available in the managed device.

Syntax: Integer

Access: Read-only

hostName

Represents the name of the host in printable ASCII characters.

Syntax: Display string

Access: Read-only

ping

Setting the packet internet groper (ping) variable results in an echo request packet (also known as *ICMP Echo Request*) being sent to an address and then awaiting an ICMP Echo reply. Results from this echo protocol can help in evaluating the path-to-host reliability, delays over the path, and whether the host can be reached or is functioning. The ping variable can be used on interfaces running the IP or AppleTalk protocols.

The instance identifier for the ping object is *protocol. address.count.size.timeout*, where the protocol can be either Protocol—1 for IP or 2 for AppleTalk. The address is an IP address or an AppleTalk address.

Count—A decimal number between 1 and 2147483647. The count is the number of echo requests to send.

Size—A decimal number between 32 and 18024. The size is the bytes in each echo request packet.

Timeout—A decimal number between 0 and 3600. Timeout is the number of seconds to wait for a reply to each echo request.

Syntax: Integer

Access: Read-write

romId

Contains a printable octet string that contains the system bootstrap description and version identification.

Syntax: Display string

Access: Read-only

whyReload

Contains a printable octet string that contains the reason why the system was last restarted.

Syntax: Display string

Access: Read-only

Buffer

The following variables are used to monitor the amount and type of buffer space available within a managed device. *Buffers* are blocks of memory used to hold network packets. There are five types of buffers based on size: *small*, *middle*, *big*, *large*, and *huge*. There are several pools of different-sized buffers. These pools grow and shrink based upon demand. Some buffers are temporary and are created and destroyed as warranted. Others are permanently allocated.

bufferFail

Contains the total number of allocation requests that have failed due to lack of any free buffers.

Syntax: Integer

Access: Read-only

bufferNoMem

Counts the number of failures due to a lack of memory to create a new buffer.

Syntax: Integer

Access: Read-only

Buffer Elements

Buffer elements are blocks of memory used in internal operating system queues.

bufferElCreate

Contains the number of new buffer elements created for the managed device.

Syntax: Integer

Access: Read-only

bufferElFree

Contains the number of buffer elements that are not currently allocated and are available for use in the managed device.

Syntax: Integer

Access: Read-only

bufferElHit

Contains the number of successful attempts to allocate a buffer element when needed.

Syntax: Integer

Access: Read-only

bufferElMax

Contains the maximum number of buffer elements the managed device can have.

Syntax: Integer

Access: Read-only

bufferElMiss

Contains the number of allocation attempts that failed because there were no buffer elements available.

Syntax: Integer

Access: Read-only

Small Buffers

Small buffer sizes are configurable.

bufferSmCreate

Contains the number of small buffers created in the managed device.

Syntax: Integer

Access: Read-only

bufferSmFree

Contains the number of small buffers that are currently available to the managed device.

Syntax: Integer

Access: Read-only

bufferSmHit

Contains the number of successful attempts to allocate a small buffer when needed.

Syntax: Integer

Access: Read-only

bufferSmMax

Contains the maximum number of small buffers that can be allocated to the managed device.

Syntax: Integer

Access: Read-only

bufferSmMiss

Contains the number of allocation attempts that failed because there were no small buffers available.

Syntax: Integer

Access: Read-only

bufferSmSize

Provides the size (in bytes) of small buffers.

Syntax: Integer

Access: Read-only

bufferSmTotal

Provides the total number of small buffers allocated to the managed device.

Syntax: Integer

Access: Read-only

bufferSmTrim

Contains the small buffers that have been destroyed in the managed device.

Syntax: Integer

Access: Read-only

Middle Buffers

Middle buffer sizes are configurable.

bufferMdCreate

Contains the number of middle buffers created in the managed device.

Syntax: Integer

Access: Read-only

bufferMdFree

Contains the number of middle buffers that are currently available to the managed device.

Syntax: Integer

Access: Read-only

bufferMdHit

Contains the number of successful attempts to allocate a middle buffer when needed.

Syntax: Integer

Access: Read-only

bufferMdMax

Contains the maximum number of middle buffers that can be allocated to the managed device.

Syntax: Integer

Access: Read-only

bufferMdMiss

Contains the number of allocation attempts that failed because there were no middle buffers available.

Syntax: Integer

Access: Read-only

bufferMdSize

Provides the size (in bytes) of middle buffers.

Syntax: Integer

Access: Read-only

bufferMdTotal

Provides the total number of middle buffers allocated to the managed device.

Syntax: Integer

Access: Read-only

bufferMdTrim

Contains the middle buffers that have been destroyed in the managed device.

Syntax: Integer

Access: Read-only

Big Buffers

Big buffer sizes are configurable.

bufferBgCreate

Contains the number of big buffers created in the managed device.

Syntax: Integer

Access: Read-only

bufferBgFree

Contains the number of big buffers that are currently available to the managed device.

Syntax: Integer

Access: Read-only

bufferBgHit

Contains the number of successful attempts to allocate a big buffer when needed.

Syntax: Integer

Access: Read-only

bufferBgMax

Contains the maximum number of big buffers that can be allocated to the managed device.

Syntax: Integer

Access: Read-only

bufferBgMiss

Contains the number of allocation attempts that failed because there were no big buffers available.

Syntax: Integer

Access: Read-only

bufferBgSize

Provides the size (in bytes) of big buffers.

Syntax: Integer

Access: Read-only

bufferBgTotal

Provides the total number of big buffers allocated to the managed device.

Syntax: Integer

Access: Read-only

bufferBgTrim

Contains the big buffers that have been destroyed in the managed device.

Syntax: Integer

Access: Read-only

Large Buffers

Large buffer sizes are configurable.

bufferLgCreate

Contains the number of large buffers created in the managed device.

Syntax: Integer

Access: Read-only

bufferLgFree

Contains the number of large buffers that are currently available to the managed device.

Syntax: Integer

Access: Read-only

bufferLgHit

Contains the number of successful attempts to allocate a large buffer when needed.

Syntax: Integer

Access: Read-only

bufferLgMax

Contains the maximum number of large buffers that can be allocated to the managed device.

Syntax: Integer

Access: Read-only

bufferLgMiss

Contains the number of allocation attempts that failed because there were no large buffers available.

Syntax: Integer

Access: Read-only

bufferLgSize

Provides the size (in bytes) of large buffers.

Syntax: Integer

Access: Read-only

bufferLgTotal

Provides the total number of large buffers allocated to the managed device.

Syntax: Integer

Access: Read-only

bufferLgTrim

Contains the large buffers that have been destroyed in the managed device.

Syntax: Integer

Access: Read-only

Huge Buffers

Huge buffer sizes are configurable.

bufferHgCreate

Contains the number of huge buffers created in the managed device.

Syntax: Integer

Access: Read-only

bufferHgFree

Contains the number of huge buffers that are currently available to the managed device.

Syntax: Integer

Access: Read-only

bufferHgHit

Contains the number of successful attempts to allocate a huge buffer when needed.

Syntax: Integer

Access: Read-only

bufferHgMax

Contains the maximum number of huge buffers that can be allocated to the managed device.

Syntax: Integer

Access: Read-only

bufferHgMiss

Contains the number of allocation attempts that failed because there were no huge buffers available.

Syntax: Integer

Access: Read-only

bufferHgSize

Provides the size (in bytes) of huge buffers.

Syntax: Integer

Access: Read-only

bufferHgTotal

Provides the total number of huge buffers allocated to the managed device.

Syntax: Integer

Access: Read-only

bufferHgTrim

Contains the huge buffers that have been destroyed in the managed device.

Syntax: Integer

Access: Read-only

CPU Utilization

The following variables provide statistics on the CPU utilization of a device:

avgBusy1

Provides a cumulative average of the CPU usage percentage over a 1-minute period.

Syntax: Integer

Access: Read-only

avgBusy5

Provides a cumulative average of the CPU usage percentage over a 5-minute period.

Syntax: Integer

Access: Read-only

avgBusyPer

Provides the percentage of CPU usage over the first 5-second period in the scheduler. The scheduler determines which process or task takes priority over another and triggers them accordingly.

Syntax: Integer

Access: Read-only

ciscoContactInfo

Provides the Cisco name and address for reference purposes. This MIB variable applies only to router products that were purchased from Cisco.

Syntax: Display string

Access: Read-only

Environmental Monitor Card

The environmental monitor card is provided only with the Cisco AGS+ router. This card checks input air temperature and air flow through the system card cage and card cage backplane power supplies. It also provides nonvolatile and system bus memory for the system. The Cisco 7000 has built-in environmental-monitoring functionality, and so does not use the card. The Cisco 7000 router provides environmental monitoring, reporting, and if necessary, system shutdown.

All MIB variables in this group apply to the Cisco AGS+. A subset of those variables apply to the Cisco 7000. The following variables are used to poll and display power supply voltage and air temperature (in Celsius) in an AGS+ to help prevent system problems.

envBurnDate

Provides the date of the calibration of the environmental monitor card. (AGS+ only)

For example:

```
calibrated on 2-14-93.
```

Syntax: Display string

Access: Read-only

envFirmVersion

Provides the firmware level of the environmental monitor card. (AGS+ only)

For example:

```
Environmental controller firmware version 2.0
```

Syntax: Display string

Access: Read-only

envPresent

Indicates whether there is an environmental monitor card in a router.

