eTrap

Software manual

Table of Contents

Introduction

eTrap is a windows service that converts the windows events into SNMP traps. The eTrap SNMP traps can be easily integrated into monitoring systems like NAGIOS to be able to monitor the windows systems and their services based on the generated windows events. The windows events that are to be converted can be effectively filtered, so no unimportant events are sent out as SNMP traps.

eTrap provides a simple and effective solution to keep your Windows based systems under control and to be able to monitor them in a very detailed way.

System Requirements

Operating system

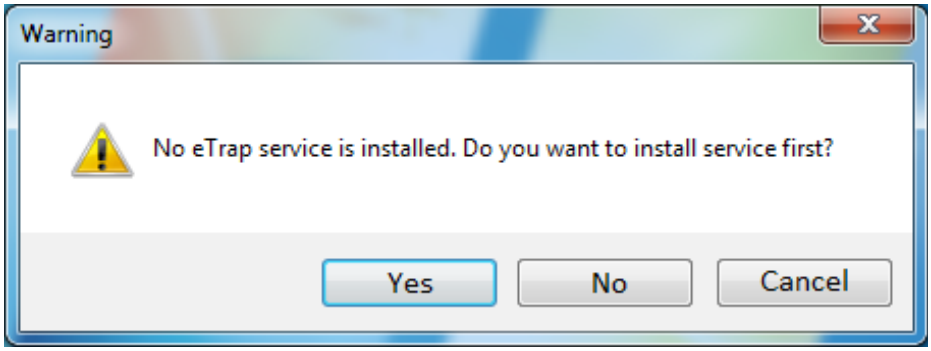
- Windows XP
- Windows Vista
- Windows 7
- Windows 8
- Windows 8.1
- Windows 10
- Windows Server 2003
- Windows Server 2008
- Windows Server 2012
- Windows Server 2016

.NET

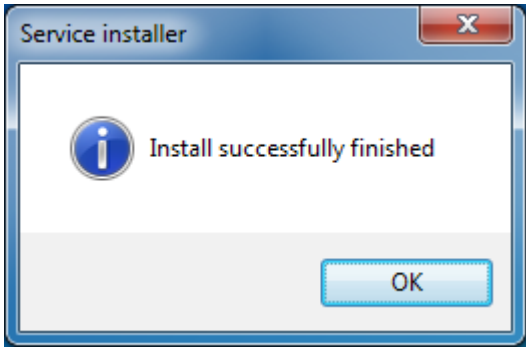
- 2.0 or above

Installation

Download eTrap application from
http://download.smartoservice.com/etrap/etrap_configurator.exe
Run etrap_configurator.exe as Administrator on the computer that needs to be monitored. eTrap notifies you to install the service on its first run.



Use "yes" to install the service. The installer notifies you when the installation finished.



Configuration

Run etrap_configurator.exe as Administrator on the computer to be configured to change the settings of the eTrap service.

Settings tab

Use settings tab to add the SNMP Trap destination host(s) (ip address or domain name). It is usually your NAGIOS monitoring server.

Select the windows event logs that should be monitored. Only events that are generated in one of the selected logs will be converted into SNMP Traps.

You can fine tune the behavior of the service defining extra parameters under *Advanced settings* section:

