F

Events Commands

The following pages are the UNIX-equivalent manpages for the various Events commands. This section describes the command-line options for each command and what that command will do. The order of these is:

- AgentENL (8N)
- AgentMon (8N)
- EventsCli (8N)
- FieldCk
- testtab (1)

AgentENL (8N)

AgentENL - dynamically supports multiple SMUX subagents

SYNOPSIS

AgentENL [config_file]

DESCRIPTION

AgentENL is an SNMP/SMUX-compliant agent daemon. Subagents may be dynamically registered and de-registered without having to bring the AgentENL down. The UDP and IP port numbers defined within the relevant standards will be used unless the file /etc/services contains entries specifying alternate ports.

COMMUNITY entries determine which community strings are valid. If no COMMUNITY entries are present, ANY community will be accepted by the agent. For each community, the set of supported operations is specified by the ALLOW clause.

The MEMBERS clause in a community entry restricts the set of managers allowed to use that community. If the MEMBERS clause is absent from a COMMUNITY entry, any manager may use that community.

MANAGER entries determine where traps should be sent. If the PORT clause is absent, then TRAPS are sent to port 162, as specified in RFC 1157. If the WITH COMMUNITY clause is absent, the community string “public” is used.

OPTIONS

config_file

The filename (or pathname) of the configuration file. The configuration file should be edited to specify the SNMP manager hostnames, Community names, and privileges. The default is to use the configuration file $ENLIGHTEN/config/AgentENL.config, where $ENLIGHTEN represents the top-level product installation directory.

DIAGNOSTICS

Complains and exits if no configuration file is found or if the socket/port is already in use.

AgentMon (8N)

AgentMon - workstation monitoring program

SYNOPSIS

AgentMon [-t testname]

AgentMon -x

DESCRIPTION

AgentMon monitors tests as defined in the text file testtab.{HOSTNAME}. This testtab file is expected to be in $ENLIGHTEN/config, where $ENLIGHTEN represents the top-level product installation directory. Each test may also be remotely monitored and managed via SNMP. When a test value exceeds one or more of the user-defined thresholds, AgentMon may emit an Enterprise-Specific Trap, send an alarm via email, start a user-specified program that could possibly take corrective action, and/or notify PEP.

The local user may define and alter tests by editing testtab. {HOSTNAME}. The network manager may manage tests by using the SNMP SET commands from almost any SNMP-compliant network management application. All enterprise-specific traps may be individually enabled and disabled via SNMP. AgentMon does not rely on SNMP and will run without hindrance in non-SNMP environments.

OPTIONS

-t testname

Disregard the normal testing interval for the named test and run the test every 15 seconds. (See RESTRICTIONS later in this section.)

-x

Causes AgentMon to print (to stdout) its impression of the current test's configuration. The output is identical in form to the testtab file and represents the logical sum of the test defaults and the current contents of the testtab file. This is a good way of verifying AgentMon understands your testtab edits.

NOTES

The testtab file does not exist prior to the first invocation of AgentMon. It will be created immediately after start-up, however.

DIAGNOSTICS

Complains and exits if a connection with the AgentENL (or other agent) cannot be made. AgentMon will exit gracefully if communication with the agent goes away.

RESTRICTIONS

Only one test may be specified with the -t option. This is intended to be primarily a systems troubleshooting aid. The -t option has no effect on process monitoring; processes are always checked every 30 seconds.

SEE ALSO

AgentENL.8N testtab.5

EventsCli (8N)

EventsCli - Command Line Interface for sending alarms to PEP, SNMP, and so on.

SYNOPSIS

```
EventsCli -u units -v measured_value -n name -s severity
-t trap_num -T threshold
```

DESCRIPTION

EventsCli is a user interface to the Events alarm dispatching facilities. It can be used to send SNMP traps, notify PEP, and log events. These events can be routed from external processes that call EventsCli each time a condition warrants reporting.

EventsCli is intended to be an interface between your existing system monitoring programs and scripts. These programs and scripts can call EventsCli and then rely on the SNMP traps, Events data logging, and PEP to take the corrective action you desire.

OPTIONS

-u units

The units value specifies the unit of measure for the value measured.

-v measured_value

The value being reported. This can be any data type including a short string.

-n name

A brief name, which must be quoted if it contains white space. This could be a test name or other descriptive text.