Syntax: Integer (0 = no, 1 = yes, but unavailable to SNMP, 2 = yes and available to SNMP for AGS+ routers, 3 = yes and available to SNMP for Cisco 7000 routers)

Access: Read-only

envSerialNumber

Provides the serial number of the environmental monitor card. (AGS+ only)

Following is an example of a serial number:

```
00220846
```

Syntax: Display string

Access: Read-only

envTechnicianID

Provides the technician ID for the environmental monitor card. (AGS+ only)

Following is an example of a technician ID:

```
rma
```

Syntax: Display string

Access: Read-only

envTestPt1Descr

Test point 1 is the temperature of air entering the AGS+ and the Cisco 7000 router. (AGS+ and Cisco 7000)

Syntax: Display string

Access: Read-only

envTestPt1last

Provides the temperature of air entering the AGS+ and the Cisco 7000 router when the last shutdown occurred. If the input air temperature exceeds 109°F (43°C) in an AGS+, an error is detected, and the CSC-ENVM card shuts down the power supply.

Syntax: Integer

Access: Read-only

envTestPt1MarginVal

Provides warning and fatal threshold values of the internal intake air for the AGS+ router and Cisco 7000.

Syntax: Integer

Access: Read-only

envTestPt1Measure

Provides the current temperature of air entering the router. (AGS+ and Cisco 7000)

Syntax: Display string

Access: Read-only

envTestPt1warn

Indicates whether the air temperature entering the router is at warning level. (AGS+ and Cisco 7000)

Syntax: Integer (1 = warning, 2 = no warning)

Access: Read-only

envTestPt2Descr

Test point 2 is the temperature of air leaving the router. (AGS+ and Cisco 7000)

Syntax: Display string

Access: Read-only

envTestPt2last

Provides the temperature of air leaving the router when the last shutdown occurred. (AGS+ and Cisco 7000)

Syntax: Integer

Access: Read-only

envTestPt2MarginVal

Provides the fatal threshold value for the exhaust air flow of the router. (AGS+ and Cisco 7000)

Syntax: Integer

Access: Read-only

envTestPt2Measure

Provides the temperature of the exhaust air flow of the router. (AGS+ and Cisco 7000)

Syntax: Integer

Access: Read-only

envTestPt2warn

Indicates whether the temperature of air flow leaving the router is at a warning level. (AGS+ and Cisco 7000)

Syntax: Integer (1 = warning, 2 = no warning)

Access: Read-only

envTestPt3Descr

Test point 3 is the +5-volt (V) line on the router.

Syntax: Display string

Access: Read-only

envTestPt3last

Provides the value of the +5V line when the last shutdown occurred. (AGS+ and Cisco 7000)

Syntax: Integer

Access: Read-only

envTestPt3MarginPercent

Provides the warning and fatal thresholds for the +5V line to the power supply on the AGS+ router. The warning threshold is 5 percent above or below +5V. The fatal threshold at which the router shuts down is 10 percent above or below +5V. (AGS+ only)

Syntax: Integer

Access: Read-only

envTestPt3Measure

Provides the current value for the +5V line to the power supply on the router. The value is expressed in millivolts. (AGS+ and Cisco 7000)

Syntax: Integer

Access: Read-only

envTestPt3warn

Indicates whether the +5V line to the power supply is at warning level. The warning threshold is 5 percent above or below +5V. (AGS+ and Cisco 7000)

Syntax: Integer (1 = warning, 2 = no warning)

Access: Read-only

envTestPt4Descr

Test point 4 is the +12V line to the power supply of the router.

Syntax: Display string

Access: Read-only

envTestPt4last

Provides the value of the +12V line when the last shutdown occurred.

Syntax: Integer

Access: Read-only

envTestPt4MarginPercent

Provides the warning and fatal thresholds for the +12V line to the power supply on the AGS+ router. The warning threshold is 10 percent above or below +12V. The fatal threshold at which the router shuts down is 15 percent above or below +12V. (AGS+ only)

Syntax: Integer

Access: Read-only

envTestPt4Measure

Provides the current value (in millivolts) of the +12V line to the power supply of the router.

Syntax: Integer

Access: Read-only

envTestPt4warn

Indicates whether the +12V line to the power supply is at warning level. The warning threshold is 10 percent above or below +12V.

Syntax: Integer (1 = warning, 2 = no warning)

Access: Read-only

envTestPt5Descr

Test point 5 is the –12V line to the power supply of the router.

Syntax: Display string

Access: Read-only

envTestPt5last

Provides the value of the –12V line when the last shutdown occurred.

Syntax: Integer

Access: Read-only

envTestPt5MarginPercent

Provides the warning and fatal thresholds for the –12V line to the power supply on the AGS+ router. The warning threshold is 10 percent above or below –12V. The fatal threshold at which the router shuts down is 15 percent above or below –12V. (AGS+ only)

Syntax: Integer

Access: Read-only

envTestPt5Measure

Provides the current value (in millivolts) of the –12V line to the power supply of the router.

Syntax: Integer

Access: Read-only

envTestPt5warn

Indicates whether the –12V line to the power supply on the AGS+ router is at warning level. The warning threshold is 10 percent above or below –12V.

Syntax: Integer (1 = warning, 2 = no warning)

Access: Read-only

envTestPt6Descr

Test point 6 is the –5V line to the power supply of the AGS+ router and +24V line to the power supply of the Cisco 7000 router.

Syntax: Display string

Access: Read-only

envTestPt6last

Provides the value of the –5V line to the power supply of the AGS+ router and +24V line to the power supply of the Cisco 7000 router when the last shutdown occurred.

Syntax: Integer

Access: Read-only

envTestPt6MarginPercent

Provides the warning and fatal thresholds for the –12V line to the power supply on the AGS+ router. The warning threshold is 5 percent above or below –5V. The fatal threshold at which the router shuts down is 10 percent above or below –5V. (AGS+ only)

Syntax: Integer

Access: Read-only

envTestPt6Measure

Provides the current value (in millivolts) of the –5V line to the power supply of the AGS+ router and +24V line to the power supply of the Cisco 7000 router.

Syntax: Integer

Access: Read-onlyenvTestPt6warn

Indicates whether the –5V line to the power supply on the AGS+ router is at warning level. The warning threshold is 5 percent above or below –5V.

For the Cisco 7000, this variable indicates whether the +P24V line to the power supply is at warning level.

Syntax: Integer (1 = warning, 2 = no warning)

Access: Read-only

envType

Provides the type of environmental card (for example, CSC-ENVM).

Syntax: Display string

Access: Read-only

Host Configuration File

The following variables are used to monitor and set host configuration file information:

hostConfigAddr

Provides the address of the host that provided the host configuration file for a specific device. The *host configuration file* contains commands that apply to one network server in particular.

Syntax: Network address

Access: Read-only

hostConfigName

Provides the name of the last host configuration file used by the device.

Syntax: Display string

Access: Read-only

hostConfigProto

Provides the protocol that supplied the host configuration file.

Syntax: Integer (1 = IP, 2 = MOP, 3 = not applicable)

Access: Read-only

hostConfigSet

Allows the Network Management System (NMS) to load a new host configuration file via Trivial File Transfer Protocol (TFTP) onto the managed device and indicate the name of this configuration file. The instance ID is the IP address of the TFTP host. The display string indicates the name of the configuration file.

Syntax: Display string

Access: Write-only

Network Configuration File

The following variables are used to monitor and remotely set network configuration file information for the device:

netConfigAddr

Provides the address of the host that supplied the network configuration file for the managed device. The *network configuration file* contains commands that apply to all network servers and terminal servers on a network.

Syntax: Network address

Access: Read-only

netConfigName

Provides the name of the network configuration file that resides on the managed device.

Syntax: Display string

Access: Read-only

netConfigProto

Provides the protocol that supplied the network configuration file.

Syntax: Integer

Access: Read-only

netConfigSet

Loads a new network configuration file via Trivial File Transfer Protocol (TFTP) onto the managed device and indicates the name of this configuration file. The instance ID is the IP address of the TFTP host. The display string indicates the name of the configuration file.

Syntax: Display string

Access: Write-only

System Configuration

The following variables are used to monitor and set system-wide parameters:

sysClearARP

Performs a clearing of the entire Address Resolution Protocol (ARP) cache and Internet Protocol (IP) route cache. The ARP provides dynamic mapping between IP addresses and Ethernet addresses. The ARP cache table keeps a record of these mappings and can be cleared for maintenance purposes.

The IP route cache controls the use of a high-speed switching cache for IP routing. The route cache is enabled by default and allows outgoing packets to be load balanced on a per-destination basis. The *sysClearARP* variable helps clear the IP route cache for maintenance purposes.

Syntax: Integer

Access: Write-only

sysClearInt

Clears an interface that is given *IfIndex* as a value. To clear an interface, take the *ifIndex* for the interface (for example, a value of 4) and set the *sysClearInt* variable to the *ifIndex* value of 4.

Syntax: Integer

Access: Write-only

sysConfigAddr

Provides the address of the host that supplied the system boot image for the managed device. New versions of software can be downloaded over the network with boot image files. The new file takes effect the next time the managed device is reloaded.

Syntax: Network address

Access: Read-only

sysConfigName

Provides the name of the system boot image file. New versions of software can be downloaded over the network with boot image files. The new file takes effect the next time the managed device is reloaded.

Syntax: Display string

Access: Read-only

sysConfigProto

Provides the protocol type that supplied the system boot image.

Syntax: Integer

Access: Read-only

writeMem

Writes the current (running) router configuration into nonvolatile memory where it can be stored and retained even if the router is reloaded. Erase configuration memory if 0.