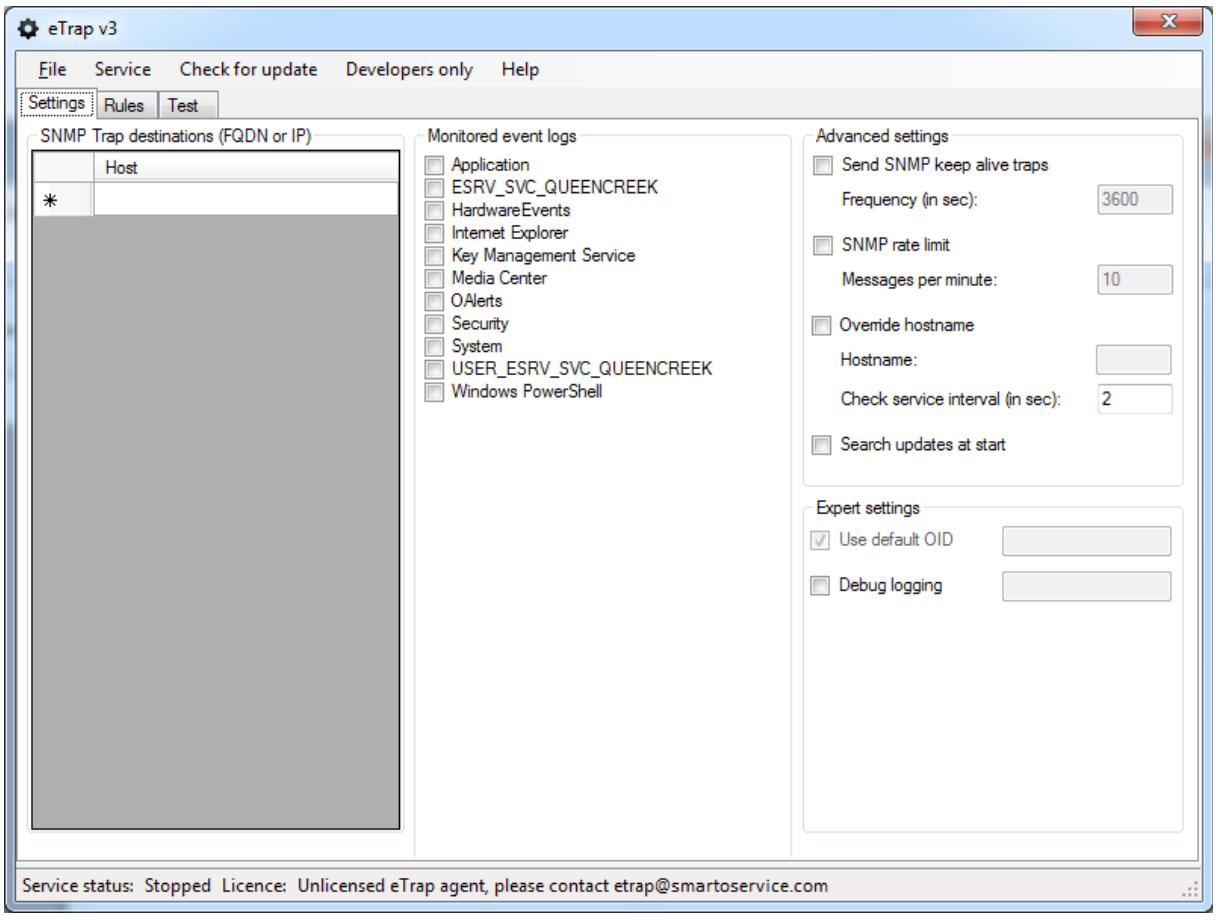
Send SNMP keep alive traps Frequency (in sec): If given, eTrap service generates special keep-alive traps with the given frequency. This makes it possible to monitor the health of eTrap service itself.

SNMP rate limit (max message per minute): If given, the SNMP Traps from the same source and with the same event id will be suppressed after the given limit reached.

Override hostname Hostname: eTrap normally uses the computer name it runs on as the source parameter in the SNMP Trap. This can be overridden with this field.

Search updates at start: If selected eTrap application will check for updates on every start.

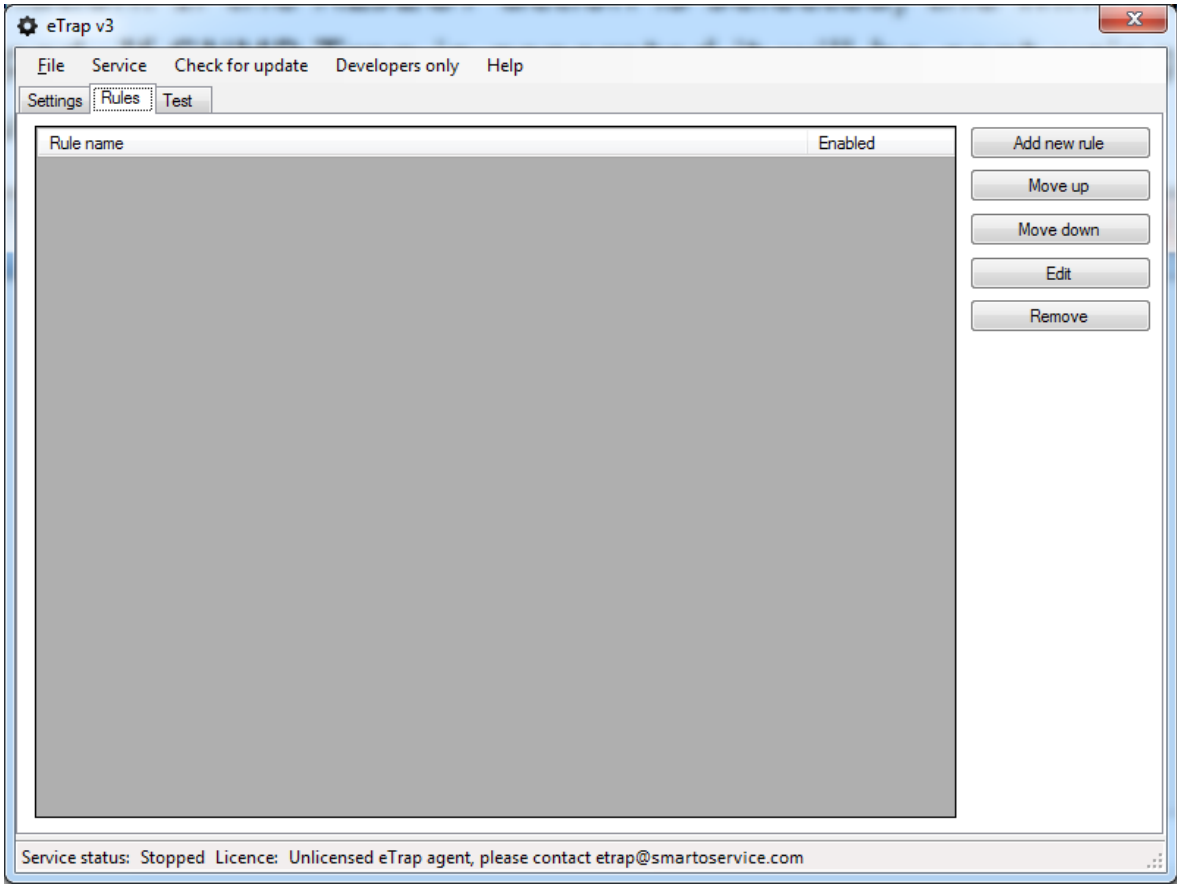
Debug logging: If selected and the input field contains a valid file name eTrap application will write debug information into the given file.