-s setting state

Setting state describes the severity values; for example, “high”, “low”, or “OK”.

-s severity

Severity is a number between a low of 1 and a high of 5.

-t rap_num

Events has 12 general-purpose TRAPs for EventsCli. These are numbered from 500 to 511. The trap number you choose can be arbitrary, but must be within the range indicated.

-T threshold

The name of the threshold exceeded. This is a very short description of what happened; that is, “low alarm threshold.” In real life, any brief text is acceptable.

EXAMPLE

EventsCli -n DatabaseA -T “high tps” -u “tps” -v 12345 -s 5 -t 501

would produce the following comment and report:

DatabaseA is doing 12345 tps and has exceeded the high tps threshold. The event will be reported with the highest severity level (5) and will be sent using trap number 501.

RESTRICTIONS

EventsCli must be invoked on a machine that has AgentMon running.

FieldCk

FieldCk - a simple utility for verifying an API test.

SYNOPSIS

FieldCk

DESCRIPTION

When configuring an API test, you must specify a filename, which field (or column) contains the data, and which field (or column) contains the optional label. To check your data entry, you may use FieldCk to read the file and display the data and label fields.

FieldCk is started without options and will prompt you for all needed information. The file will be parsed with the fields you designated and the check-results will be sent to stdout.

If FieldCk parses the file correctly, you may be assured AgentMon will also do so.

testtab (1)

testtab - AgentMon capability database

SYNOPSIS

$ENLIGHTEN/config/testtab.{HOSTNAME}

DESCRIPTION

testtab is the capability database for the AgentMon subagent. The format is similar to other UNIX system administrative files, but is simpler to use. AgentMon accesses this database when it starts up and frequently checks it for any changes.

At time of installation, this file may not yet exist. When no testtab file exists, all default tests and all test capability defaults are assumed and the testtab file is automatically created.

Each entry in this file represents a test description. A test description consists of a line containing the test name followed by additional lines containing a list of keywords and, possibly, values. There are three types of keywords (capabilities):

Boolean

Capabilities that have only two states (on/off, enable/disable, etc.) are specified by simply using that capability name to activate it or by prepending the name with an exclamation point ("!") to deactivate it.

Numerics and
Strings

A capability that requires a numeric (or alphabetic) value is specified by separating the capability name and value with an equal sign.

All capabilities are delimited by colons (":"). Values that must contain a colon must be escaped with a backslash. Any test description line not terminated with a backslash ("\") is assumed to be the last line of that test description.

CAPABILITIES

Name	Type	Default	Comment
testfreq	int	5	Minimum testing interval (in minutes).
alarmfreq	int	60	Minimum time between alarms (in minutes).
notify	str	root	Specifies who receives any alarms.
mailer	str	/bin/mail	Specifies which program will deliver the alarm if a username was given in the notify field.
log	boolean	false	Indicates whether logging is enabled or not.
!log	boolean	true	Indicates logging is disabled.
delta	int/float	0	If logging is enabled, the most recent value measured will be logged if it differs by at least this amount from the previously logged value.
pep	boolean	false	If true, alarms are also sent to PEP.
age	int	0	For monitoring queues only. If not zero, only files more than “age” minutes old are counted as “old.”
high	int/float	0	High-level alarm set point.
low	int/float	0	Low-level alarm set point.
+rate	float	0.0	Positive rate change set point.
- rate	float	0.0	Negative rate change set point.

Name	Type	Default	Comment
+jump	int/float	0	Positive jump set point.
jump	int/float	0	Negative jump set point.
command	str	null	This is a pathname to a script/executable.
regx_	R/E	null	regx1-regx31 are for File Clamping tests. They hold lists of regular expressions used in the test.
units	str	“units”	These values are preset and “changes” are silently ignored.

NOTES

Most tests measure integer values. Some measure floating-point values. Set point parameters should be given by using the same variable type as the test result. However, failure to comply will only result in rounding errors.

The jump and rate thresholds compare the current test value with the last measured value. Therefore, they are comparing change over time (as defined by testfreq).

The values for the unit’s capability are built in and cannot be changed. It is shown here for completeness, as AgentMon includes this field whenever it re-writes the testtab file and whenever it is started with the -x command line option.

If the notify field is set to “nobody”, that is, :notify=nobody:, then no email will be sent.