Syntax: Integer

Access: Write-only

writeNet

Sends a copy of the current configuration via Trivial File Transfer Protocol (TFTP) to a remote host. When it is stored on the host, the configuration file can be edited and retrieved by other network entities.

Syntax: Display string

Access: Write-only

Terminal Server Group

Following are variables that can be applied to terminal servers. This group contains terminal-server-specific information on a per-line basis, such as line status, line type, line speed, type of flow control, and type of modem.

Note This section describes the terminal server functionality of the router and communication server.

tsLine

Number of physical lines on the device.

Syntax: Integer

Access: Read-only

Terminal Server Line Table

The local terminal server line table, *ltsLineTable*, contains all of the variables described in this section. The index to this table is the number of the terminal server line. If there are *n* number of terminal lines associated with the device, there will be *n* rows in the table.

Table 10 Terminal Server Line

Line Number	tsLineActive	tsLineType	and so on
1	Contains all of the variables described in this section.		
2			
and so on			

tsLineActive

Indicates whether this line is active.

Syntax: Integer (1 = active, 2 = not active)

Access: Read-only

tsLineAutobaud

Indicates whether the line is set to autobaud detection so that it can adapt to the rate at which data is being sent to it.

Syntax: Integer (1 = autobaud, 2 = not autobaud)

Access: Read-only

tsLineEsc

Indicates what is used to represent the escape (Esc) character. The escape character allows a user to break out of active sessions.

Syntax: Display string

Access: Read-only

tsLineFlow

Indicates the type of flow control the line is using. The flow can be controlled from software or hardware. Input indicates that the flow control is coming from the device to the terminal server. Output indicates flow control is provided by the terminal server.

Syntax: Integer

Access: Read-only

The possible integer values follow:

1 = unknown
2 = none
3 = software-input
4 = software-output
5 = software-both
6 = hardware-input
7 = hardware-output
8 = hardware-both

tsLineLoc

Describes the physical location of the line.

Syntax: Display string

Access: Read-onlytsLineModem

Describes the type of modem control the line is using.

Syntax: Integer

Access: Read-only

The possible integer values follow:

1 = unknown
2 = none
3 = call-in
4 = call-out
5 = cts-required
6 = rs-is-cd

The integer values mean the following:

Call-in indicates dial-in modems that use the status of Data Terminal Ready (DTR) to determine whether or not answer an incoming call.

Call-out indicates modems that raise data terminal ready (DTR) to see if Clear To Send (CTS) becomes high as an indication that the host has noticed its signal.

Cts-required indicates the form of modem control that requires CTS to be high throughout the use of the line.

rs-is-cd is used for lines with high-speed modems. The modem answers the call if DTR is high, uses its Carrier Detect (CD) signal to reflect the carrier presence, and has its CD signal wired to the ring input of the terminal server.

tsLineNoise

Provides the number of garbage characters received while the line is inactive.

Syntax: Integer

Access: Read-only

tsLineNses

Indicates the number of current sessions on the line.

Syntax: Integer

Access: Read-only

tsLineRotary

Specifies the number of the rotary group to which the line belongs. If the first line in a rotary group is busy, a connection can be made to the next free line.

Syntax: Integer

Access: Read-only

tsLineScrlen

Provides the length (in lines) of the screen of the terminal attached to the line.

Syntax: Integer

Access: Read-only

tsLineScrwid

Provides the width (in characters) of the screen of the terminal attached to the line.

Syntax: Integer

Access: Read-only

tsLineSestmo

Specifies the interval (in seconds) for closing the connection when there is no input or output traffic during a session.

Syntax: Integer

Access: Read-only

tsLineSpeedin

Indicates the input speed at which the line is running.

Syntax: Integer

Access: Read-only

tsLineSpeedout

Indicates the output speed at which the line is running.

Syntax: Integer

Access: Read-only

tsLineTerm

Describes the terminal type of the line.

Syntax: Display string

Access: Read-only

tsLineTmo

Specifies the interval (in seconds) for closing the connection when there is no input or output traffic on the line.

Syntax: Integer

Access: Read-only

tsLineType

Describes the terminal line type.

Syntax: Integer

Access: Read-only

The possible integer values follow:

1 = unknown
2 = console
3 = terminal
4 = line-printer
5 = virtual-terminal
6 = auxiliary

tsLineUser

Provides Terminal Access Controller Access System (TACACS) username and indicates whether or not TACACS is enabled on this line. TACACS servers provide security for accessing terminals remotely.

Syntax: Display string

Access: Read-only

tsLineUses

Indicates the number of times a connection has been made to or from this line.

Syntax: Integer

Access: Read-only

End of Table

Terminal Server Line Session Table

The terminal server line session table, *ltsLineSessionTable*, contains the following six variables: *tslineSesAddr, tslineSesCur, tslineSesDir, tslineSesIdle, tslineSesName,* and *tslineSesType.*

For simplification, Table 11 shows values for three of the variables contained in the terminal server line session table. The index to the table is the session number and line number. Line 1 in the first session illustrates a Telnet connection. The session was started by the terminal. The remote host for this session is located at the IP address of 131.38.141.244.

Table 11 Terminal Server Line Session

Session no. Line no.	tslineSesAddr	tslineSesDir	tslineSesType
1, 1	131.38.141.244	3	5
2, 4	138.121.128.243	2	3

tslineSesAddr

Provides the address of the remote host for this session.

Syntax: Network address

Access: Read-only

tslineSesCur

Indicates whether this session is currently active.

Syntax: Integer (1 = active, 2 = not active)

Access: Read-only

tslineSesDir

Indicates whether this session was started by another device (incoming) or by the terminal (outgoing).

Syntax: Integer

Access: Read-only

The possible integer values follow:

1 = unknown
2 = incoming
3 = outgoing

tslineSesIdle

Indicates the amount of time (in seconds) that this session has been idle.

Syntax: Integer

Access: Read-only

tslineSesName

Provides the name of the remote host for this session.

Syntax: Display string

Access: Read-only

tslineSesType

Describes the type of session that is currently active.

Syntax: Integer

Access: Read-only

The possible integer values follow:

1 = unknown
2 = X.3 Packet Assembler/Disassembler (PAD)
3 = stream (enables a raw TCP stream with no Telnet-control sequences)
4 = rlogin (for making remote connection to a host—part of TCP/IP)
5 = telnet (for making remote connection to a host)—UNIX protocol)
6 = Transmission Control Protocol (TCP)
7 = local area transport (LAT)
8 = Maintenance Operation Protocol (MOP)
9 = Serial Line Internet Protocol (SLIP)
10 = XRemote (provides support for X Windows over a serial line)

End of Table

Terminal Server Messages

The following variables pertain to terminal server message parameters:

tsMsgDuration

Sets the length of time (in milliseconds) allocated to reissue a message. The minimum nonzero setting is 10000.0. A setting of 0 will not repeat the message.

Syntax: Integer

Access: Read-write

tsMsgIntervaltim

Sets the interval (in milliseconds) that occurs between reissues of the same message. The minimum (nonzero) setting for this interval is 10000.0 milliseconds. If set to 10000.0, the intervals will become more frequent as the message duration gets close to expiring. For example, 2 hours, 1 hour, 30 minutes, 5 minutes, and 1 minute.

Syntax: Integer

Access: Read-write

tsMsgSend

Determines what action to take after the message has been sent.

Syntax: Integer

Access: Read-write

The possible integer values follow:

1 = nothing
2 = reload
3 = message done
4 = abort

tsMsgText

Sets the text of the message. Up to 256 characters can be included in the message.

Syntax: Display string

Access: Read-write

tsMsgTmpBanner

Determines whether or not to use the message text as a temporary banner.

Syntax: Integer (1 = no, 2 = yes, in addition to the regular banner)

Access: Read-write

tsMsgTtyLine

Selects the TTY line to which you want the message sent. Setting this variable to –1 will send the message to all TTY lines.

Syntax: Integer

Access: Read-write

Transmission Control Protocol (TCP) Group

These variables can be applied to Cisco products running the Transmission Control Protocol (TCP). These variables provide statistics on the number of input and output bytes and packets for TCP connections.

TCP Connection Table

The TCP connection table, *ltcpConnTable*, contains the following five variables: *loctcpConnElapsed*, *loctcpConnInBytes*, *loctcpConnInPkts*, *loctcpConnOutBytes*, and *loctcpConnOutPkts*.

The index to this table includes the local host address and port number and the remote host address and port number for each TCP connection that is active for the device. These values are represented by *tcpConnLocalAddress*, *tcpConnLocalPort*, *tcpConnRemAddress*, and *tcpConRemPort*.

For *n* number of TCP connections, there are *n* rows in the table. The value *n* can change at any time if another TCP connection opens or if an existing TCP connection closes.

In Table 12, TCP A represents the first TCP connection in the table. The TCP A connection shows 100 input bytes, 100 output bytes, 85 input packets, 85 output packets for the connection. The connection has been established for 60 seconds, or 6000 Timeticks.

Table 12 TCP Connection Table

ltcpConnTable	Elapsed	InBytes	InPkts	OutBytes	OutPkts
TCP A	6000	100	85	100	85
TCP B	4500	200	90	130	100
TCP C	9000	300	100	250	95

loctcpConnElapsed

Provides the length of time that the TCP connection has been established.

Syntax: Timeticks

Access: Read-only

loctcpConnInBytes

Provides the number of input bytes for the TCP connection.

Syntax: Integer

Access: Read-only

loctcpConnInPkts

Provides the number of input packets for the TCP connection.