Rules tab

eTrap uses rules to determine which windows events should be converted to SNMP traps. If no rules are given, no windows events are converted to SNMP Traps.

After the installation a single default rule is generated that lets all the events converted to SNMP Traps.

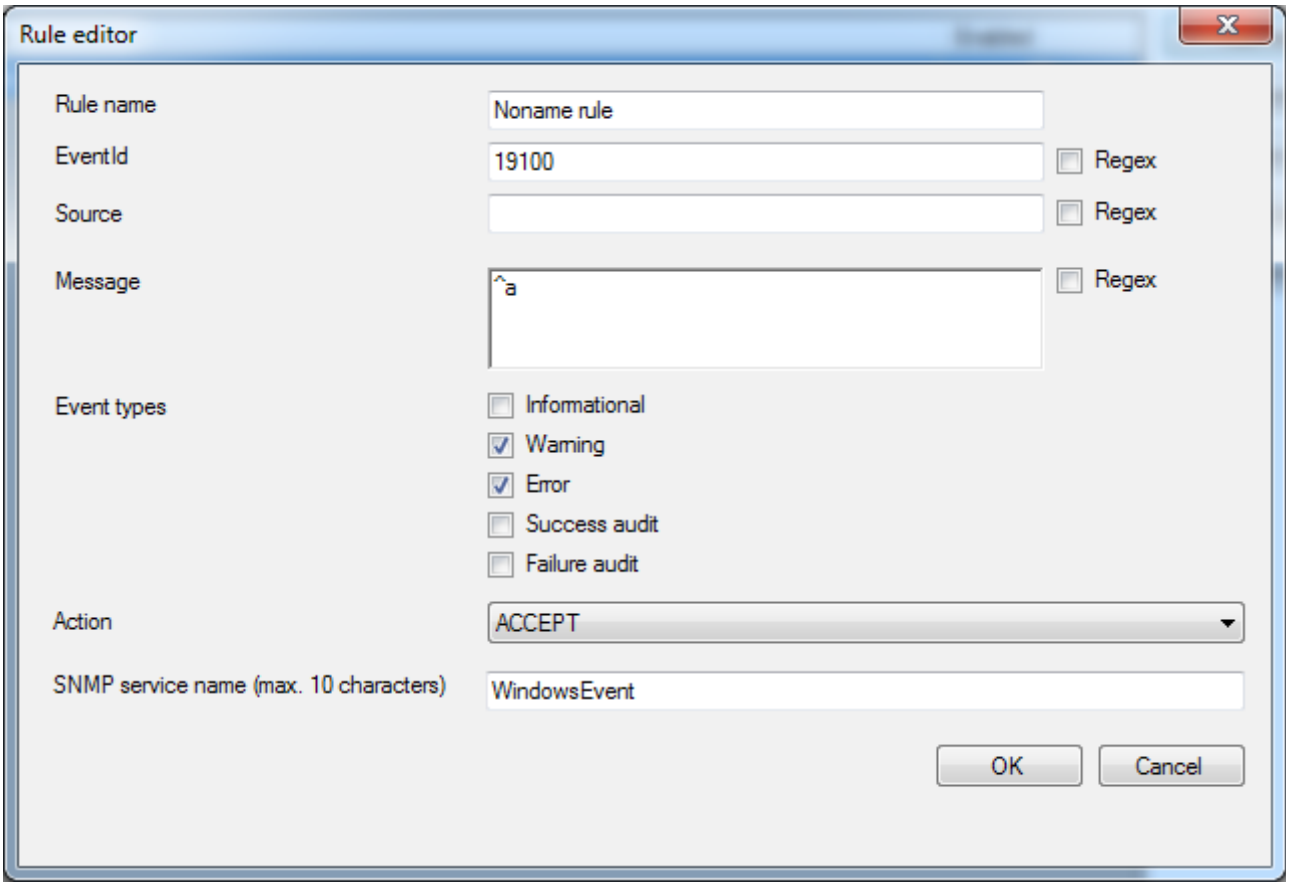


The rules are checked top to bottom. If a rule applies the rest of the rules are ignored.
To add/edit/remove or arrange the rules use the appropriate button on the right.
When editing or adding a new rule you can use filters for

- Event id
- Source
- Message
- Event Types

The rule matches only if ALL the filters are matched.
If the rule matches the selected action (*ACCEPT* or *REJECT*) will be applied.

