NAME

snmp.conf - configuration files for the Net-SNMP applications

DESCRIPTION

Applications built using the Net-SNMP libraries typically use one or more configuration files to control various aspects of their operation. These files (**snmp.conf** and **snmp.local.conf**) can be located in one of several locations, as described in the *snmp_config(5)* manual page.

In particular, `/etc/snmp/snmp.conf` is a common file, containing the settings shared by all users of the system. `~/.snmp/snmp.conf` is a personal file, with the settings specific to a particular user.

HOST-SPECIFIC FILES

Host-specific files may also be loaded and will be searched for if a transport name is specified that matches a *PATH/hosts/HOST.conf* file. For example, if you wanted a particular host to use SNMPv2c by default you could create a ~/.snmp/hosts/NAME.conf file and in it put:

```
defVersion 2c
```

Any connections set to connect to the hostname *NAME* will use SNMPv2c. Also see the *transport* token below for additional host-specific examples.

Host-specific configuration files are loaded at the time the connection is opened. Thus they're generally loaded after all other configuration files and can be used to override settings from the generic files.

To avoid loading any host-specific config files set "dontLoadHostConfig true" in your snmp.conf file.

COMMAND-LINE OPTIONS

All of the tokens described in this file can be used on the command line of Net-SNMP applications as well by prefixing them with "−−". EG, specifying *−−dontLoadHostConfig=true* on the command line will turn of loading of the host specific configuration files.

IMPORTANT NOTE

Several of these directives may contain sensitive information (such as pass phrases). Configuration files that include such settings should only be readable by the user concerned.

As well as application-specific configuration tokens, there are several directives that relate to standard library behaviour, relevant to most Net-SNMP applications. Many of these correspond to standard command-line options, which are described in the *snmpcmd(1)* manual page.

These directives can be divided into several distinct groups.

CLIENT BEHAVIOUR

defDomain application domain

The transport domain that should be used for a certain application type unless something else is specified.

defTarget application domain target

The target that should be used for connections to a certain application if the connection should be in a specific domain.

defaultPort PORT

defines the default UDP port that client SNMP applications will attempt to connect to. This can be overridden by explicitly including a port number in the *AGENT* specification. See the *snmpcmd(1)* manual page for more details.

If not specified, the default value for this token is 161.

transport HOSTSPECIFIER

This special token should go into a hostname-specific configuration file in a *hosts* sub-directory. For example if the file *hosts/foo.conf* exists in the search path it will be loaded if a transport name of *foo* was used. Within the foo.conf file you may put both general snmp.conf settings as well as a special *transport* string to specify the destination to connect to. For example, putting:

```
transport tcp:foo.example.com:9876
```

in the *hosts/foo.conf* file will make applications referencing the *foo* hostname (e.g. *snmpget*) to actually connect via TCP to *foo.exmaple.com* on port 9876.

defVersion (1|2c|3)

defines the default version of SNMP to use. This can be overridden using the **−v** option.

defCommunity STRING

defines the default community to use for SNMPv1 and SNMPv2c requests. This can be overridden using the **−c** option.

alias NAME DEFINITION

Creates an aliased tied to NAME for a given transport definition. The alias can the be referred to using an alias: prefix. Eg, a line of "alias here udp:127.0.0.1:6161" would allow you to use a destination host of "alias:here" instead of "udp:127.0.0.1:6161". This becomes more useful with complex transport addresses involving IPv6 addresses, etc.

dumpPacket yes

defines whether to display a hexadecimal dump of the raw SNMP requests sent and received by the application. This is equivalent to the **−d** option.

doDebugging (1|0)

turns on debugging for all applications run if set to 1.

debugTokens TOKEN[,TOKEN...]

defines the debugging tokens that should be turned on when *doDebugging* is set. This is equivalent to the **−D** option.

16bitIDs yes

restricts requestIDs, etc to 16-bit values.

The SNMP specifications define these ID fields as 32-bit quantities, and the Net-SNMP library typically initialises them to random values for security. However certain (broken) agents cannot handle ID values greater than 2^16 - this option allows interoperability with such agents.

clientaddr [<transport-specifier>:]<transport-address>

specifies the source address to be used by command-line applications when sending SNMP requests. See *snmpcmd(1)* for more information about the format of addresses.

This value is also used by **snmpd** when generating notifications.

clientRecvBuf INTEGER

specifies the desired size of the buffer to be used when receiving responses to SNMP requests. If the OS hard limit is lower than the *clientRecvBuf* value, then this will be used instead. Some platforms may decide to increase the size of the buffer actually used for internal housekeeping.