Syntax: Integer

Access: Read-only

loctcpConnOutBytes

Provides the number of output bytes for the TCP connection.

Syntax: Integer

Access: Read-only

loctcpConnOutPkts

Provides the number of output packets for the TCP connection.

Syntax: Integer

Access: Read-only

End of Table

Temporary Variables

This section is equivalent to the experimental space defined by the Structure of Management Information (SMI). It contains variables that are useful to have, but are beyond the ability of Cisco to control and maintain. Support for these variables can change with each Cisco Systems software release.

Note Unlike the *mib921.txt* file, this quick reference guide organizes variable groups and variables within groups alphabetically so you can quickly look up descriptions of MIB variables.

The temporary variables section includes the following groups of variables:

AppleTalk Group

Variables in this group can be used with all Cisco products running the AppleTalk protocol. These variables provide such information as total number of input and output packets, number of packets with errors, and number of packets with Address Resolution Protocol (ARP) requests and replies.

atArpprobe

Indicates the total number of input ARP probe packets.

Syntax: Integer

Access: Read-only

atArpreply

Indicates the total number of AppleTalk ARP reply packets output.

Syntax: Integer

Access: Read-only

atArpreq

Indicates the total number of input AppleTalk ARP request packts.

Syntax: Integer

Access: Read-only

atAtp

Indicates the total number of AppleTalk ATP packets received.

Syntax: Integer

Access: Read-only

atBcastin

Indicates the total number of AppleTalk input broadcast packets.

Syntax: Integer

Access: Read-only

atBcastout

Indicates the total number of AppleTalk output broadcast packets.

Syntax: Integer

Access: Read-only

atChksum

Indicates the total number of AppleTalk input packets with checksum errors.

Syntax: Integer

Access: Read-only

atDdpbad

Indicates the total number of illegal-sized AppleTalk Datagram Delivery Protocol (DDP) packets received.

Syntax: Integer

Access: Read-only

atDdplong

Indicates the total number of long DDP packets received.

Syntax: Integer

Access: Read-only

atDdpshort

Indicates the total number of short DDP packets received.

Syntax: Integer

Access: Read-only

atEcho

Indicates the total number of AppleTalk echo packets received.

Syntax: Integer

Access: Read-only

atEchoill

Indicates the total number of illegal AppleTalk echo packets received.

Syntax: Integer

Access: Read-only

atForward

Indicates the total number of AppleTalk packets forwarded.

Syntax: Integer

Access: Read-only

atHopcnt

Indicates the total number of AppleTalk input packets that have exceeded the maximum hop count.

Syntax: Integer

Access: Read-only

atInmult

Indicates the total number of AppleTalk input packets with multicast addresses.

Syntax: Integer

Access: Read-only

atInput

Indicates the total number of input AppleTalk packets.

Syntax: Integer

Access: Read-only

atLocal

Indicates the total number of AppleTalk input packets for this host.

Syntax: Integer

Access: Read-only

atNbpin

Indicates the total number of AppleTalk Name Binding Protocol (NBP) packets received.

Syntax: Integer

Access: Read-only

atNbpout

Indicates the total number of NBP packets sent.

Syntax: Integer

Access: Read-only

atNoaccess

Indicates the total number of AppleTalk packets dropped due to access control.

Syntax: Integer

Access: Read-only

atNobuffer

Indicates the total number of AppleTalk packets lost due to no memory.

Syntax: Integer

Access: Read-only

atNoencap

Indicates the total number of AppleTalk packets that were dropped because they could not be encapsulated.

Syntax: Integer

Access: Read-only

atNoroute

Indicates the total number of number of AppleTalk packets dropped because the router did not know where to forward them.

Syntax: Integer

Access: Read-only

atNotgate

Indicates the total number of AppleTalk input packets received while AppleTalk routing was not enabled.

Syntax: Integer

Access: Read-only

atOutput

Indicates the total number of AppleTalk output packets.

Syntax: Integer

Access: Read-only

atRtmpin

Indicates the total number of AppleTalk Routing Table Maintenance Protocol (RTMP) packets received.

Syntax: Integer

Access: Read-only

atRtmpout

Indicates the total number of RTMP packets sent.

Syntax: Integer

Access: Read-only

atUnknown

Indicates the total number of unknown AppleTalk input packets.

Syntax: Integer

Access: Read-only

atZipin

Indicates the total number of AppleTalk Zone Information Protocol (ZIP) packets received.

Syntax: Integer

Access: Read-only

atZipout

Indicates the total number of ZIP packets sent.

Syntax: Integer

Access: Read-only

Chassis Group

Variables in this group apply to the Cisco chassis and provide information about the hardware within the chassis such as the system software version of the read-only memory (ROM) and the type of chassis (whether it is Cisco 2000, Cisco 3000, and so on).

The Chassis Card table, *cardTableEntry*, contains information on a per-chassis basis and includes the following variables: *cardDescr*, *cardHwVersion*, *cardIndex*, *cardSerial*, *cardSlotNumber*, *cardSwVersion*, and *cardType*. The index to this table is *cardIndex*. If the device has *n* number of cards, the table will contain *n* number of rows.

chassisId

Provides the unique ID number for the chassis. This number contains the value of the CPU serial number or ID number (if available); otherwise it will be empty. This number also can be set with *snmp-server chassis-id*. An example of a value for a CPU serial number is 00160917.

Syntax: Display string

Access: Read-write

chassisType

Indicates the type of chassis for the product. For example, c4000 indicates a Cisco 4000 router.

Syntax: Integer

Access: Read-only

The following are integer values for this variable:

1 = Unknown

2 = Multibus (for example, CGS or ASM)

3 = AGS+

4 = IGS

5 = Cisco 2000

6 = Cisco 3000

7 = Cisco 4000

8 = Cisco 7000

9 = Communication server 500

chassisVersion

Provides the chassis hardware revision level, or an empty string if the information is unavailable. Examples of the types of chassis versions are D or AO.

Syntax: Display string

Access: Read-only

configRegister

Indicates the value of the configuration register.

Syntax: Integer

Access: Read-only

configRegNext

Indicates the value of the configuration register at next reload.

Syntax: Integer

Access: Read-only

nvRAMSize

Provides the nonvolatile configuration memory in bytes.

Syntax: Integer

Access: Read-only

nvRAMUsed

Provides the number of bytes of nonvolatile configuration memory in use.

Syntax: Integer

Access: Read-only

processorRam

Provides the bytes of RAM available to the CPU of the device.

Syntax: Integer

Access: Read-only

romVersion

Provides the ROM system software version, or an empty string if unavailable. Following is an example of the type of information provided by the *romVersion* variable:

```
System Bootstrap, Version 4.5(3), SOFTWARE [fc1]
Copyright (c) 1986-1992 by cisco Systems
```

Syntax: Display string

Access: Read-only

romSysVersion

Provides the software version of the system software in ROM, or an empty string if the information is unavailable. Following is an example of the type of information provided by the *romSysVersion* variable:

```
GS Software (GS3), Version 9.21(3127) [jdoe 106]
Copyright (c) 1986-1993 by cisco Systems, Inc.
Compiled Thu 08-Apr-93 09:55
```

Syntax: Display string

Access: Read-only Chassis Card table

Chassis Interface Table

The Chassis Interface table, *cardTabl*e, contains the *cardTableEntry* variable. The Cisco Card table, *cardTableEntry*, contains seven entries, or rows. These entries are the following variables: *cardIndex*, *cardType*, *cardDescr*, *cardSerial*, *cardHwVersion*, *cardSwVersion*, and

cardSlotNumber. The index to this table is *cardIndex*. If there are *n* number of cards associated with the device, there will be *n* rows in the table.

For example, in Table 13, there are 4 rows.

Table 13 Card Table

cardType[1]	cardDescr	cardHwVersion	cardSerial
70	MCI interface	1.1	0
70	MCI interface	1.1	0
5	25 MHz 68040		0
24	Environmental Monitor	4	00196849

1. Although the cardTableEntry contains seven entries, this table illustrates the examples for only four entries.

cardDescr

Provides a description of the card used by the router. Examples of the descriptions are *MEC Ethernet* for an MEC board, *25MHz 68040* for the CSC/4, and *CTR Token Ring* for a CTR board.

Syntax: Display string

Access: Read-only

cardHwVersion

Provides the hardware revision level of this card, or an empty string if unavailable.

Syntax: Display string

Access: Read-only

cardIndex

Index into card table (not physical chassis slot number).

Syntax: Integer

Access: Read-only

cardSerial

Provides the serial number of this card, or zero if unavailable.

Syntax: Integer

Access: Read-only

cardSwVersion

Provides the version of the firmware or microcode installed on this card, or an empty string if unavailable. For example, *1.8* indicates MCI microcode 1.8, and *3.0 MADGE 1.01/4.02, TI 000000* indicates CSC-R16M.

Syntax: Display string

Access: Read-only

cardSlotNumber

Provides the chassis slot number. A value of –1 is provided if it is not applicable or cannot be determined.