If the action is *ACCEPT* the Windows event will be converted to SNMP Trap and send out to the trap destination(s). If the *REJECT* action is selected, the windows event will be silently dropped.



For the filters the following alternatives can be given:

> "Empty string" always matches.
>
> If the string typed into the filter field can be found anywhere in the corresponding field of the event and the "*regex*" check box is not checked the filter matches.
>
> The string typed into the filter field interpreted as a "regular expression" if "*regex*" check box is checked. This regular expression is to be applied on the corresponding field of the event. If the regular expression matches, the filter matches too.

The filters are case insensitive.

Examples

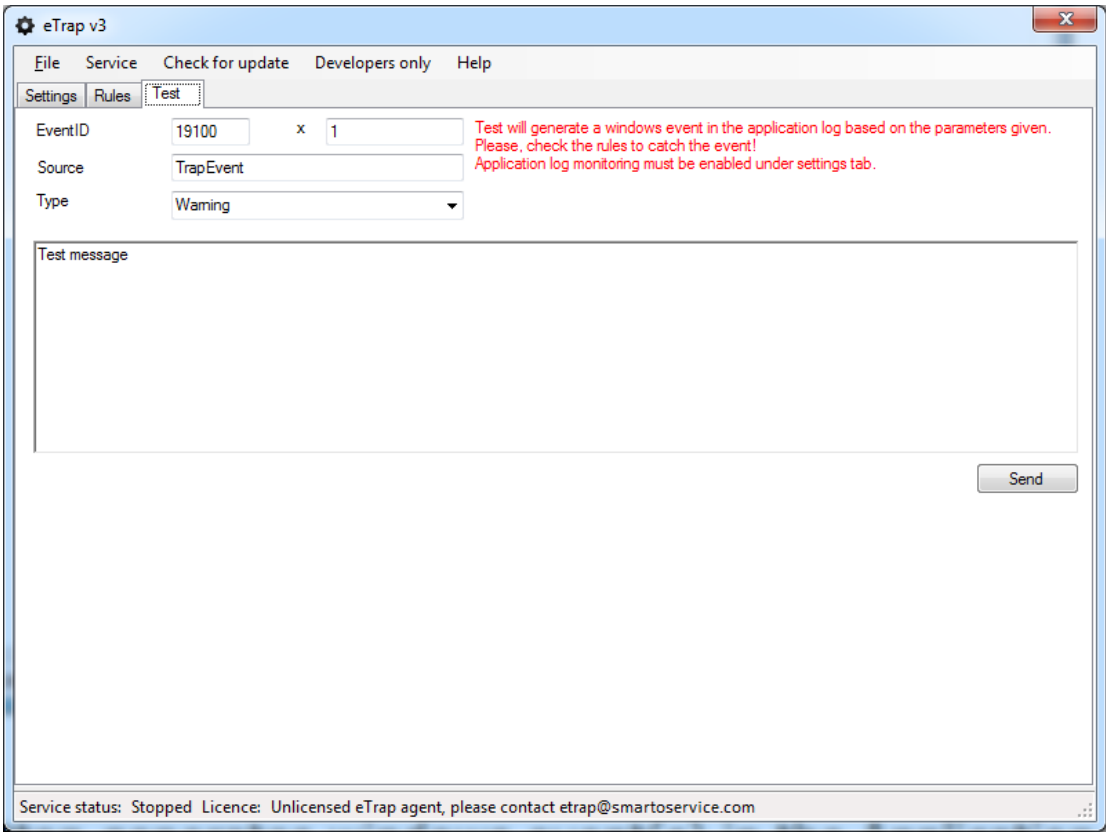
Examples of filters for the *EventId* field can be found in the table below. (filters for the *Source* and the *Message* fields can be constructed the same way):

Description	Content of the EventID field	Regular expression check box
Filter that matches Event Id 19100	19100	Not checked
Filter that matches Event Id that contains the number of "1" (like 1, 21, 29100)	1	Not checked
Filter that matches Event Id that starts with number of "1" (like 1, 12, 19100)	^1.*	Checked
Filter that matches both Envet Id 19100 and 19200.	19100|19200	Checked

If SNMP Trap is generated it will be sent using the Service name that was given in the *SNMP service name* field (ex.: eTrapWarning). The default value for the service name is "WindowsEvent".