This directive will be ignored if the platforms does not support *setsockopt()*.

clientSendBuf INTEGER

is similar to *clientRecvBuf*, but applies to the size of the buffer used when sending SNMP requests.

noRangeCheck yes

disables the validation of varbind values against the MIB definition for the relevant OID. This is equivalent to the **−Ir** option.

This directive is primarily relevant to the **snmpset** command, but will also apply to any application that calls *snmp_add_var()* with a non-NULL value.

noTokenWarnings

disables warnings about unknown config file tokens.

reverseEncodeBER (1|yes|true|0|no|false)

controls how the encoding of SNMP requests is handled.

The default behaviour is to encode packets starting from the end of the PDU and working backwards. This directive can be used to disable this behaviour, and build the encoded request in the (more obvious) forward direction.

It should not normally be necessary to change this setting, as the encoding is basically the same in either case - but working backwards typically produces a slightly more efficient encoding, and hence a smaller network datagram.

dontLoadHostConfig (1|yes|true|0|no|false)

Specifies whether or not the host-specific configuration files are loaded. Set to "true" to turn off the loading of the host specific configuration files.

retries INTEGER

Specifies the number of retries to be used in the requests.

timeout INTEGER

Specifies the timeout in seconds between retries.

SNMPv3 SETTINGS

defSecurityName STRING

defines the default security name to use for SNMPv3 requests. This can be overridden using the **−u** option.

defSecurityLevel noAuthNoPriv|authNoPriv|authPriv

defines the default security level to use for SNMPv3 requests. This can be overridden using the **−l** option.

If not specified, the default value for this token is *noAuthNoPriv*.

Note: *authPriv* is only available if the software has been compiled to use the OpenSSL libraries.

defPassphrase STRING

defAuthPassphrase STRING

defPrivPassphrase STRING

define the default authentication and privacy pass phrases to use for SNMPv3 requests. These can be overridden using the **−A** and **−X** options respectively.

The **defPassphrase** value will be used for the authentication and/or privacy pass phrases if either of the other directives are not specified.

defAuthType MD5|SHA

defPrivType DES|AES

define the default authentication and privacy protocols to use for SNMPv3 requests. These can be overridden using the **−a** and **−x** options respectively.

If not specified, SNMPv3 requests will default to MD5 authentication and DES encryption.

Note: If the software has not been compiled to use the OpenSSL libraries, then only MD5 authentication is supported. Neither SHA authentication nor any form of encryption will be available.

defContext STRING

defines the default context to use for SNMPv3 requests. This can be overridden using the **−n** option.

If not specified, the default value for this token is the default context (i.e. the empty string "").

defSecurityModel STRING

defines the security model to use for SNMPv3 requests. The default value is "usm" which is the only widely used security model for SNMPv3.

defAuthMasterKey 0xHEXSTRING

defPrivMasterKey 0xHEXSTRING

defAuthLocalizedKey 0xHEXSTRING

defPrivLocalizedKey 0xHEXSTRING

define the (hexadecimal) keys to be used for SNMPv3 secure communications. SNMPv3 keys are frequently derived from a passphrase, as discussed in the *defPassphrase* section above. However for improved security a truely random key can be generated and used instead (which would normally has better entropy than a password unless it is amazingly long). The directives are equivalent to the short-form command line options **–3m**, **–3M**, **–3k**, and **–3K**.

Localized keys are master keys which have been converted to a unique key which is only suitable for on particular SNMP engine (agent). The length of the key needs to be appropriate for the authentication or encryption type being used (auth keys: MD5=16 bytes, SHA1=20 bytes; priv keys: DES=16 bytes (8 bytes of which is used as an IV and not a key), and AES=16 bytes).

sshtosnmpsocket PATH

Sets the path of the **sshtosnmp** socket created by an application (e.g. snmpd) listening for incoming ssh connections through the **sshtosnmp** unix socket.

sshtosnmpsocketperms MODE [OWNER [GROUP]]

Sets the mode, owner and group of the **sshtosnmp** socket created by an application (e.g. **snmpd**) listening for incoming ssh connections through the **sshtosnmp** unix socket. The socket needs to be read/write privileged for SSH users that are allowed to connect to the SNMP service (VACM access still needs to be granted as well, most likely through the TSM security model).

sshusername NAME

Sets the SSH user name for logging into the remote system.

sshpubkey FILE

Set the public key file to use when connecting to a remote system.

sshprivkey FILE

Set the private key file to use when connecting to a remote system.

SERVER BEHAVIOUR

persistentDir DIRECTORY

defines the directory where **snmpd** and **snmptrapd** store persistent configuration settings.

If not specified, the persistent directory defaults to /var/lib/snmp

noPersistentLoad yes

noPersistentSave yes

disable the loading and saving of persistent configuration information.