Syntax: Integer

Access: Read-only

cardType

Syntax: Integer

Access: Read-only

The possible integer values follow:

1 = unknown

2 = csc1

3 = csc2

4 = csc3

5 = csc4

6 = rp

20 = csc-m

21 = csc-mt

22 = csc-mc

23 = csc-mcplus

24 = csc-envm

40 = csc-16

81 = csc-r16

82 = csc-r16m

83 = csc-1r

84 = csc-2r

56 = sci4s

57 = sci2s2t

58 = sci4t

59 = mci1t

60 = mci2t

61 = mci1s

62 = mci1s1t

41 = csc-p

50 = csc-a

51 = csc-e1

52 = csc-e2

53 = csc-y

54 = csc-s

55 = csc-t

80 = csc-r

81 = csc-r16

82 = csc-r16m

83 = csc-1r

84 = csc-2r

56 = sci4s

57 = sci2s2t

58 = sci4t

59 = mci1t

60 = mci2t

61 = mci1s

62 = mci1s1t

63 = mci2s

64 = mci1e

65 = mci1e1t

66 = mci1e2t

67 = mci1e1s

68 = mci1e1s1t

69 = mci1e2s

70 = mci2e

71 = mci2e1t

72 = mci2e2t

73 = mci2e1s

74 = mci2e1s1t

75 = mci2e2s

100 = csc-cctl1

101 = csc-cctl2

110 = csc-mec2

111 = csc-mec4

112 = csc-mec6

113 = csc-fci

114 = csc-fcit

115 = csc-hsci

116 = csc-ctr

150 = sp

151 = eip

152 = fip

153 = hip

154 = sip

155 = trip

156 = fsip

200 = npm-4000-fddi-sas

201 = npm-4000-fddi-das

chassisSlots

Provides the number of slots in this chassis. A value of –1 is provided if the number is not applicable or cannot be determined.

Syntax: Integer

Access: Read-only

DECnet Group

This section describes the Cisco MIB variables pertaining to monitoring and managing a device running the DECnet protocol. These variables gather information, such as hop count, host name, total packets received and sent, and number of packets with header errors.

Note The terms *Level 1* and *Level 2* are used often with these variables. Level 1 routers can communicate with end nodes and with other Level 1 routers in an area. Level 2 routers can communicate with Level 1 routers in the same area and with Level 2 routers in different areas. The term *hellos* is also used. Hosts acknowledge the addresses of other hosts by listening to host hello messages. Hosts learn about nearby routers by listening to router hello messages.

dnBadhello

Provides the total number of received bad hello messages.

Syntax: Integer

Access: Read-only

dnBadlevel1

Provides the total number of bad Level 1 routing packets that have been received.

Syntax: Integer

Access: Read-only

dnBigaddr

Provides the total number of addresses that are too large.

Syntax: Integer

Access: Read-only

dnDatas

Provides the total number of received data packets.

Syntax: Integer

Access: Read-only

dnFormaterr

Provides the total number of DECnet packets received with header errors.

Syntax: Integer

Access: Read-only

dnForward

Provides the total number of DECnet packets that have been forwarded.

Syntax: Integer

Access: Read-only

dnHellos

Provides the total number of hello messages received.

Syntax: Integer

Access: Read-only

dnHellosent

Provides the total number of output hello messages.

Syntax: Integer

Access: Read-only

dnLevel1s

Provides the total number of Level 1 routing packets received.

Syntax: Integer

Access: Read-only

dnLevel1sent

Provides the total number of Level 1 routing packets sent.

Syntax: Integer

Access: Read-only

dnLevel2s

Provides the total number of Level 2 routing packets received.

Syntax: Integer

Access: Read-only

dnLevel2sent

Provides the total number of Level 2 routing packets sent.

Syntax: Integer

Access: Read-only

dnNoaccess

Provides the total number of packets dropped due to access control failure.

Syntax: Integer

Access: Read-only

dnNoencap

Provides the total number of packets that were dropped because they could not be encapsulated.

Syntax: Integer

Access: Read-only

dnNomemory

Provides the total number of transactions denied due to lack of memory.

Syntax: Integer

Access: Read-only

dnNoroute

Provides the total number of packets that were dropped because the router did not know where to forward them.

Syntax: Integer

Access: Read-only

dnNotgateway

Provides the total number of packets that were received while not routing DECnet.

Syntax: Integer

Access: Read-only

dnNotimp

Provides the total number of unknown control packets received.

Syntax: Integer

Access: Read-only

dnNotlong

Provides the total number of received packets not in the long DECnet format. This number should always be zero.

Syntax: Integer

Access: Read-only

dnNovector

Provides the total number of missing routing vectors. Occurs when a packet is received for which there is no entry in the routing table.

Syntax: Integer

Access: Read-only

dnOtherhello

Provides the total number of hello messages received from another area by a Level 1 router.

Syntax: Integer

Access: Read-only

dnOtherlevel1

Provides the total number of Level 1 routing packets received from another area.

Syntax: Integer

Access: Read-only

dnOtherlevel2

Provides the total number of Level 2 routing packets that have been received from another area.

Syntax: Integer

Access: Read-only

dnReceived

Provides the number of total DECnet packets received.

Syntax: Integer

Access: Read-only

dnToomanyhops

Provides the total number of packets received that exceeded the maximum hop count set for this device and have been discarded.

Syntax: Integer

Access: Read-only

DECnet Area Routing Table

The DECnet area routing table, *dnAreaTable*, includes the following seven variables: *dnAAge, dnACost, dnAHop, dnAIfIndex, dnANextHop, dnAPrio,* and *dnArea.* The index for this table is the DECnet area, or *dnArea.* If there are *n* number of areas for the device, there will be *n* rows in the table.

For example, in Table 14, the DECnet area is 44, the cost is 3, and the maximum number of hops allowed is 2. The interface used to get to area 44 is number 1; the address for the next hop is 46.5; the routing table was updated 30 seconds ago; and the next hop area is prioritized as 1.

Table 14 DECnet Area Routing

dn Area	dnACost	dnAHop	dnA IfIndex	dnA Next Hop	dnAAge	dnA Prio
44	3	2	1	46.5	30	1
24	60	4	2	24.7	12	2
6	17	2	3	6.4	60	3

dnAAge

Provides the age (in seconds) of an area route. When a route is used or has been verified as functional, its age is reset to 0. If a route is not used, its age will gradually grow. Eventually, routes with large ages are cleared out.

Syntax: Integer

Access: Read-only

dnACost

Provides the cost of the router area. The cost value can be an integer from 1 through 63. The cost signifies routing preference. The lower the cost, the better the path.

Syntax: Integer

Access: Read-only

dnAHop

Provides the maximum number of hops for a route to a distant area that the router will accept.

Syntax: Integer

Access: Read-only

dnAIfIndex

Provides the instance ID of the interface providing the next hop address to the area. A zero denotes self. The DECnet table is indexed by *dnArea*. For example, dnAIfIndex.5 is the *ifIndex* for the next hop to DECnet area 5; dnAifIndex 7 is the *ifIndex* for the next hop to DECnet area 7; and so on.

If *dnAIfIndex.5* is set to the value of 4, to get to the next hop for DECnet area 5, the router sends the packet via the interface that has an *ifIndex* of 4.

Syntax: Integer

Access: Read-only

dnANextHop

Provides the DECnet address for the next hop.

Syntax: Octet string

Access: Read-only

dnAPrio

Provides the priority of the next hop router for an area route.

Syntax: Integer

Access: Read-only

dnArea

Indicates the DECnet area for the device.

Syntax: Integer

Access: Read-only

End of Table

DECnet Host Table

The DECnet host table, *dnHostTable*, contains the following seven variables: *dnHAge*, *dnHCost*, *dnHHop*, *dnHIfIndex*, *dnHNextHop*, *dnHost*, and *dnHPrio*.

In Table 15, the first DECnet host address in the table is 44.5. Its cost is 3; the number of hops to the host is 4; and the interface number 1 provides the next hop to address 55.6. The route was updated 30 seconds ago, and the priority for the next hop is set to 4.

Table 15 DECnet Host

dnHost	dnH Cost	dnH Hop	dnHIfIndex	dnH Next Hop	dnH Age	dnH Prio
44.5	3	4	1	55.6	30	4
54.6	1	3	2	33.2	20	3
23.2	2	1	3	25.1	60	2

dnHAge

Provides the age (in seconds) of the route to the host.When a route is used or has been verified as functional, its age is reset to 0. If a route is not used, the age of the route will gradually grow. Eventually, routes with large ages are cleared out.

Syntax: Integer

Access: Read-only

dnHCost

Provides the cost of the path to this device.

Syntax: Integer

Access: Read-only

dnHHop

Provides the number of hops to this device.

Syntax: Integer

Access: Read-only

dnHIfIndex

Provides the index of the interface to the next hop address to the node. 0 denotes self.

Syntax: Integer

Access: Read-only

dnHost

Provides the DECnet node address.

Syntax: Integer

Access:

dnHNextHop

Provides the DECnet address of the next hop destination.

Syntax: Octet string

Access: Read-only

dnHPrio

Provides the priority of the next hop router for the node.

Syntax: Integer

Access: Read-only

End of Table

DECnet Interface Table

The DECnet interface table, *dnIfTabl*e, contains the *dnIfCost* variable. The index to this table is *ifIndex*, or the interface number. If there are *n* number of interfaces associated with the device, there will be *n* rows in the table.

For example, in Table 16, interface 1 has a cost of 20; interface 2 has a cost of 31; and so on.

Table 16 DECnet Interface

Interface Number	dnIfCost
1	20
2	31
3	12

dnIfCost

Indicates the cost of this interface.

Syntax: Integer

Access: Read-only

End of Table

Novell Group

The variables in this group can be used with all Cisco products running the Novell protocol. These variables provide such information as total number of input and output packets, number of packets with errors, and number of packets with service access point (SAP) requests and replies.

novellBcastin

Indicates the total number of Novell input broadcast packets.