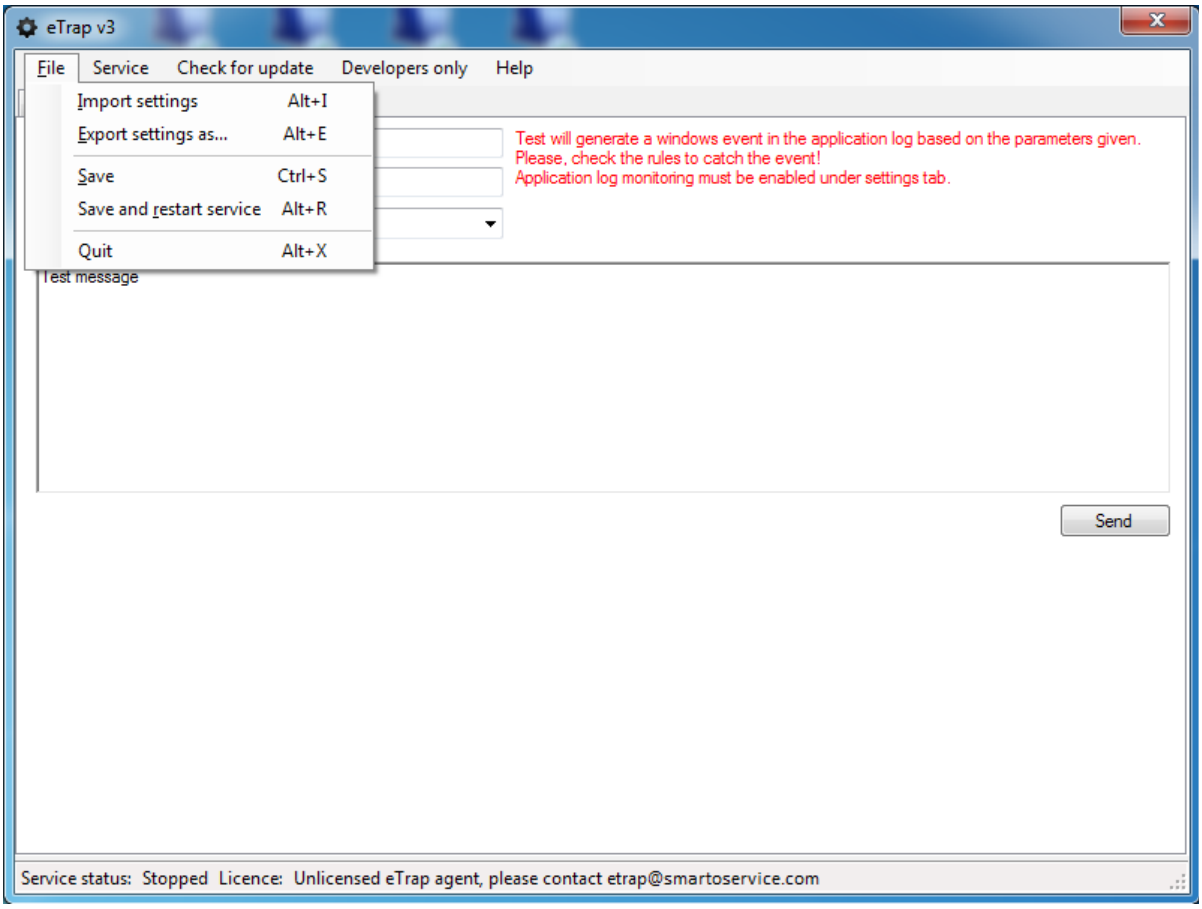
Test tab

The test tab can be used to test the functionality of the eTrap service and its rules.



Clicking on the "Send" button you can generate windows event(s) in the Application log. The Event will be generated with the given *Event ID*, *Source*, *Type* and *Message*. Please note that the event will be generated in the Application log, so only rules that are applies on the Application log will be checked. Also the Application log has to be selected on the settings tab to let the eTrap service to monitor the test messages. When a test message sent, and eTrap service find an appropriate rule matching, it will send out the SNMP trap or ignore the message based on the Action field of the matching rule.

File menu

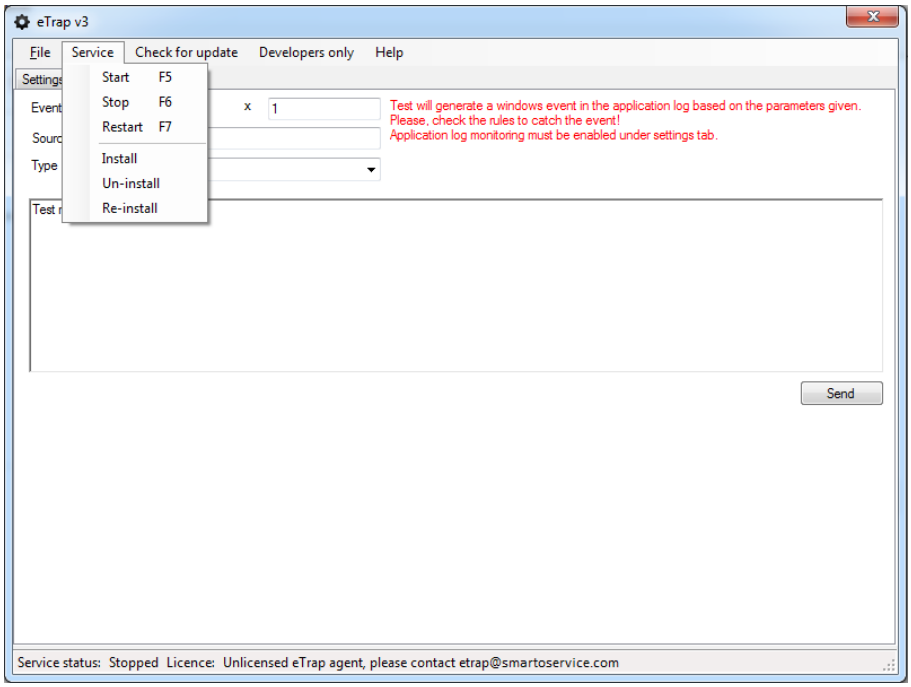


Using the file menu we can export the current settings (from the Settings and Rules tab) into a file. If TXT format is selected for the export, only the rules are exported.