Note: This will break SNMPv3 operations (and other behaviour that relies on changes persisting across application restart). Use With Care.

tempFilePattern PATTERN

defines a filename template for creating temporary files, for handling input to and output from external shell commands. Used by the *mkstemp()* and *mktemp()* functions.

If not specified, the default pattern is `"/tmp/snmpdXXXXXX"`.

serverRecvBuf INTEGER

specifies the desired size of the buffer to be used when receiving incoming SNMP requests. If the OS hard limit is lower than the *serverRecvBuf* value, then this will be used instead. Some platforms may decide to increase the size of the buffer actually used for internal housekeeping.

This directive will be ignored if the platforms does not support *setsockopt().*

serverSendBuf INTEGER

is similar to *serverRecvBuf*, but applies to the size of the buffer used when sending SNMP responses.

MIB HANDLING

mibdirs DIRLIST

specifies a list of directories to search for MIB files. This operates in the same way as the **−M** option - see *snmpcmd(1)* for details. Note that this value can be overridden by the **MIBDIRS** environment variable, and the **−M** option.

mibs MIBLIST

specifies a list of MIB modules (not files) that should be loaded. This operates in the same way as the **−m** option - see *snmpcmd(1)* for details. Note that this list can be overridden by the **MIBS** environment variable, and the **−m** option.

mibfile FILE

specifies a (single) MIB file to load, in addition to the list read from the *mibs* token (or equivalent configuration). Note that this value can be overridden by the **MIBFILES** environment variable.

showMibErrors (1|yes|true|0|no|false)

whether to display MIB parsing errors.

commentToEOL (1|yes|true|0|no|false)

whether MIB parsing should be strict about comment termination. Many MIB writers assume that ASN.1 comments extend to the end of the text line, rather than being terminated by the next "−−" token. This token can be used to accept such (strictly incorrect) MIBs.
Note that this directive was previous (mis-)named *strictCommentTerm*, but with the reverse behaviour from that implied by the name. This earlier token is still accepted for backwards compatibility.

mibAllowUnderline (1|yes|true|0|no|false)

whether to allow underline characters in MIB object names and enumeration values. This token can be used to accept such (strictly incorrect) MIBs.

mibWarningLevel INTEGER

the minimum warning level of the warnings printed by the MIB parser.

OUTPUT CONFIGURATION

logTimestamp (1|yes|true|0|no|false)

Whether the commands should log timestamps with their error/message logging or not. Note that output will not look as pretty with timestamps if the source code that is doing the logging does incremental logging of messages that are not line buffered before being passed to the logging routines. This option is only used when file logging is active.

printNumericEnums (1|yes|true|0|no|false)

Equivalent to **−Oe**.

printNumericOids (1|yes|true|0|no|false)

Equivalent to **−On**.

dontBreakdownOids (1|yes|true|0|no|false)

Equivalent to **−Ob**.

escapeQuotes (1|yes|true|0|no|false)

Equivalent to **−OE**.

quickPrinting (1|yes|true|0|no|false)

Equivalent to **−Oq**.

printValueOnly (1|yes|true|0|no|false)

Equivalent to **−Ov**.

dontPrintUnits (1|yes|true|0|no|false)

Equivalent to **−OU**.

numericTimeticks (1|yes|true|0|no|false)
Equivalent to **−Ot**.

printHexText (1|yes|true|0|no|false)
Equivalent to **−OT**.

hexOutputLength integer
Specifies where to break up the output of hexadecimal strings. Set to 0 to disable line breaks. Defaults to 16.

suffixPrinting (0|1|2)
The value 1 is equivalent to **−Os** and the value 2 is equivalent to **−OS**.

oidOutputFormat (1|2|3|4|5|6)
Maps −O options as follow: −Os=1, −OS=2, −Of=3, −On=4, −Ou=5. The value 6 has no matching −O option. It suppresses output.

extendedIndex (1|yes|true|0|no|false)
Equivalent to **−OX**.

noDisplayHint (1|yes|true|0|no|false)
Disables the use of DISPLAY-HINT information when parsing indices and values to set. Equivalent to **−Ih**.

FILES

System-wide configuration files:
/etc/snmp/snmp.conf
/etc/snmp/snmp.local.conf

User-specific configuration settings:
$HOME/.snmp/snmp.conf
$HOME/.snmp/snmp.local.conf

Destination host specific files:
/etc/snmp/hosts/HOSTNAME.conf
$HOME/.snmp/hosts/HOSTNAME.conf

SEE ALSO

snmp_config(5), netsnmp_config_api(3), snmpcmd(1).