Syntax: Integer

Access: Read-only

novellBcastout

Indicates the total number of Novell output broadcast packets.

Syntax: Integer

Access: Read-only

novellChksum

Indicates the total number of Novell input packets with checksum errors.

Syntax: Integer

Access: Read-only

novellFormerr

Indicates the total number of Novell input packets with header errors.

Syntax: Integer

Access: Read-only

novellForward

Indicates the total number of Novell packets forwarded.

Syntax: Integer

Access: Read-only

novellHopcnt

Indicates the total number of Novell input packets that have exceeded the maximum hop count.

Syntax: Integer

Access: Read-only

novellInmult

Indicates the total number of Novell input multicast packets.

Syntax: Integer

Access: Read-only

novellInput

Indicates the total number of Novell input packets.

Syntax: Integer

Access: Read-only

novellLocal

Indicates the total number of Novell input packets for this host.

Syntax: Integer

Access: Read-only

novellNoencap

Indicates the total number of Novell packets dropped due to output encapsulation failure.

Syntax: Integer

Access: Read-only

novellNoroute

Indicates the total number of Novell packets dropped because the router did not know where to forward them.

Syntax: Integer

Access: Read-only

novellOutput

Indicates the total number of Novell output packets.

Syntax: Integer

Access: Read-only

novellSapout

Indicates the total number of Novell service access point (SAP) request packets sent.

Syntax: Integer

Access: Read-only

novellSapreply

Indicates the total number of Novell SAP reply packets sent.

Syntax: Integer

Access: Read-only

novellSapreqin

Indicates the total number of Novell SAP request packets received.

Syntax: Integer

Access: Read-only

novellSapresin

Indicates the total number of Novell SAP response packets received.

Syntax: Integer

Access: Read-only

novellUnknown

Indicates the total number of unknown Novell input packets.

Syntax: Integer

Access: Read-only

Virtual Network System (VINES) Group

The variables in this group can be used with all Cisco products running the Banyan Virtual Network System (VINES) protocol. This protocol is derived from the Xerox Network Systems (XNS) protocol. These variables provide information such as total number of input and output packets, number of packets with errors, and number of packets with Internet Control Protocol (ICP) requests and replies.

vinesBcastfwd

Indicates the total number of VINES broadcast packets forwarded.

Syntax: Integer

Access: Read-only

vinesBcastin

Indicates the total number of VINES input broadcast packets.

Syntax: Integer

Access: Read-only

vinesBcastout

Indicates the total number of VINES output broadcast packets.

Syntax: Integer

Access: Read-only

vinesCksumerr

Indicates the total number of VINES input packets with checksum errors.

Syntax: Integer

Access: Read-only

vinesClient

Indicates the next VINES sub-network number that this router will assign to a client.

Syntax: Integer

Access: Read-only

vinesEchoIn

Indicates the total number of VINES echo packets received.

Syntax: Integer

Access: Read-only

vinesEchoOut

Indicates the total number of VINES echo packets generated.

Syntax: Integer

Access: Read-only

vinesEncapsfailed

Indicates the total number of VINES packets dropped because they could not be encapsulated.

Syntax: Integer

Access: Read-only

vinesFormaterror

Indicates the total number of VINES input packets with header errors.

Syntax: Integer

Access: Read-only

vinesForwarded

Indicates the total number of VINES packets forwarded.

Syntax: Integer

Access: Read-only

vinesHopcount

Indicates the total number of VINES input packets that have exceeded the maximum hop count.

Syntax: Integer

Access: Read-only

vinesIcpIn

Indicates the total number of VINES Internet Control Protocol (ICP) packets received.

Syntax: Integer

Access: Read-only

vinesIcpOut

Indicates the total number of VINES ICP packets generated.

Syntax: Integer

Access: Read-only

vinesInput

Indicates the total number of VINES input packets.

Syntax: Integer

Access: Read-only

vinesLocaldest

Indicates the total number of VINES input packets for this host.

Syntax: Integer

Access: Read-only

vinesMacEchoIn

Indicates the total number of VINES MAC level echo packets received.

Syntax: Integer

Access: Read-only

vinesMacEchoOut

Indicates the total number of VINES Media Access Control (MAC) level echo packets generated.

Syntax: Integer

Access: Read-only

vinesMetricOut

Indicates the total number of VINES ICP metric notification packets generated.

Syntax: Integer

Access: Read-only

VinesNet

Indicates the VINES network number of this router.

Syntax: Integer

Access: Read-only

vinesNocharges

Indicates the total number of VINES broadcast packets not forwarded to all interfaces because the *no charges only* bit on the modem was set to *on*.

Syntax: Integer

Access: Read-only

vinesNoroute

Indicates the total number of VINES packets dropped because the router did not know where to forward them.

Syntax: Integer

Access: Read-only

vinesNotgt4800

Indicates the total number of VINES broadcast packets not forwarded to all interfaces because the *over 4800 bps* bit on the modem was set to *on*.

Syntax: Integer

Access: Read-only

vinesNotlan

Indicates the total number of VINES broadcast packets not forwarded to all interfaces because the *lan only* bit on the modem was set to *on*.

Syntax: Integer

Access: Read-only

vinesOutput

Indicates the total number of VINES output packets.

Syntax: Integer

Access: Read-only

vinesProxy

Indicates the total number of VINES packets that were sent to an actual Banyan server as a proxy for a client.

Syntax: Counter

Access: Read-only

vinesProxyReply

Indicates the total number of received VINES packets that were responses to proxy packets sent by the router.

Syntax: Counter

Access: Read-only

vinesSubnet

Syntax: Integer

Access: Read-only

vinesUnknown

Indicates the total number of unknown VINES input packets.

Syntax: Integer

Access: Read-only

Banyan VINES Interface Table

The Banyan VINES Interface table, *vinesIfTableEntry*, contains all the variables described in the Banyan VINES group. The index to the table is *ifIndex*. *ifIndex* indicates the number of the interface. If the device has *n* number of interfaces, the VINES Interface table will contain *n* rows.

In the Interface table shown in Table 17, the first column indicates the number of interfaces on the devices. Each of the variables in the VINES Interface table occupies one column; for example, *vinesIfMetric* is shown in a column, followed by *vinesIfEnctype* in the next column, and so on.

Table 17 Banyan VINES Interface Table

Interface Number	vinesIfMetric[1]	vinesIfEnctype	and so on
1	3	1	
2	5	3	

1. Although the MIB variables are presented alphabetically in this guide, the MIB variables in the VINES Interface table, *vinesIfTableEntry*, are not presented alphabetically in the *mib921.txt* file. Therefore, this table does not present these MIB variables alphabetically.

vinesIfAccesslist

Provides the outgoing access list number for the VINES protocol.

Syntax: Integer

Access: Read-only

vinesifArpEnabled

Indicates whether the VINES protocol ARP replies are enabled.

Syntax: Integer (0 = not enabled, 1 = enabled)

Access: Read-only

vinesIfEnctype

Indicates the type of data link encapsulation that will be used by broadcasts sent to the router.

Syntax: Integer (1 = ARPA, 3 = SNAP, 5 = HDLC, 12 = X.25, 13 = X.25, 25 = VINES TR, 27 = Frame Relay, 28 = SMDS)

Access: Read-only

vinesIfFastokay

Indicates whether fast switching is supported for the VINES protocol.

Syntax: Integer (0 = fast switching not requested or not supported, 1 = fast switching requested and supported)

Access: Read-only

vinesIfLineup

Indicates whether the VINES protocol line is up or down.

Syntax: Integer (0 = down, 1 = up)

Access: Read-only

vinesIfMetric

Provides the metric value for the VINES protocol. Banyan servers use delay metrics to compute timeouts when communicating with other hosts. The metric value is either manually assigned to the interface by using the **vines metric** command or is automatically assigned by the system.

Syntax: Integer

Access: Read-only

vinesIfPropagate

Indicates whether the VINES protocol “propagate” is enabled.

Syntax: Integer (1 = enabled, 0 = not enabled)

Access: Read-only

vinesIfRedirectInterval

Provides the redirect interval for the VINES protocol.

Syntax: Integer

Access: Read-only

vinesIfRouteCache

Indicates whether fast switching is supported for the VINES protocol.

Syntax: Integer

Access: Read-only

vinesIfRxArp0

Provides the number of input ARP query request messages for the VINES protocol. The four types of ARP messages include the following and apply to vinesIfRxArp0-vinesIfRxArp3:

- Service request (type 0)—Query to find servers
- Service response (type 1)—Server indicating its presence
- Assignment request (type 2)—Client asking to be assigned a VINES IP address
- Assignment response (type 3)—Server assigning a VINES IP address to a client

Syntax: Counter

Access: Read-only

vinesIfRxArp1

Provides the number of input ARP query response messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxArp2

Provides the number of input ARP assignment request messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxArp3

Provides the number of input ARP assignment response messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxArpIllegal

Provides the number of input illegal ARP messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxBcastDuplicate

Provides the input duplicate broadcast count for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxBcastForwarded

Provides the VINES protocol number of input packets forwarded to another interface.

Syntax: Counter

Access: Read-only

vinesIfRxBcastHelpered

Provides the VINES protocol number of input packets helpered to another server. Helpered packets are broadcasts received from a serverless network that should be thrown away according to the fields in the VINES IP header. Instead of being thrown away, they are resent on another interface, so they will be received by a VINES server.

Syntax: Counter

Access: Read-only

vinesIfRxBcastin

Provides the input broadcast count for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxChecksumError

Provides the number of input packets with checksum errors for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxEcho

Provides the number of input IPC echo messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxFormatError

Provides the number of input packets with format errors for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxForwarded

Provides the VINES protocol number of input packets forwarded to another interface.