Previously exported settings can be loaded by selecting **File → import settings** menu point.

All the modifications of the settings and the rules are applied only on saving them. Using **File → Save or file → Save and restart service** menu point we can apply unsaved changes. If the application is closed with unsaved changes, a warning message appears and user can save & apply changes before quitting the program.

Service menu

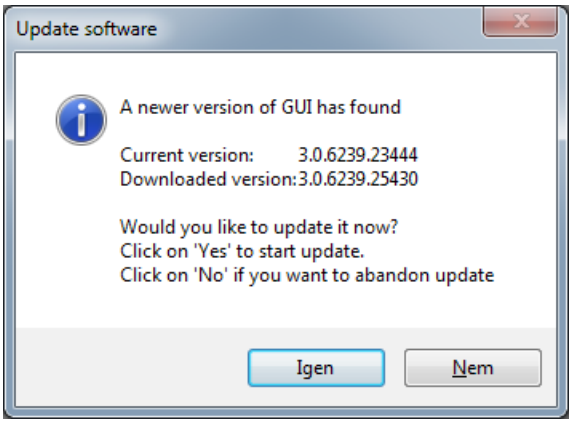


Service menu can be used to start/stop/restart the service. If any modification has been saved a service restart is needed for the new settings to be activated.

Services menu provides possibility for installing, re-installing or removing the service from the computer completely.

Check for update menu

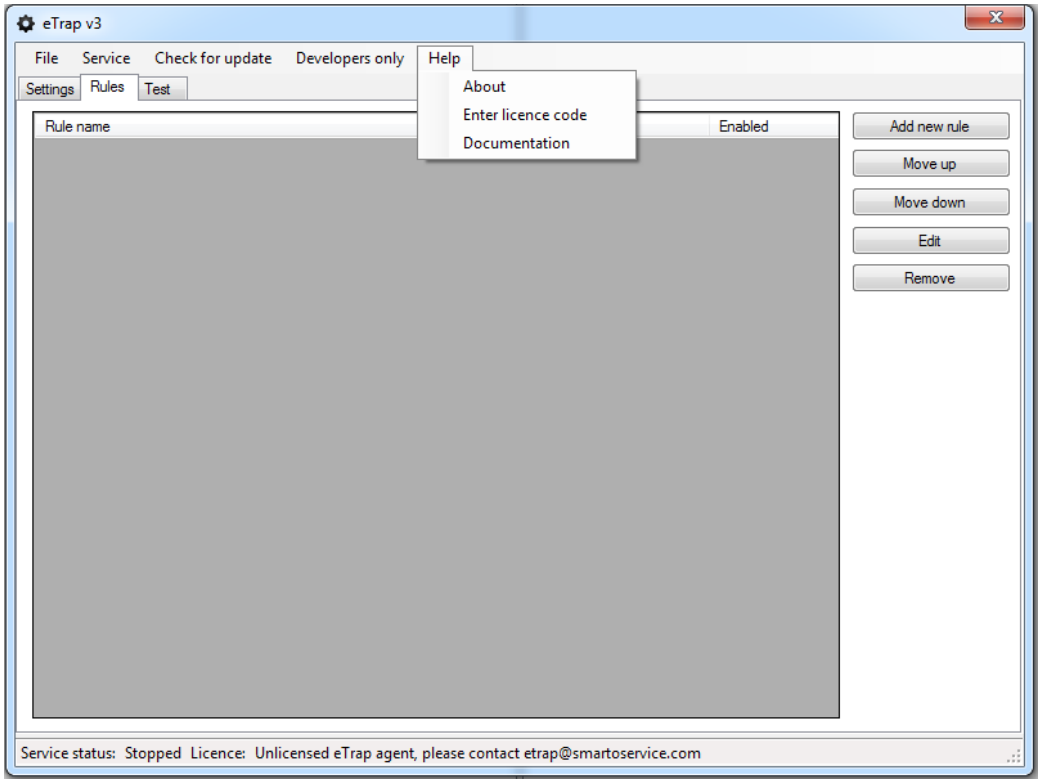
If the computer has internet access and an update is available the following window appears:



By clicking on the YES button, the software is updated.