Syntax: Counter

Access: Read-only

vinesIfRxIcpError

Provides the number of input interprocess communications (ICP) error messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxIcpIllegal

Provides the number of input illegal ICP messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxIcpMetric

Provides the number of input ICP metric messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxIpc

Provides the number of input IPC messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxLocalDest

Provides the VINES protocol number of input packets destined for this router.

Syntax: Counter

Access: Read-only

vinesIfRxMacEcho

Provides the number of input MAC layer echo frames for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxNoRoute

Provides the VINES protocol number of input packets that were dropped because there was no route to the destination.

Syntax: Counter

Access: Read-only

vinesIfRxNotEnabled

Provides the VINES protocol number of input packets that were discarded because the interface was not configured.

Syntax: Counter

Access: Read-only

vinesIfRxProxyReply

Provides the VINES protocol number of responses to proxy packets.

Syntax: Counter

Access: Read-only

vinesIfRxRtp0

Provides the number of illegal input Routing Table Protocol (RTP) type 0 messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxRtp1

Provides the number of input RTP type 1 (request for information) messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxRtp2

Provides the number of illegal input RTP type 2 messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxRtp3

Provides the number of illegal input RTP type 3 messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxRtp4

Provides the number of input RTP update messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxRtp5

Provides the number of input RTP response messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxRtp6

Provides the number of input RTP redirect messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxRtpIllegal

Provides the number of all other illegal input RTP messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxlpUnknownCnt

Provides the number of input messages from unknown VINES protocols.

Syntax: Counter

Access: Read-only

vinesIfRxlpcUnknownCnt

Provides the number of input messages from unknown VINES IPC ports.

vinesIfRxSpp

Provides the number of input SPP messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxUnknown

Provides the number of input packets of unknown VINES protocols for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfRxZeroHopCount

Provides VINES protocol number of input packets dropped due to a zero hop count.

Syntax: Counter

Access: Read-only

vinesIfServerless

Indicates whether the VINES protocol serverless support is enabled.

Syntax: Counter (1 = enabled, 0 = not enabled)

Access: Read-only

vinesIfServerlessBcast

Indicates whether VINES protocol serverless broadcasting support is enabled.

Syntax: Counter (0 = not enabled, 1 = enabled)

Access: Read-only

vinesIfSplitDisabled

Indicates whether the VINES protocol split horizon is enabled.

Syntax: Counter (0 = enabled, 1 = disabled)

Access: Read-only

vinesIfTxArp0

Provides the number of output ARP query request messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxArp1

Provides the number of output ARP query response messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxArp2

Provides the number of output ARP assignment request messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxArp3

Provides the number of input ARP assignment response messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxBcastForwarded

Provides the VINES protocol output broadcast forwarded from another interface.

Syntax: Integer

Access: Read-only

vinesIfTxBcastHelpered

Provides the VINES protocol output broadcast helpered to a Banyan server.

Syntax: Integer

Access: Read-only

vinesIfTxEcho

Provides the number of output IPC echo messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxFailedAccess

Provides the output access list failures for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxFailedDown

Provides the output interface down count for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxFailedEncaps

Provides VINES protocol output encapsulation.

Syntax: Counter

Access: Read-only

vinesIfTxForwarded

Provides the number of forwarded packets for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxIcpError

Provides the number of output IPC error messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxIcpMetric

Provides the number of output IPC metric messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxIpc

Provides the number of output ICP messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxMacEcho

Provides the number of output IPC MAC-layer echo frames for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxBcast

Provides broadcast packets that were generated by the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxNotBcastNotgt4800

Provides the VINES protocol output broadcast not sent due to high-speed class. This occurs if a received packet is marked to be sent only on network interfaces with a speed of 4800 bps or greater. The counter is incremented on interfaces with a speed of less than 4800 whenever this type of packet is received.

Syntax: Counter

Access: Read-only

vinesIfTxNotBcastNotlan

Provides the VINES protocol output broadcast not sent due to *Lan Only* class.

Syntax: Counter

Access: Read-only

vinesIfTxNotBcastPpcharge

Provides VINES protocol output broadcast not sent due to *No Charges* class.

Syntax: Counter

Access: Read-only

vinesIfTxNotBcastToSource

Provides the VINES protocol output broadcast packets that were not sent due to the interface leading back to the source.

Syntax: Counter

Access: Read-only

vinesIfTxProxy

Provides the number of proxy packets sent by the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxRtp0

Provides the number of illegal output RTP type 0 messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxRtp1

Provides the number of output RTP type 1 (request messages) for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxRtp2

Provides the number of illegal output RTP type 2 messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxRtp3

Provides the number of illegal output RTP type 3 messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxRtp4

Provides the number of output RTP type 4 (update messages) for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxRtp5

Provides the number of output RTP type 5 (response messages) for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxRtp6

Provides the number of output RTP type 6 (redirect messages) for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesIfTxSpp

Provides the number of output SPP messages for the VINES protocol.

Syntax: Counter

Access: Read-only

vinesifTxUnicast

Provides the unicast packets that were generated for the VINES protocol.

Syntax: Counter

Access: Read-only

Xerox Network Systems (XNS) Group

This group is present in all router-based products running the Xerox Network Systems (XNS) protocol.These variables provide such information as the number of packets forwarded, total number of input packets, and total number of packets transmitted with errors.

xnsBcastin

Indicates the total number of XNS input broadcast packets.

Syntax: Integer

Access: Read-only

xnsBcastout

Indicates the total number of XNS output broadcast packets.

Syntax: Integer

Access: Read-only

xnsChksum

Indicates the total number of XNS input packets with checksum errors.

Syntax: Integer

Syntax: Read-only

xnsEchorepin

Indicates the total number of XNS echo reply packets received.

Syntax: Integer

Access: Read-only

xnsEchorepout

Indicates the total number of XNS echo reply packets sent.

Syntax: Integer

Access: Read-only

xnsEchoreqin

Indicates the total number of XNS echo request packets were received.

Syntax: Integer

Access: Read-only

xnsEchoreqout

Indicates the total number of XNS echo request packets sent.

Syntax: Integer

Access: Read-only

xnsErrin

Indicates the total number of XNS error input packets.

Syntax: Integer

Access: Read-only

xnsErrout

Indicates the total number of XNS error output packets.

Syntax: Integer

Access: Read-only

xnsForward

Indicates the total number of XNS packets forwarded.

Syntax: Integer

Access: Read-only

xnsFormerr

Indicates the total number of XNS input packets with header errors.

Syntax: Integer

Access: Read-only

xnsFwdbrd

Indicates the total number of XNS broadcast packets forwarded.

Syntax: Integer

Access: Read-only

xnsHopcnt

Indicates the total number of XNS input packets that exceeded the maximum hop count.

Syntax: Integer

Access: Read-only

xnsInmult

Indicates the total number of XNS input packets received with multicast addresses.

Syntax: Integer

Access: Read-only

xnsInput

Indicates the total number of input XNS packets.

Syntax: Integer

Access: Read-only

xnsLocal

Indicates the total number of XNS input packets for this host.

Syntax: Integer

Access: Read-only

xnsNoencap

Provides the total number of XNS packets dropped because they could not be encapsulated.

Syntax: Integer

Access: Read-only

xnsNoroute

Indicates the total number of XNS packets that were discarded because the router did not know where to forward them.

Syntax: Integer

Access: Read-only

xnsNotgate

Indicates the total number of XNS input packets received while not XNS routing was not enabled.

Syntax: Integer

Access: Read-only

xnsOutput

Indicates the total number of XNS output packets.

Syntax: Integer

Access: Read-only

xnsUnknown

Indicates the total number of unknown XNS input packets.

Syntax: Integer

Access: Read-only

Public SNMP Traps Supported by Cisco

You set up SNMP traps on specific devices to obtain useful information such as the change in a device configuration or the absence of proper user authentication with a request. When the SNMP agent on the device detects a change, it immediately sends an SNMP trap to the NMS system.

Cisco products, including the routers, communication servers, and protocol translators, support the SNMP traps specified in RFC 1213, *Management Information Base for Network Management of TCP/IP-based Internets: MIB-II*. The *warmStart* trap in MIB II is not supported by Cisco.

Following are the standard SNMP traps supported by Cisco.

authenticationFailure

This trap is sent to the NMS system if the SNMP agent detects that proper user authentication was not provided with a request. User authentication enhances the security of the devices by ensuring that only privileged users with passwords are allowed to access the system.

coldStart

The SNMP agent sends a *coldStart* trap when its device has reinitialized itself and its configuration or protocol implementation has been changed.

egpNeighborloss

An *egpNeighborLoss* trap indicates that an (Exterior Gateway Protocol) EGP neighbor is down. Neighboring routers are two routers that have interfaces to a common network and exchange routing information. An exterior router uses EGP to advertise its knowledge of routes to networks within its autonomous system. It sends these advertisements to the core

routers, which then readvertise their collected routing information to the exterior router. A neighbor or peer router is any router with which the router communicates using EGP.

linkDown

A *linkDown* trap is sent by the SNMP agent to the NMS system if a link in a configuration of a device has failed. For example, the link could be a serial line connecting two devices or an Ethernet link between two networks.

linkUp

A *linkUp* trap indicates the recognition of an SNMP agent that a link in a configuration of a device has become active.

SNMP Traps Defined by Cisco

Following are the Cisco private SNMP traps that are implemented in Cisco products including the router, communications server, and protocol translator.

reload

This trap is sent after a reload command is issued.

tcpConnectionClose

The *tty* trap indicates that a TCP connection, which existed previously for a tty session, has been terminated.

Variables Supported in RFC 1285

The following variables in RFC 1285 are supported in Software Release 9.0 and later:

snmpFddiSMTNumber

snmpFddiSMTIndex

snmpFddiSNMTStationId

snmpFddiSMTOpVersionId

snmpFddiSMTHiVersionId

snmpFddiSMTLoVersionId

snmpFddiSMTCFState

snmpFddiMACNumber

snmpFddiMACNumber

snmpFddiMACSMTIndex

snmpFddiMACIndex

snmpFddiMACTReq

snmpFddiMACTNegj

snmpFddiMACTMax

snmpFddiMACTvxValue

snmpFddiMACMin

snmpFddiMACFrameCts

snmpFddiMACErrorCts

snmpFddiMACLostCts

snmpFddiMACChipSet

Support of MIBs by Cisco in Software Releases

This section lists the Cisco private MIB variables that have been introduced after Software Release 8.0.

Software Release 8.2

The following list describes the MIB variables introduced with Software Release 8.2:

writeMem

writeNet

busyPer

avgBusy1

avgBusy5

idleCount

idleWired

locIfCarTrans

locIfReliab

locIfDelay

locIfLoad

locIfCollisions

tsLineNoise

dnAreaTable

dnACost

dnAHop

dnAifIndex

dnANextHop

dnAAge

dnAPrio

vinesInput

vinesOutput

vinesLocaldest

vinesForwarded

vinesBcastin

vinesBcastout

vinesBcastfwd

vinesNotlan

vinesNotgt4800

vinesNocharges

vinesFormaterror

vinesCksumerr

vinesHopcout

vinesNoroute

vinesEncapsfailed

vinesUnkown

vinesIcpIn

vinesIcpOut

vinesMetricOut

vinesMacEchoIn

vinesMacEchoOut

vinesEchoIn

vinesEchoOut

Software Release 8.3

The following list describes the MIB variables introduced with Software Release 8.3:

bufferHgsize

bufferHgTotal

bufferHgFree

bufferHgMax

bufferHgHit

bufferHgMiss

bufferHgTrim

bufferHgCreate

locIfInputQueueDrops

locIfOutputQueueDrops

ipNoaccess

actCheckPoint

tsMsgTtyLine

tsMsgIntervaltim

tsMsgDuration

tsMsgTest

tsMsgTmpBanner

tsMsgSend

dnIfTable

dnIfCost

Software Release 9.0

The following list provides the MIB variables introduced with Software Release 9.0:

netConfigProto

hostConfigProto

sysConfigAddr

sysConfigName

sysConfigProto

sysClearARP

sysClearInt

envPresent

envTestPt1Descr

envTestPt1Measure

envTestPt2Descr

envTestPt2Measure

envTestPt3Descr

envTestPt3Measure

envTestPt4Descr

envTestPt4Measure

envTestPt5Descr

envTestPt5Measure

envTestPt6Descr

envTestPt6Measure

locIfDescr

locIfPakmon

Software Release 9.1

The following list provides the MIB variables introduced with Software Release 9.1:

envTestPt4MarginPercent

envTestPt5MarginPercent

envTestPt6MarginPercent

envTestPt1last

envTestPt2last

envTestPt3last

envTestPt4last

envTestPt5last

envTestPt6last

envTestPt1MarginVal

envTestPt2MarginVal

envTestPt3MarginVal

envTestPt4MarginVal

envTestPt5MarginVal

envTestPt6MarginVal

envTestPt1warn

envTestPt2warn

envTestPt3warn

envTestPt4warn

envTestPt5warn

envTestPt6warn

envFirmVersion

envTechnicianID

envType

envBurnDate

envSerialNumber

locIfSlowInPkts

locIfSlowOutPkts

locIfSlowInOctets

locIfSlowOutOctets

locIfFastInPkts

locIfFastOutPkts

locIfFastInOctets

locIfFastOutOctets

locIfotherInPkts

locIfotherOutPkts

locIfotherInOctets

locIfotherOutOctets

locIfipInPkts

locIfipOutPkts

locIfipInOctets

locIfipOutOctets

locIfdecnetInPkts

locIfdecnetOutPkts

locIfdecnetInOctets

locIfdecnetOutOctets

locIfxnsInPkts

locIfxnsOutPkts

locIfxnsInOctets

locIfxnsOutOctets

locIfclnsInPkts

locIfclnsOutPkts

locIfclnsInOctets

locIfclnsOutOctets

locIfappletalkInPkts

locIfappletalkOutPkts

locIfappletalkInOctets

locIfappletalkOutOctets

locIfnovellInPkts

locIfnovellOutPkts

locIfnovellInOctets

locIfnovellOutOctets

locIfapolloInPkts

locIfapolloOutPkts

locIfapolloInOctets

locIfapolloOutOctets

locIfvinesInPkts

locIfvinesOutPkts

locIfvinesInOctets

locIfvinesOutOctets

locIfbridgedInPkts

locIfbridgedOutPkts

locIfbridgedInOctets

locIfbridgedOutOctets

locIfsrbInPkts

locIfsrbOutPkts

locIfsrbInOctets

locIfsrbOutOctets

locIfchaosInPkts

locIfchaosOutPkts

locIfchaosInOctets

locIfchaosOutOctets

locIfpupInPkts

locIfpupOutPkts

locIfpupInOctets

locIfpupOutOctets

locIfmopInPkts

locIfmopOutPkts

locIfmopInOctets

locIfmopOutOctets

locIflanmanInPkts

locIflanmanOutPkts

locIflanmanInOctets

locIflanmanOutOctets

locIfstunInPkts

locIfstunOutPkts

locIfstunInOctets

locIfstunOutOctets

locIfspanInPkts

locIfspanOutPkts

locIfspanInOctets

locIfspanOutOctets

locIfarpInPkts

locIfarpOutPkts

locIfarpInOctets

locIfarpOutOctets

locIfprobeInPkts

locIfprobeOutPkts

locIfprobeInOctets

locIfprobeOutOctets

flashSize

flashFree

flashcontoller

flashcard

flashVPP

flashErase

flashEraseTime

flashEraseStatus

flashToNet

flashToNetTime

flashToNetStatus

netToFlash

netToFlashTime

netToFlashStatus

flashStatus

flashEntries

flashDirName

flashDirSize

flashDirStatus

Software Release 9.21

The following list provides the MIB variables introduced with software release 9.21:

locIfDribbleInputs

vinesProxy

vinesProxyReply

vinesNet

vinesSubNet

vinesClient

vinesIfMetric

vinesIfMetric

vinesIfEnctype

vinesIfAccesslist

vinesIfPropagate

vinesIfArpEnabled

vinesIfServerless

vinesIfServerlessBcast

vinesIfRedirectInterval

vinesIfSplitDisabled

vinesIfLineup

vinesIfFastokay

vinesIfRouteCache

vinesIfIn

vinesIfOut

vinesIfInBytes

vinesIfOutBytes

vinesIfRxNotEnabled

vinesIfRxFormatError

vinesIfRxLocalDest

vinesIfRxBcastin

vinesIfRxForwarded

vinesIfRxNoRoute

vinesIfRxZeroHopCount

vinesIfRxChecksumError

vinesIfRxArp0

vinesIfRxArp1

vinesIfRxArp2

vinesIfRxArp3

vinesIfRxArpIllegal

vinesIfRxIcpError

vinesIfRxIcpMetric

vinesIfRxIcpIllegal

vinesIfRxIpc

vinesIfRxRtp0

vinesIfRxRtp1

vinesIfRxRtp2

vinesIfRxRtp3

vinesIfRxRtp4

vinesIfRxRtp5

vinesIfRxRtp6

vinesIfRxRtpIllegal

vinesIfRxSpp

vinesIfRxUnknown

vinesIfRxBcastHelpered

vinesIfRxBcastForwarded

vinesIfRxBcastDuplicate

vinesIfRxEcho

vinesIfRxMacEcho

vinesIfRxProxyReply

vinesIfTxUnicast

vinesIfTxBcast

vinesIfTxForwarded

vinesIfTxFailedEncaps

vinesIfTxFailedAccess

vinesIfTxFailedDown

vinesIfTxNotBcastToSource

vinesIfTxNotBcastNotlan

vinesIfTxNotBcastNotgt4800

vinesIfTxNotBcastPpcharge

vinesIfTxBcastForwarded

vinesIfTxBcastHelpered

vinesIfTxArp0

vinesIfTxArp1

vinesIfTxArp2

vinesIfTxArp3

vinesIfTxIcpError

vinesIfTxIcpMetric

vinesIfTxIpc

vinesIfTxRtp0

vinesIfTxRtp1

vinesIfTxRtp2

vinesIfTxRtp3

vinesIfTxRtp4

vinesIfTxRtp5

vinesIfTxRtp6

vinesIfTxSpp

vinesIfTxEcho

vinesIfTxMacEcho

vinesIfTxProxy

chassisType

chassisVersion

chassisId

romVersion

romSysVersion

processorRam

nvRAMSize

nvRAMUsed

configRegister

configRegNext

cardTable

cardTableEntry

cardIndex

cardType

cardDescr

cardSerial

cardHwVersion

cardSwVersion

cardSlotNumber

chassisSlots