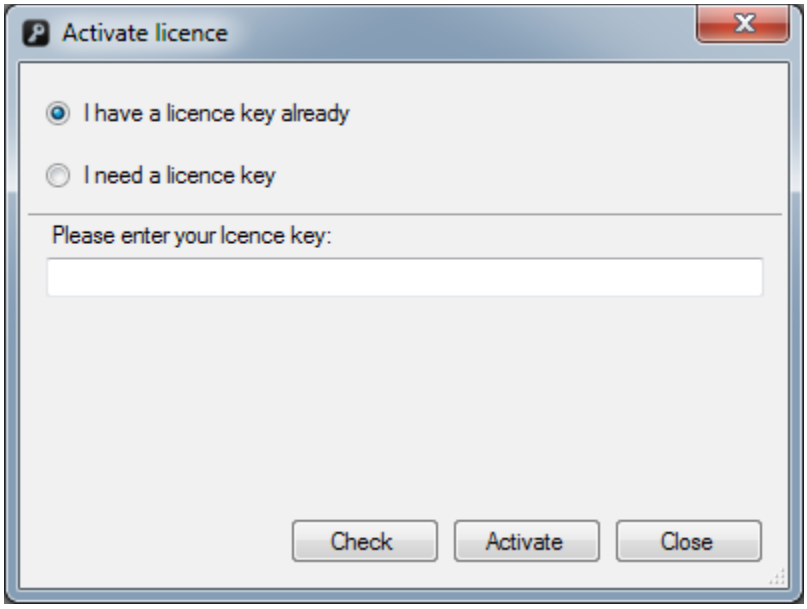
Help menu

You can find general information about the software under **Help → About** menu. The actual version of the user's manual is available in PDF from **Help → Documentation**.

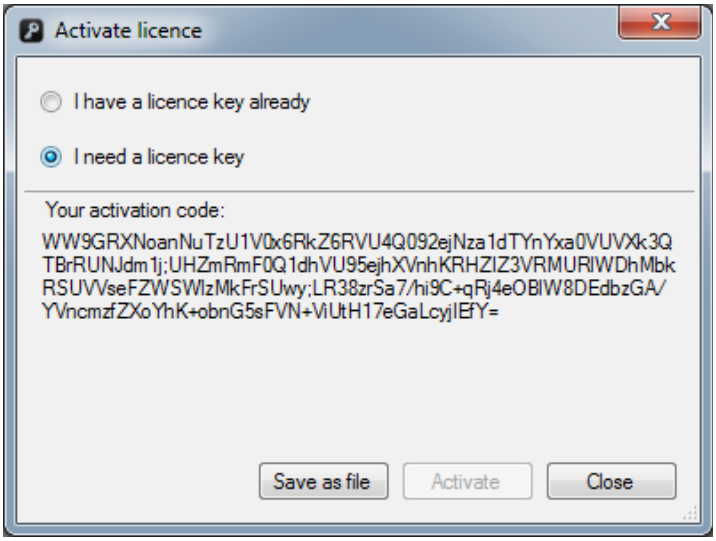


The eTrap service can be used in free or advanced modes.
The free mode is completely free and can be used without any restriction, but comes as is; without any warranty. The free and advenced modes of the eTrap service are identical except that the free mode can handle only 5 rules and sends SNMP Traps with the message "To see the message of the event please visit http://www.smartoservice.com/licenses", while the advanced mode handles arbitrary number of rules and sends the Message of the originating Windows event.

You can use the **Help → Enter license code** menu point to enter or change license code for Advanced mode.



To acquire a license code for Advanced mode you have to select *I need a license key* option to generate an activation key.



You can copy the key or save it into a file. This activation key is necessary to buy an Advenced license key for the eTrap service running on the Windows host.
To buy the Advanced license please, visit http://www.smartoservice.com/licenses and follow the instructions given on the page.

NAGIOS integration

eTrap Nagios integration can be achieved in many different ways. Here you can find a suggested method for a standalone NAGIOS monitoring server that runs on Linux; a best practice that has been developed for many years now.

The suggested solution for NAGIOS integration uses *snmptrapd* and *snmptt* services to receive and convert eTrap traps to NAGIOS checks.

Requisites

A linux server with a NAGIOS instance up and running.
The following packages (or equivalent, depending on linux distribution being used):

- nagios-plugins-snmp
- net-snmp
- net-snmp-utils
- snmptt

Configuration

The suggested configuration files are the followings:
/etc/snmp/snmptrapd.conf

```
#Example configuration file for snmptrapd
#authCommunity  TYPES           COMMUNITY  [SOURCE [OID | -v VIEW ]]
authCommunity   log,execute,net public
traphandle default /usr/sbin/snmptthandler
disableAuthorization    yes
#donotlogtraps           yes
```

/etc/snmp/snmptt.eTrap.conf

```
#SNMPTT config file for eTrap
EVENT event .1.3.6.1.4.1.29037.8.9.0.1 "Status Events" Normal
FORMAT This trap is sent when a windows event should be forwarded $*
EXEC /usr/local/sbin/submit_check_result $1 $2 $3 “$4 $5”
SDESC
This trap is sent when a windows event should be forwarded
Variables:
  1: iMooLTrapSource
  2: iMooLTrapService
  3: iMooLTrapSeverity
  4: iMooLTrapTimeStamp
  5: iMooLTrapInfo
EDESC
#
EVENT keepalive .1.3.6.1.4.1.29037.8.9.0.2 "Status Events" Normal
FORMAT This trap is sent as keepalive message $*
EXEC /usr/local/sbin/submit_check_result $1 $2 $3 “$4 $5”
SDESC
```

```
This trap is sent as keepalive message
Variables:
  1: iMooLTrapSource
  2: iMooLTrapService
  3: iMooLTrapSeverity
  4: iMooLTrapTimeStamp
  5: iMooLTrapInfo
EDESC
```

The /etc/snmp/snmptrapd.conf should be extended with the reference to /etc/snmp/snmptt.conf.eTrap file.
Here you can find the extra line in yellow and the place where that line should be inserted:

```
snmptt_conf_files = <<END
...
/etc/snmp/snmptt.conf.eTrap
...
END
```

Please note, that a bug exists in several releases of snmptt causes misinterpretation of severity, so we suggest to change the following line:

```
net_snmp_perl_cache_enable = 1
```

to

```
net_snmp_perl_cache_enable = 0
```

/usr/local/sbin/submit_check_result

```
#!/bin/sh

# SUBMIT_CHECK_RESULT
# Written by Ethan Galstad (egalstad@nagios.org)
# Last Modified: 02-18-2017
#
# This script will write a command to the Nagios command
# file to cause Nagios to process a passive service check
# result. Note: This script is intended to be run on the
# same host that is running Nagios. If you want to
# submit passive check results from a remote machine, look
# at using the nsca addon.
#
# Arguments:
# $1 = host_name (Short name of host that the service is
# associated with)
# $2 = svc_description (Description of the service)
# $3 = return_code (An integer that determines the state
# of the service check, 0=OK, 1=WARNING, 2=CRITICAL,
# 3=UNKNOWN).
# $4 = plugin_output (A text string that should be used
# as the plugin output for the service check)
#

echocmd="/bin/echo"
CommandFile="/var/spool/nagios/cmd/nagios.cmd"

# get the current date/time in seconds since UNIX epoch
```

```
datetime=`date +%s`

HOST="$1"

#converting textual severity into numeric if necessary
SEV=0
case "$3" in
        warning)
                SEV=1
                ;;
        critical)
                SEV=2
                ;;
        info)
                SEV=0
                ;;
        *)
                SEV="$3"
                ;;
esac

# create the command line to add to the command file
cmdline="[$datetime] PROCESS_SERVICE_CHECK_RESULT;$1;$2;$SEV;$4"

# append the command to the end of the command file
`$echocmd $cmdline >> $CommandFile`
```

Note: The /usr/local/sbin/submit_check_result script must be executable (`chmod a+x /usr/local/sbin/submit_check_result`).

A sample NAGIOS service definition:
/etc/nagios/conf.d/winserver.contoso.com.cfg

```
define host{
        use                             generic-host
        name                            windowsserver
        host_name                       windowsserver.contoso.com; hostaname
        alias                           windowsserver            ; alias
        address                         192.168.0.1              ; IP address
        check_period                    24x7                     ; non-stop
        max_check_attempts              2                        ;
        check_command                   check-host-alive         ;
        notification_period             24x7                     ;
        notification_interval           120                      ;
        notification_options            d,u,r                    ;
        contact_groups                  admins                   ;
        }

define service{
        name                            WindowsEvent
        service_description             WindowsEvent
        check_period                    none
        max_check_attempts              1
        active_checks_enabled           0
        passive_checks_enabled          1
        flap_detection_enabled          0
```

```
        is_volatile                     1
        parallelize_check               1
        obsess_over_service             0
        check_freshness                 0
        event_handler_enabled           1
        process_perf_data               1
        retain_status_information       1
        retain_nonstatus_information    1
        notification_interval           60
        notification_options            w,u,c,r,f
        notification_period             24x7
        notification_interval           1
        check_command                   check-host-alive
        contact_groups                  admins
        notification_interval           31536000
        }
```

The *snmptrapd, snmptt* and *nagios* services have to be restarted if their configuration were modified.

Frequently Asked Questions

What are the exact SNMP messages eTrap sends?

The MIB file that describes the eTrap SNMP TRAPs is available from the eTrap software under development menu point.

eTrap sends SNMP traps with OID: .1.3.6.1.4.1.29037.8.9.0.1. The TRAP contains the following variables:

1:Source – The hostname of the sender host (ex.: winserver.contoso.com)

2 Service – The name of the service the trap is sent (ex.: WindowsEvent)

3 Severity – The severity of the windows event (OK or Critical or Error)

4 TimeStamp – The time stamp of the windows event

5 Info – A concatenated string. (Containing the widows event source, event ID and event message if Advanced mode is used).

eTrap sends regular keep-alive SNMP traps with OID: .1.3.6.1.4.1.29037.8.9.0.2

1:Source – The hostname of the sender host (ex.: ad.contoso.com)

2 Service – The name of the service (always *etrap* for keep-alive)

3 Severity – The severity (always *OK* for keep-alive)

4 TimeStamp – The time stamp of the keep-alive message

5 Info – A concatenated string containing uptime information of the eTrap service

License information

Warranty

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright