EventReporter Configuration Documentation

version 19.0

Adiscon GmbH

November 12, 2025

Contents

About EventReporter

EventReporter is an integrated, modular, and distributed solution for system management.**

Microsoft Windows 10™, Windows 11™, Windows Server 2016™, Windows Server 2019™, and Windows Server 2022™ are highly capable operating systems (we will call all of them “Windows” in the following documentation). However, their standard event reporting mechanisms are rather limited. Administrators seeking complete control over their server environment need to regularly check the server event logs. Adiscon’s EventReporter provides central notification of any events logged to the Windows system event log. Messages can be delivered via email and syslog protocol.

The initial product - called EvntSLog - was specifically written with mixed Windows and UNIX environments in mind. It supported the syslog protocol only. It is currently in use by many large-scale commercial organizations, universities, and government bodies (like the military) all around the world. EventReporter empowers data center operators to integrate Windows event logs into their central syslog setup. Administrative duties and exception notification can easily be built via Unix-based scripting.

But small sized organizations also demanded relief from checking server logs. As such, EventReporter allows delivery of Windows event notifications via standard Internet email. Each server’s events are gathered, filtered according to rules set up by the administrator and - if they matter - forwarded to the admin. Especially small sized organizations operating a single server can be rest assured that they will not miss any important log entries.

EventReporter can be teamed with other MonitorWare line of products. In this scenario, it provides a totally centralized and automated event log collection, monitoring, and analysis solution. If you are looking for a solution that not only can forward event information but also monitor additional system settings, you might want to have a look at the MonitorWare Agent.

EventReporter is also a great tool for computer resellers, consultants, and other service providers in need to monitor their customer’s systems.

The product is easy to install and configure, uses only minimal system resources and is proven to be reliable. Furthermore, it is extremely inexpensive.

Manual

Introduction

In this first chapter we describe the features, components and system requirements.

Features

Centralized Logging

This is the key feature. EventReporter allows consolidation of multiple Windows event logs and forward them automatically to either a system process or an administrator.

Ease of Use

Using the new EventReporter client interface, the product is very easy to setup and customize. We also support full documentation and support for large-scale unattended installations.

Syslog Support

Windows Event Messages can be forwarded using standard Syslog protocol. Windows severity classes are mapped to the corresponding Syslog classes. syslog facility codes are fully supported.

SETP Support

setp was originally developed for MonitorWare but now it is a key feature added in EventReporter 6.2 Professional Edition. Windows Event Messages can be forwarded using SETP protocol. Click here for more information on SETP.

Email Support

Windows Event Log information can also be delivered via standard Internet email. This option is an enabler for smaller organizations or service providers unattended monitoring their client's servers.

Local Filtering

EventReporter can locally filter events based on the Windows Event Log type (e.g. "System" or "Application") as well as severity.

IPv6

Support for IPv6 is available in all network related facilities of the engine. All network related actions will automatically detect IPv6 and IPv4 target addresses if configured. You can also use DNS resolution to resolve valid IPv6 addresses. Network related Services can either use IPv4 or IPv6 as internet protocol. In order to support both protocols, you will need to create two services. The only exception is the RELP Listener, which uses IPv4 and IPv6 automatically if available.

Runs on a large Variety of Windows Systems

Windows 2019/2016/2012/10/8/2008(R2)/7/Vista/2008/2003/2003(R2)/XP/2000 Workstation or Server – MonitorWare Agent runs on all of them.

Support for End-of-Life operating systems is only partially available. Only a minimal service installation may be possible. More details: information for a mass rollout

Robustness

EventReporter is running in a large number of installations. It is written to perform robustly even under unusual circumstances. Its reliability has been proven at customers' side since 1997.

Remote Administration

The client interface can be used to remotely manage EventReporter instances.

Minimal Resource Usage

EventReporter has no noticeable impact on system resources. It was specifically written with minimal resource usage in mind. In typical scenarios, its footprint is barely traceable. This ensures it can also be installed on heavily loaded servers.

Full Windows Event Log Decoding

EventReporter can fully decode all types of Windows Event Log entries. It has the same capabilities like event viewer.

Windows Service

The EventReporter Service is implemented as a native multithreaded Windows service. It can be controlled via the control panel services applet or the computer management MMC.

Double Byte Character Set Support (e. g. Japanese)

EventReporter supports characters encoded in double byte character sets (DBCS). This is mostly used with Asian languages like Japanese or Chinese. All DBCS strings are forwarded correctly to the syslog daemon or email recipient. However, the receiving side must also be able to process DBCS correctly. Adiscon's syslog daemon for Windows, WinSyslog, does so. The output character encoding is selectable and supports Shift-JIS, JIS, and EUC-JP for Japanese users.

Multi-Language Client

The EventReporter client comes with multiple languages ready to go. Out of the box English, German, and Japanese are supported. Languages can be switched instantly. Language settings are specific to a user.

Additional languages can be easily integrated using Adiscon's brand new XML based localization technology. We ask customers interested in an additional language for a little help with the translation work (roughly 1 hour of work). Adiscon will then happily create a new version. This service is free!

Friendly User Interface

New Cloning feature has been also added to the EventReporter Client. In short you can now clone a Ruleset, a Rule, an Action, or a Service with one mouse click. Move up and Move down function has been added for Actions in the EventReporter Client. The EventReporter Client Wizards has been enhanced for creating Actions, Services, and RuleSets. And other minute changes!

Multiple RuleSets - Rules - Actions

With EventReporter as many "RuleSets", "Rules" and "Actions" as necessary can be defined.

multiple rulesets - rules - actions

Handling for low-memory cases

MWAgent allocates some emergency memory on startup. If the system memory limit is reached, it releases the emergency memory and locks the queue. That means not more items can be queued, this prevents a crash of the Agent and the queue is still being processed. Many other positions in the code have been hardened against out-of-memory scenarios.

Components

EventReporter Client

The EventReporter Client is used to configure all components and features of EventReporter. The client can also be used to create a configuration profile on a base system. That profile can later be distributed to a large number of target systems.

EventReporter Service

The EventReporter Service - called "the service" - runs as an Windows Service and coordinates all log processing and forwarding activity at the monitored system (server or workstation).

The service is the only component that needs to be installed on a monitored system. The EventReporter service is called the product "engine". As such, we call systems with only the service installed the engine-only installations.

The EventReporter service runs in the background without any user intervention. It can be controlled via the control panel "services" applet or the "Computer Management" MMC. The service operates as follows:

After starting, it periodically reads the Windows Event Log. Each message is formatted and then sent to the given Syslog daemon or email recipient. After all entries have been read, EventReporter goes to sleep and waits a given amount of time without any processing. This so-called "sleep period" is user configurable. As soon as the service returns from the sleep period, it once again iterates through the Windows event logs. This processing continues until the process is stopped.

Due to its optimized structure, EventReporter uses only very minimal processing power. How much it uses mainly depends on how long the sleep period is. We recommend a sleep period between 1 and 5 minutes for Syslog delivery and some hours up to 1 day for email delivery. However, feel free to customize this value according to your needs. We strongly recommend not using sleep periods of 500 milliseconds or less (although possible).

x64 Build

The installer inherits the 32bit as well as the 64bit edition. It determines directly, which version is suitable for your operating system and therefore installs the appropriate version. Major compatibility changes for the x64 platform have been made in the Service core. For details see the changes listed below:

- ODBC Database Action fully runs on x64 now. Please note that there are currently very few ODBC drivers for x64 available!
- Configuration Registry Access, a DWORD Value will now be saved as QWORD into the registry. However the Configuration Client and Win32 Service Build can handle these data type and convert these values automatically into DWORD if needed. The Configuration Client will remain a win32 application. Only the Service has been ported to the x64 platform.

A note on cross updates from Win32 to x64 Edition of EventReporter!

It is not possible to update directly from Win32 to x64 Edition using setup upgrade method. The problem is that a minor upgrade will NOT install all the needed x64 components. Only a full install will be able to do this. Therefore, in order to perform a cross update, follow these instructions:

1. Create a backup of your configuration, save it as registry or xml file (See the Configuration Client Computer Menu)
2. Uninstall EventReporter.
3. Install EventReporter by using the x64 Edition of the setup.
4. Import your old settings from the registry or xml file.

System Requirements

EventReporter requires very limited resources of the machine to run optimally. The actual minimum requirements to run the application depend on the type of installation. If only the client is installed, they are a bit higher otherwise the service needs a few enabling it to run on a large variety of machines – even the highly utilized ones.

Client

- The client can be installed on Windows 10, Windows 11, and Windows Server 2016/2019/2022. The operating system variant (Workstation, Server …) is irrelevant. Note: For legacy systems (Windows XP, Server 2003), older versions are available - contact Adiscon for details.
- The client is suited for 32bit and 64bit operating systems. It runs automatically on each platform in 32Bit or 64Bit mode.
- The client uses Microsoft .Net Framework technology. The Installer will automatically install the necessary .Net Framework components before installation. A network connection maybe required in order to download additional components.
- The client requires roughly 8 MB RAM in addition to the operating system minimum requirements. It also needs around 5 MB of disk space.

Service

- The service has fewer requirements.
- It works under the same operating system versions.
- At runtime, the base service requires 5 MB of main memory and less than 5 MB of disk space. However, the actual resources used by the agent largely depend on the services configured.
- Please note that the 32Bit Service is limited to 2GB of usable memory. The 64Bit version does not have any limit. A typical Syslog message (including overhead) takes roughly 4-8 KB. With 1024 MB, you can buffer up to 100,000-200,000 messages in 1024 MB.
- On Windows Server editions, additional event logs (such as “DNS Server”, “File Replication Service”, and “Directory Service”) are automatically supported.

Product Tour

EventReporter - Quick Tour

Monitoring

Event Log Monitor V1

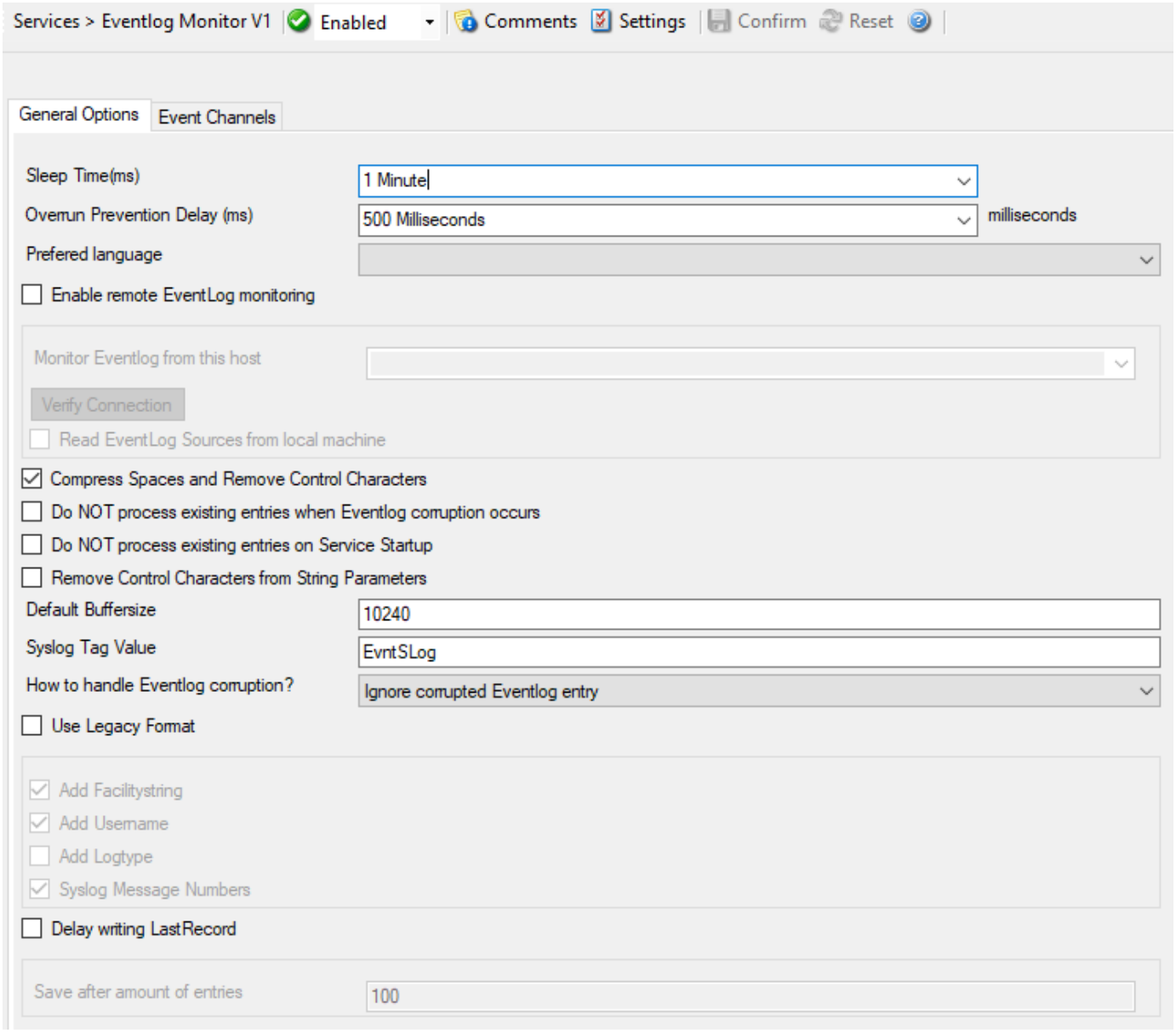
Complete Windows Event Log Monitoring

All currently-existing logs are fully supported: the standard Windows Event Logs, additional logs introduced by modern Windows as well as custom event logs and the Window Vista event logging system. Also supported are Windows Event Log files. That feature supports NAS-devices, which often offer log information in Windows Event Log file format (.evt). By monitoring these files, SAN devices, too, can be monitored in near-real-time.

The Windows Event Log Monitor service processes Windows Event Logs. These are the native Windows event repositories. All “well-behaved” applications log messages into them. With the Event Log Monitor, they can automatically be parsed, filtered, processed, and forwarded. The full set of log sources in supported, including modern Windows specific logs, custom event logs and event log files (e.g. from SAN devices).

General Options

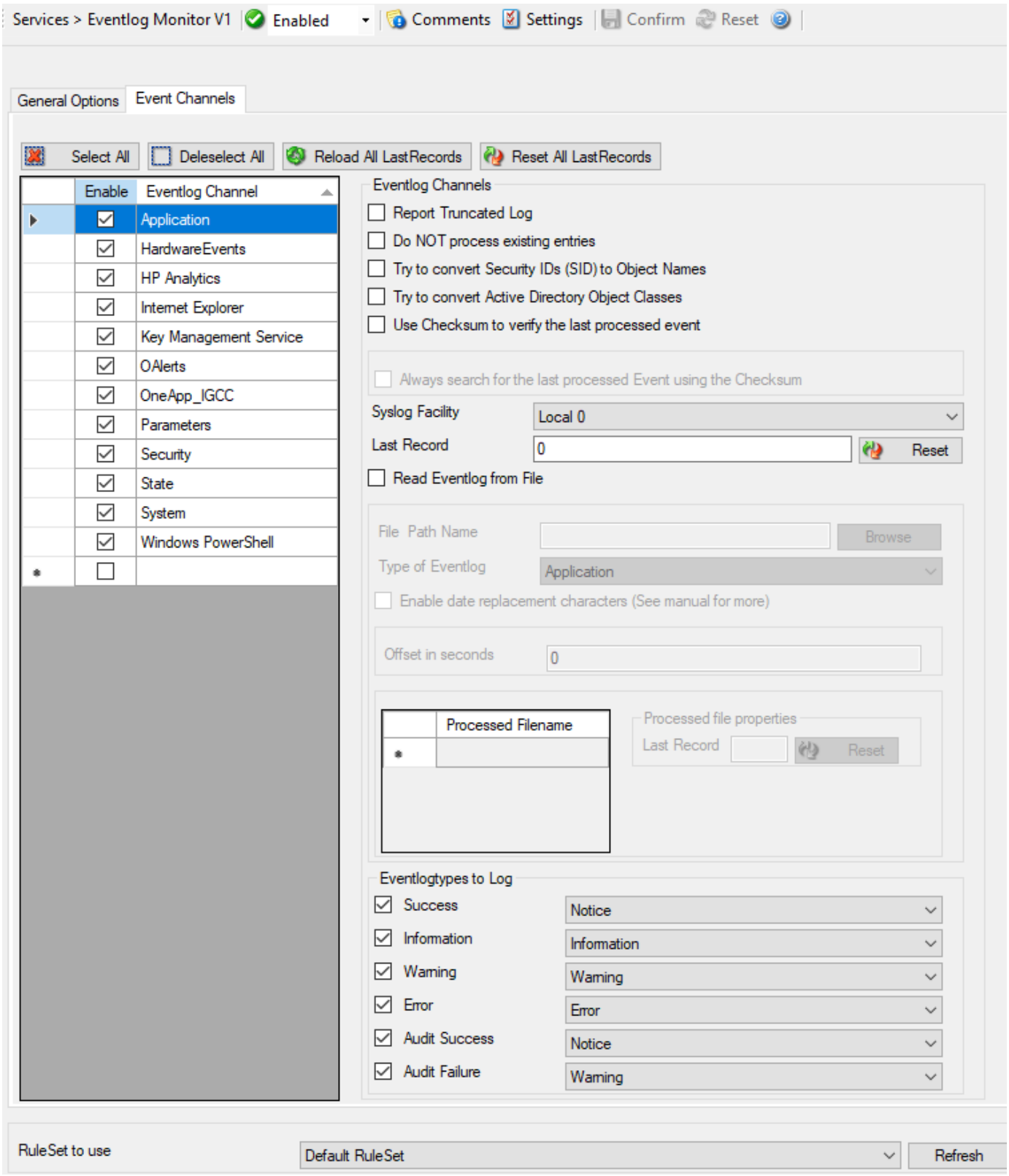
General Options available on this form are “Sleep Time” and “Overrun Prevention Delay”. Sleep time specifies the time after which event log monitor should check for new events whereas overrun prevention delay allows configuring a delay after generating an event.



- Event Log Monitor V1 - General Options*

Event Log Channels Tab

The “Event Log Channels” configures per-event-log settings. The corresponding log will only be processed if the respective “Enable” checkbox is checked. The parameters are common to all logs and each dialog looks similar:



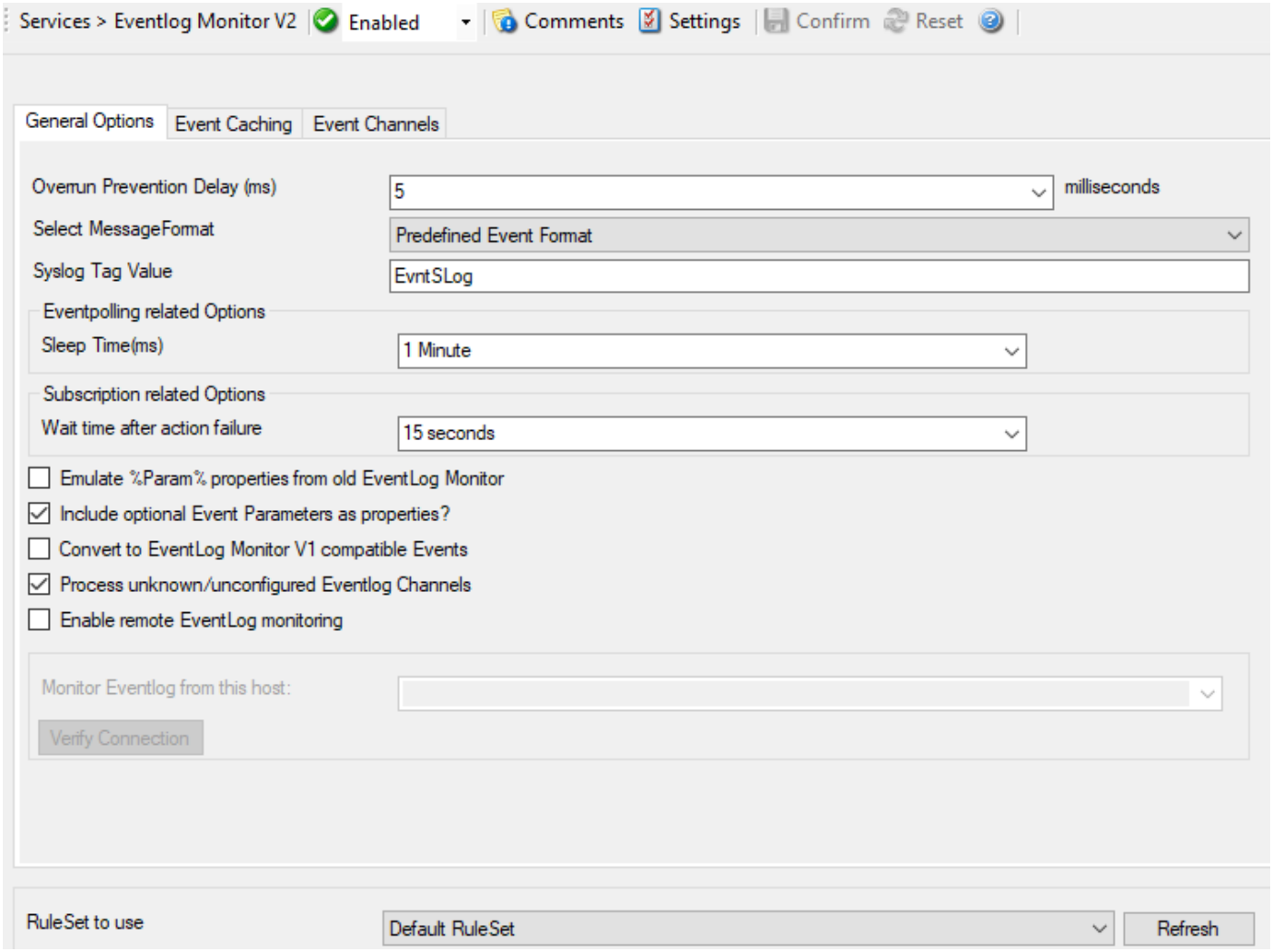
- Event Log Monitor V1 Event Channels*

Further details can be found here event log monitor v1.

Event Log Monitor V2

The Windows Event Log Monitor V2 service processes Windows Event Logs. All currently-existing logs are fully supported: the EventLog Monitor V2 is especially designed for the use with Windows Vista, Windows 7, and higher. This makes you capable of using the all-new and advanced Event Log structure of the most recent version of Windows.

This is the key feature of MonitorWare Agent, when used with Windows Vista, 2008, or higher. It is necessary to use this service, because of the vast changes that came along with this new operating system. The old EventLog Monitor wasn't capable of reading all the new Logs. Therefore it was necessary to create a separate tool for Vista, 2008, or higher. With all Logs being stored in one place, this should give you more power over them, than before.



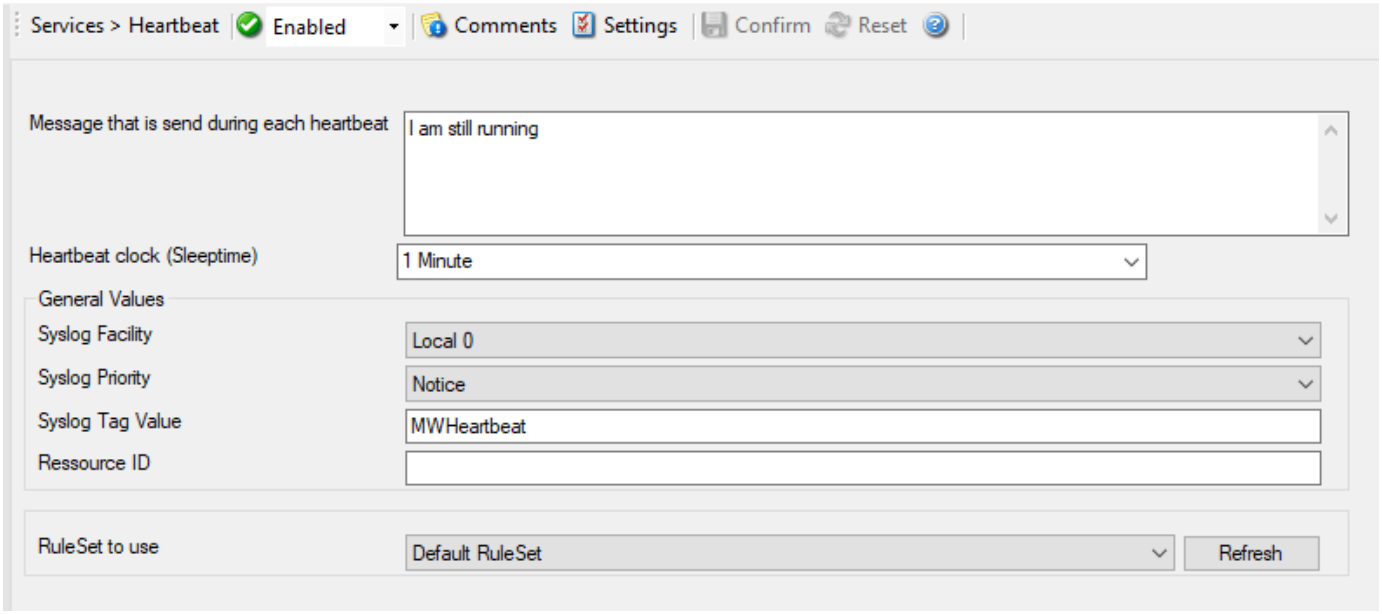
- Event Log Monitor V2*

Further details can be found here: event log monitor v2.

filtering

Heartbeat

The Heartbeat Process can be used to continuously check if everything is running well. It generates an information unit every specified time interval. That information unit can be forward to a different system. If it does not receive additional packets within the configured interval, it can doubt that the sender is either in trouble or already stopped running.

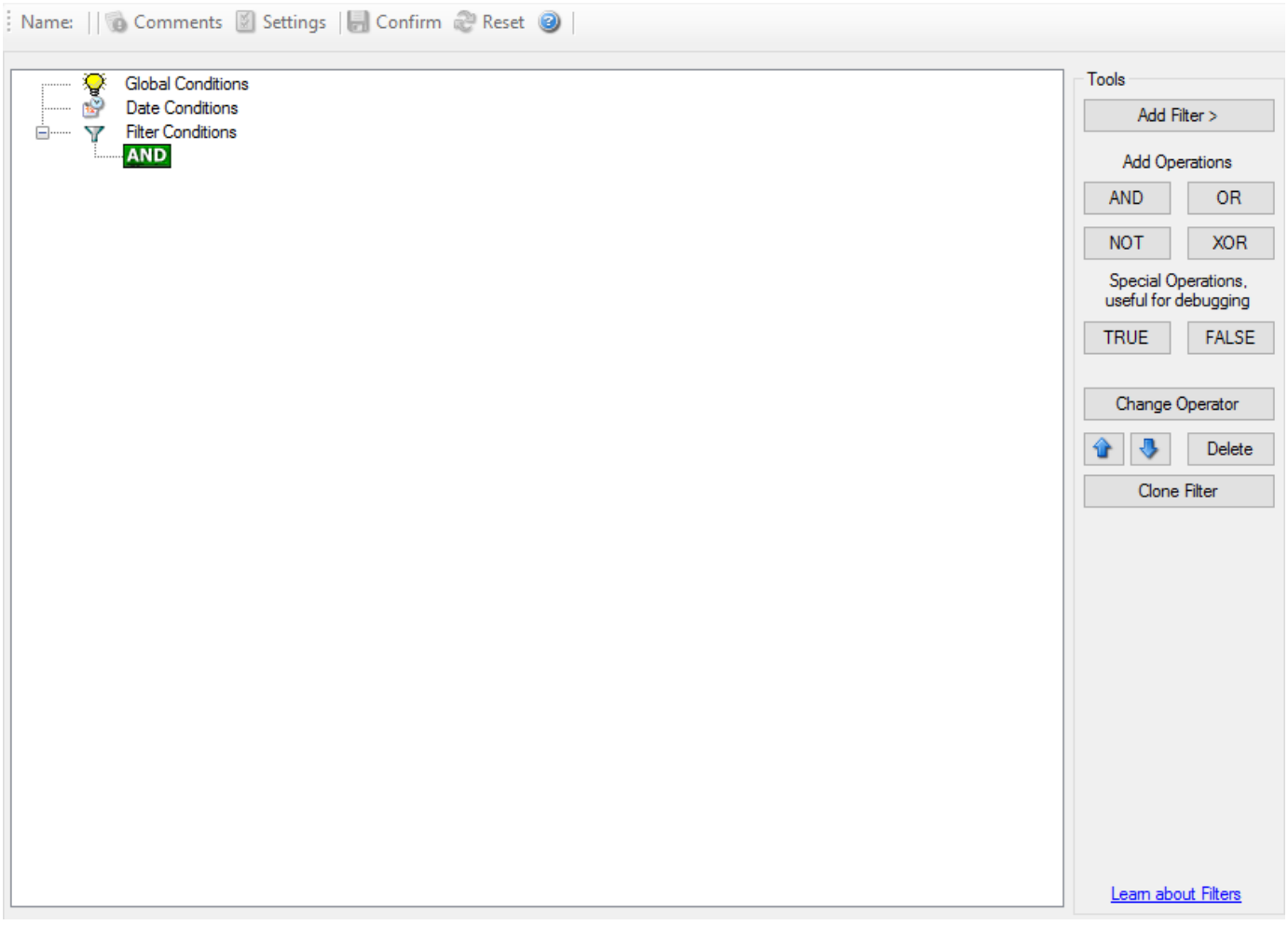


- Heartbeat*

Further details can be found here: heartbeat.

Local Filtering

EventReporter can locally filter events based on the any event log property (like event id, severity or even partial message content). Flexible boolean operations and nested conditions are supported. That way, you can discard messages you are not interested in and alert high-priority message e.g. via email.

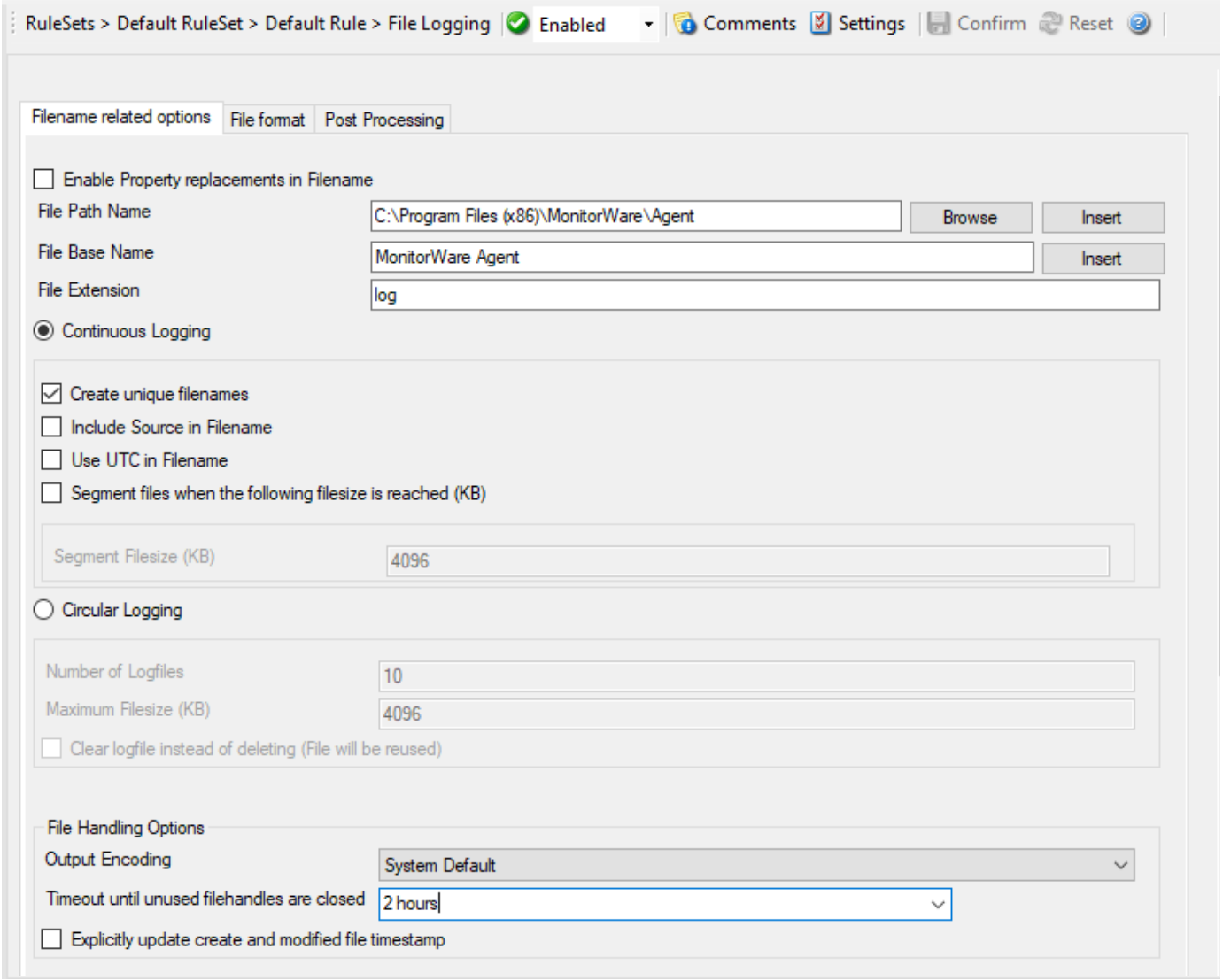


Further details can be found here: filter conditions.

log storage

Write to File

All incoming events – no matter what source they came from – can be stored persistently. Options include archiving in databases as well as log files. File logging is used to write text files of received messages. One file per day is written. New entries are appended to the end of the file.

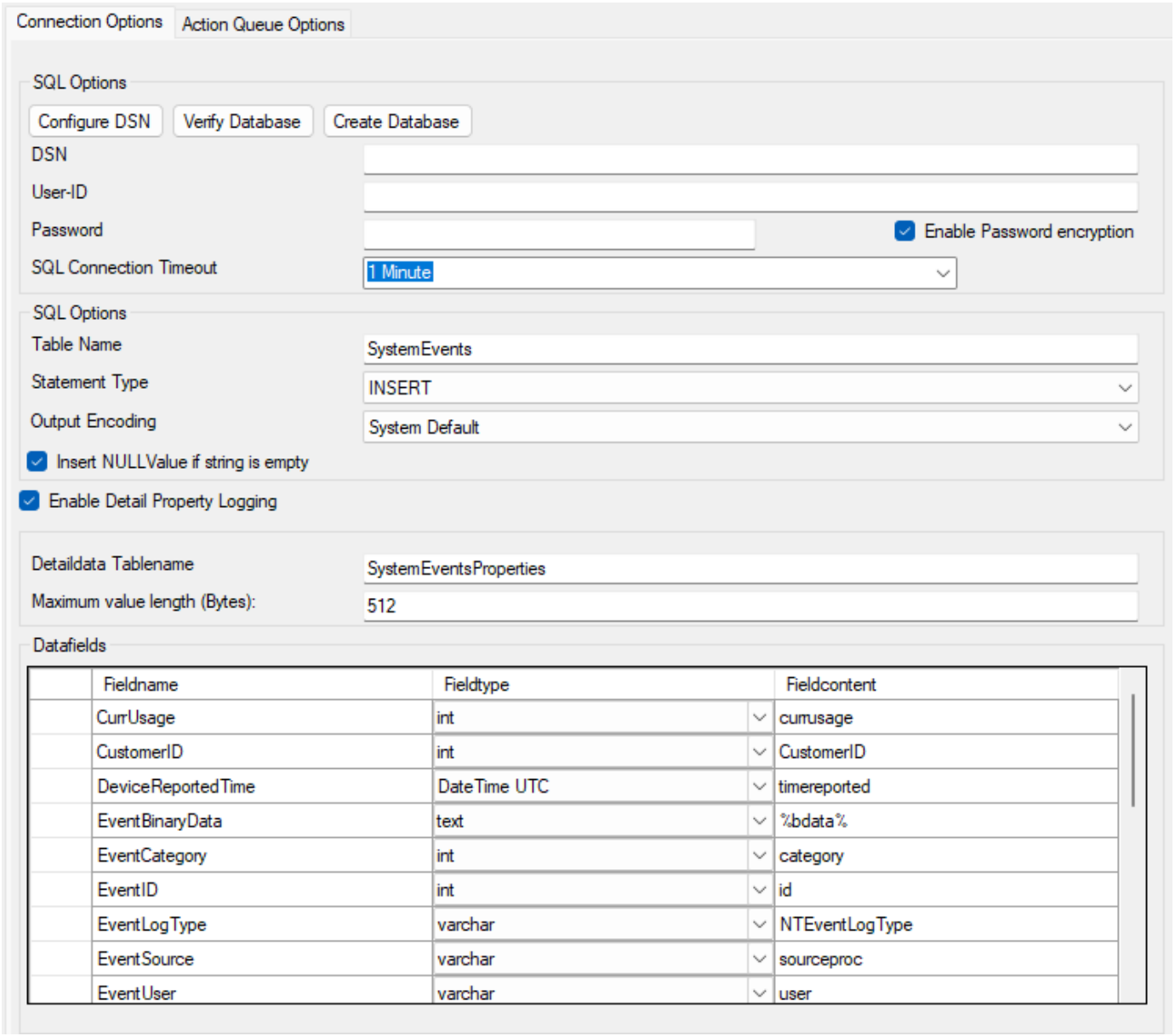


- Write to File*

Further details can be found here: write to file.

Write to Database

Database logging allows persisting all incoming messages to a database. Once they are stored inside the database, different message viewers as well as custom applications can easily browse them.

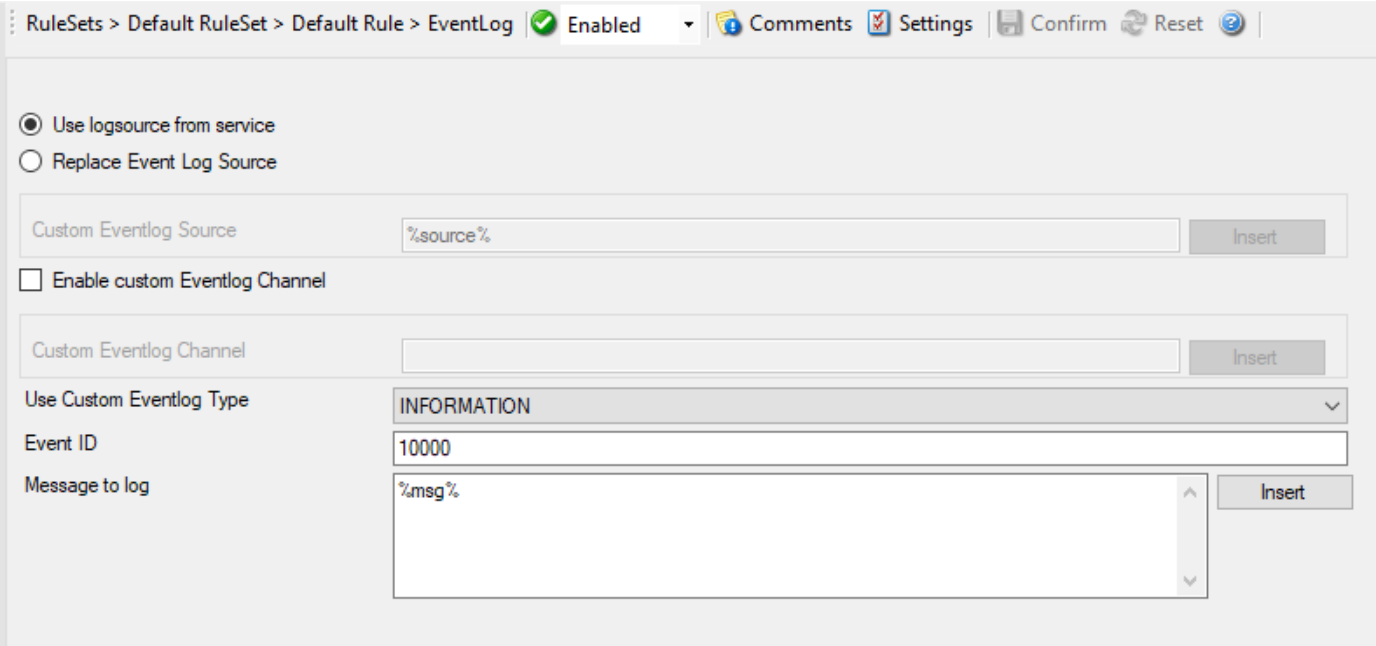


- Write to Database*

Further details can be found here: write to database.

Write to Event Log

Allows any event (e.g. syslog, SNMP trap, protocol probes) to be written to the Windows Event Log. It is used to configure the logging to the Windows or XP event log. It is primarily included for legacy purposes.



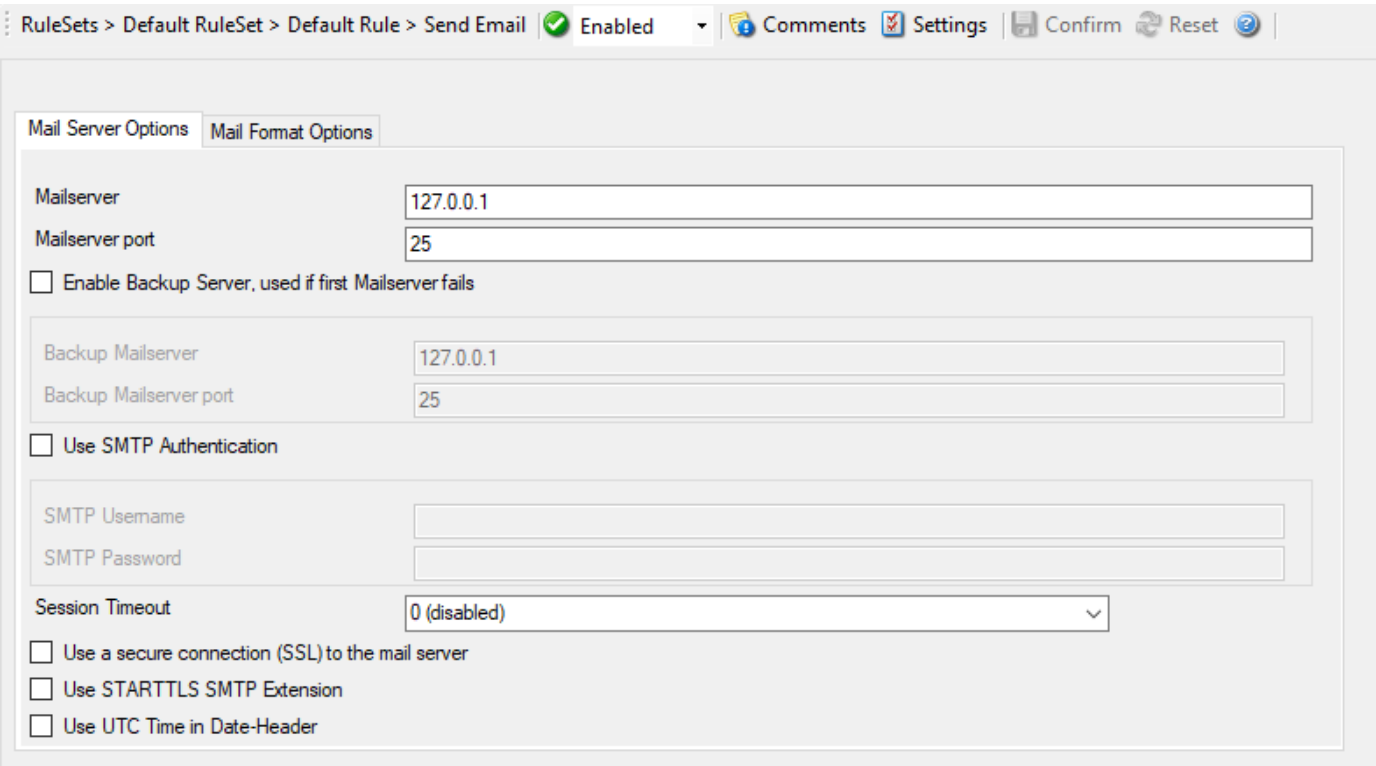
- Write to Event Log*

Further details can be found here: event log.

alerting

Forward via eMail

Events of any kind can be forwarded via eMail. This is most often used for alerting. Together with your cell phone's provider eMail to messaging functionality, you can often send events directly to your cell phone. You can use this feature to receive eMail messages in your mail boxes.



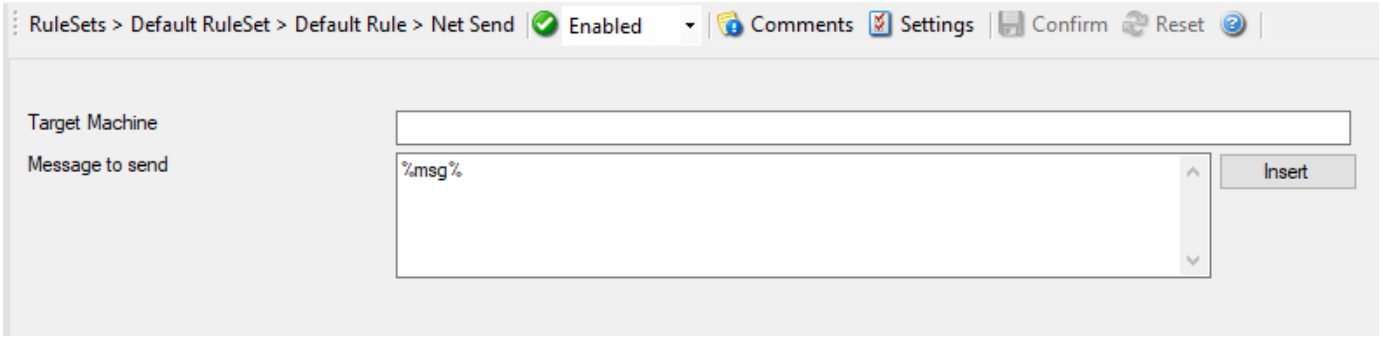
- Forward via eMail*

Here is an example how to receive email notifications when certain events happen.

Further details can be found here: action send email.

Net Send

This helps to send short alert messages to recipient machine via Windows Net Send facility. Great for alerting logged-on administrators.



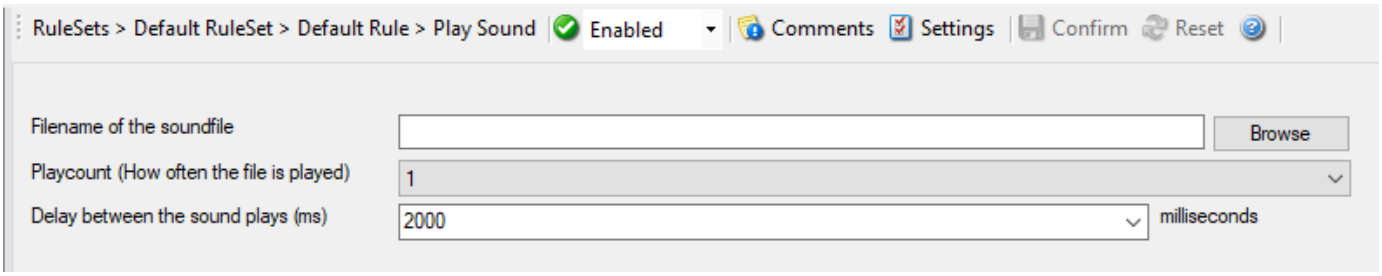
- Net Send*

Here is an example how to receive notifications via net send.

Further details can be found here: net send.

Play Sound

This action allows you to play a sound file.



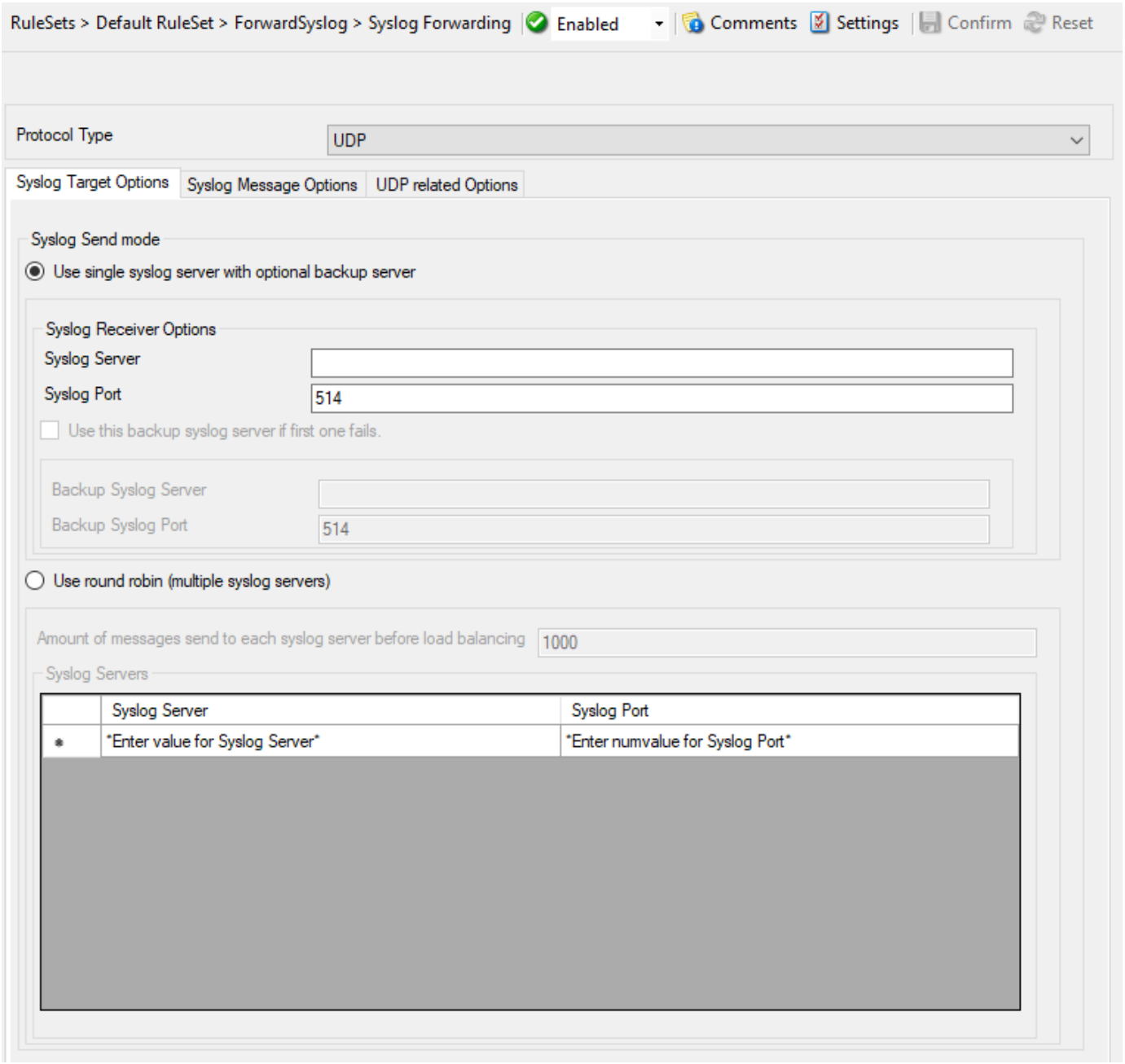
- Play Sound*

Further details can be found here: play sound.

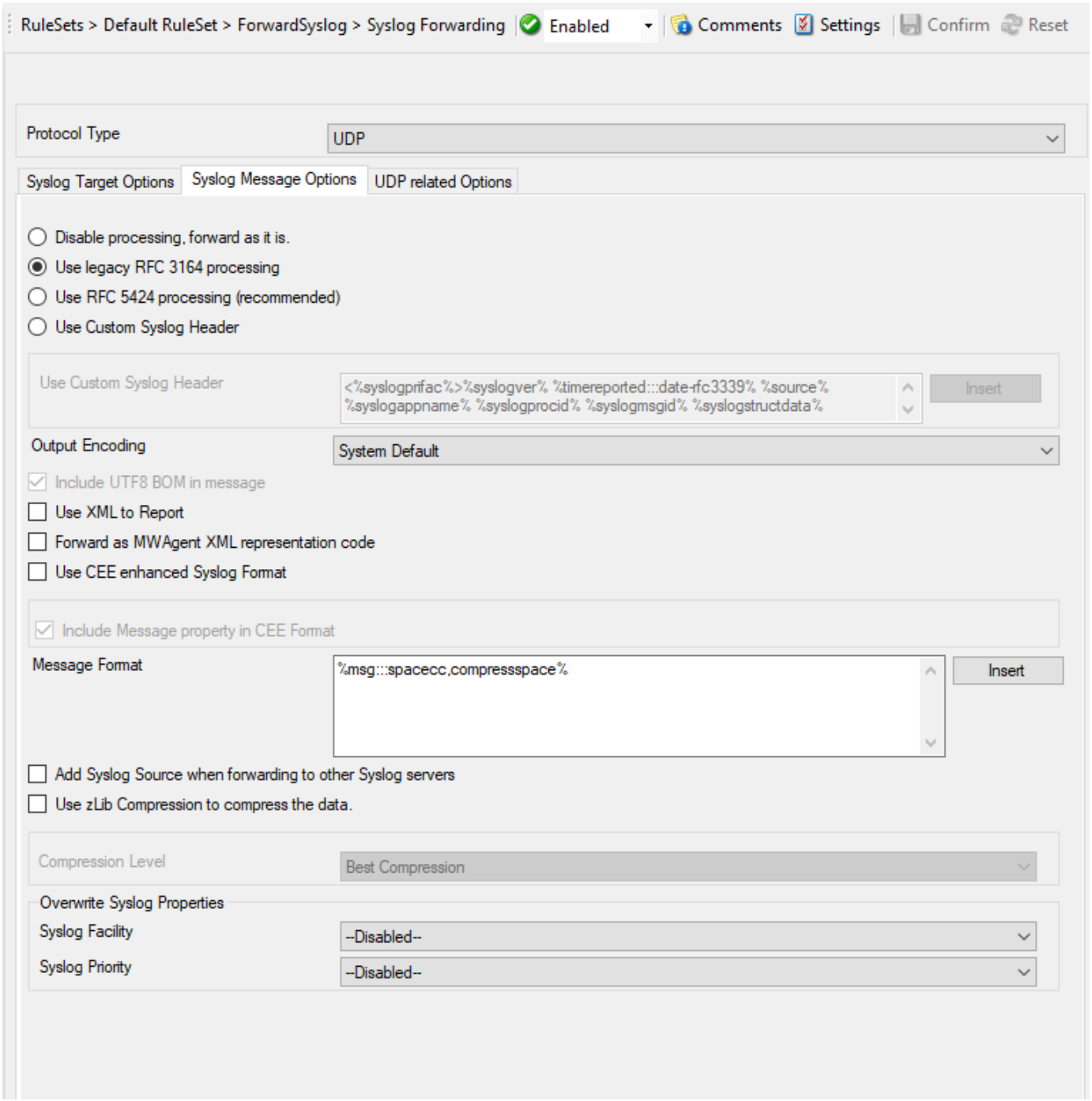
miscellaneous

Syslog Forwarding

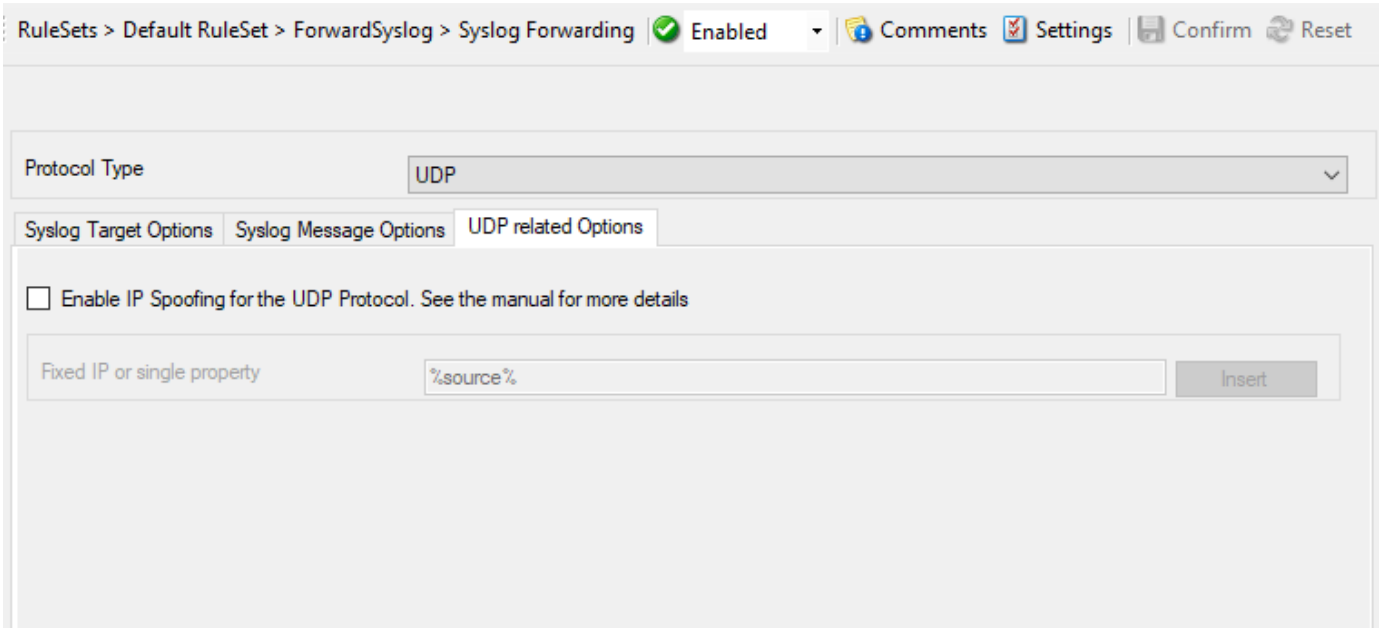
EventReporter can locally filter events based on the any event log property (like event id, severity or even partial message content). Flexible boolean operations and nested conditions are supported. That way, you can discard messages you are not interested in and alert high-priority message e.g. via email.



- Syslog Target Options*



- Syslog Message Options*



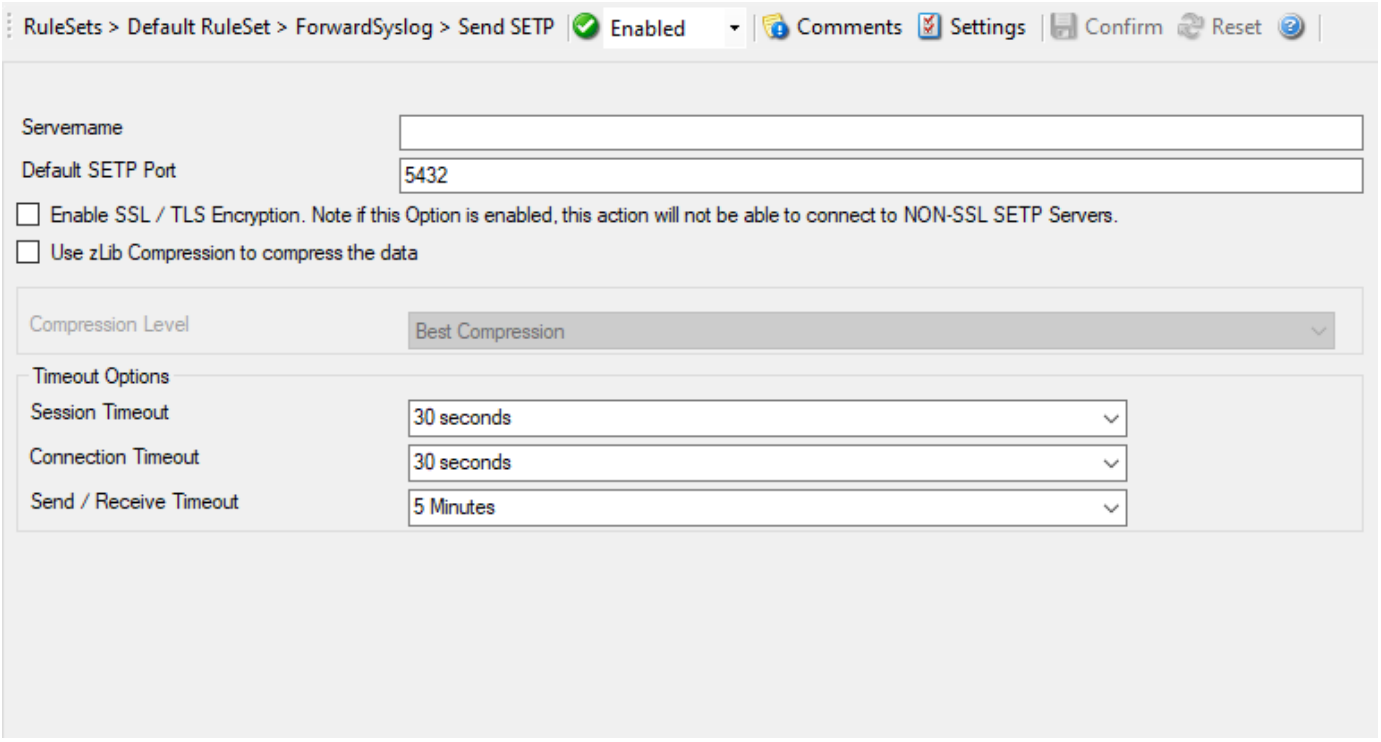
- UDP related Options*

Further details can be found here: syslog forwarding.

Forward via SETP

SETP is Adiscon's protocol for reliably transferring event messages in native format. It is primarily of use for those that want to build a larger, centralized event repository. SETP can optionally be SSL-encrypted.

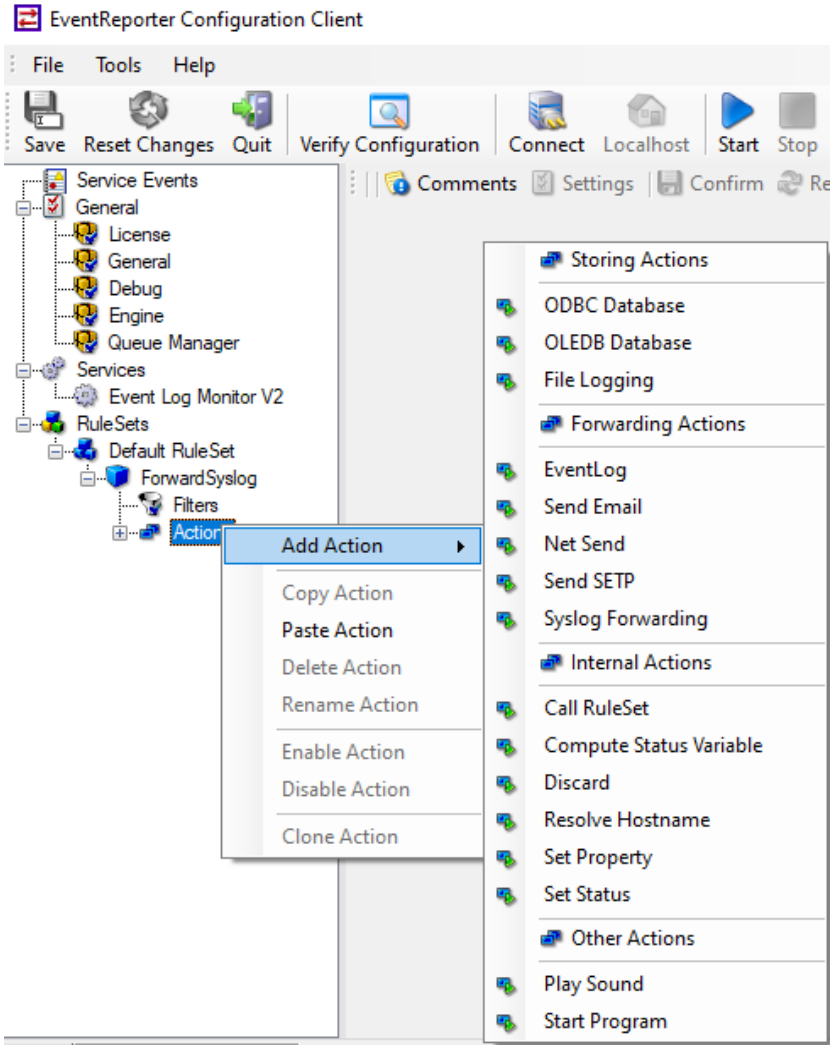
SETP was originally developed for MonitorWare but now it is a key feature added first in EventReporter 6.2. NT Event Messages can be forwarded using SETP protocol. Windows Event Log are monitored successfully as well.



Further details can be found here: send setp.

Powerful Event Processing

EventReporter has a powerful and flexible rule engine that processes all events based on a configured set of actions. An unlimited number of rules and actions allows tailoring to the specific needs.

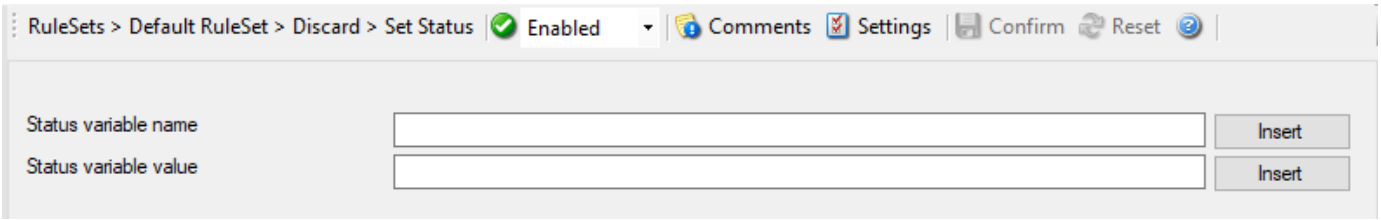


- Powerful Event Processing*

Further details can be found here: actions.

Set Status

Each information unit has certain properties e.g. EventID, Priority, Facility etc. You can create a new property and assign any valid desired value as well as filter to it. This is great for very demanding situations where complex rule sets are needed.

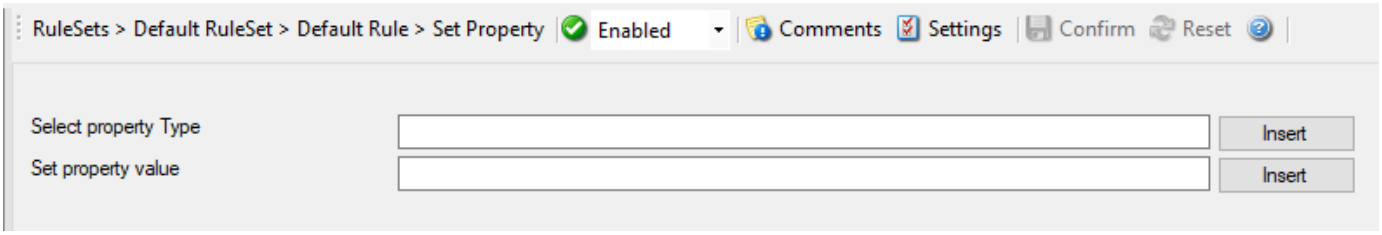


- Set Status*

Further details can be found here: set status.

Set Property

With the “Set Property”, some properties of the incoming message can be modified. This is especially useful if an administrator would like to e.g. rename two equally named servers.



Further details can be found here: set property.

Start Program

With this, an external program can be run. Any valid Windows executable can be run. This includes actual programs (EXE files) as well as scripts like batch files (.BAT) or VB scripts (.vbs). Start Program can, for example, be combined with the service monitor to restart failed services.

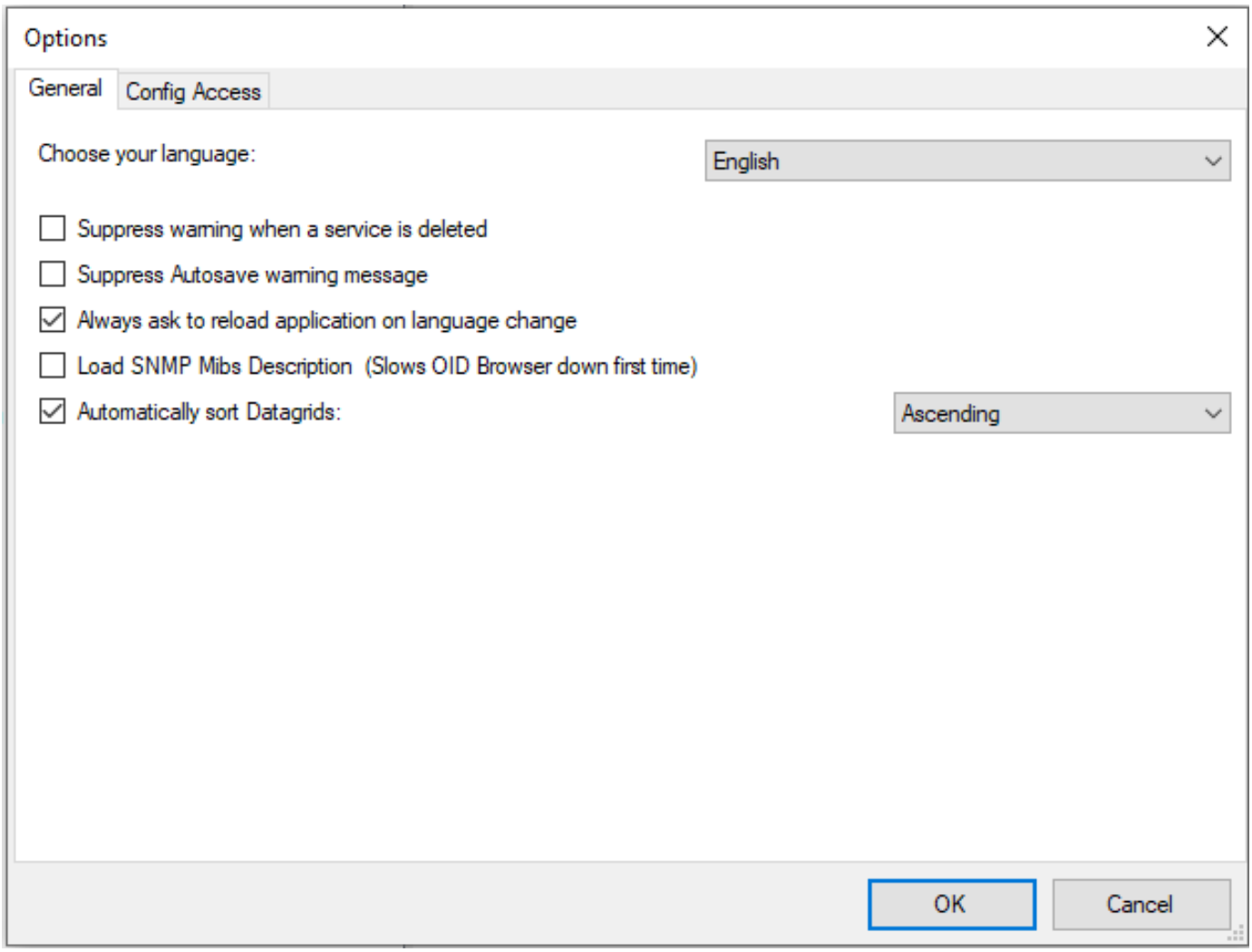
- Start Program*

Further details can be found here: start program.

Multi-Language Client

The Client comes with multiple languages ready to go. Out of the box English, German and Japanese are supported. Languages can be switched instantly. Language settings are specific to a user.

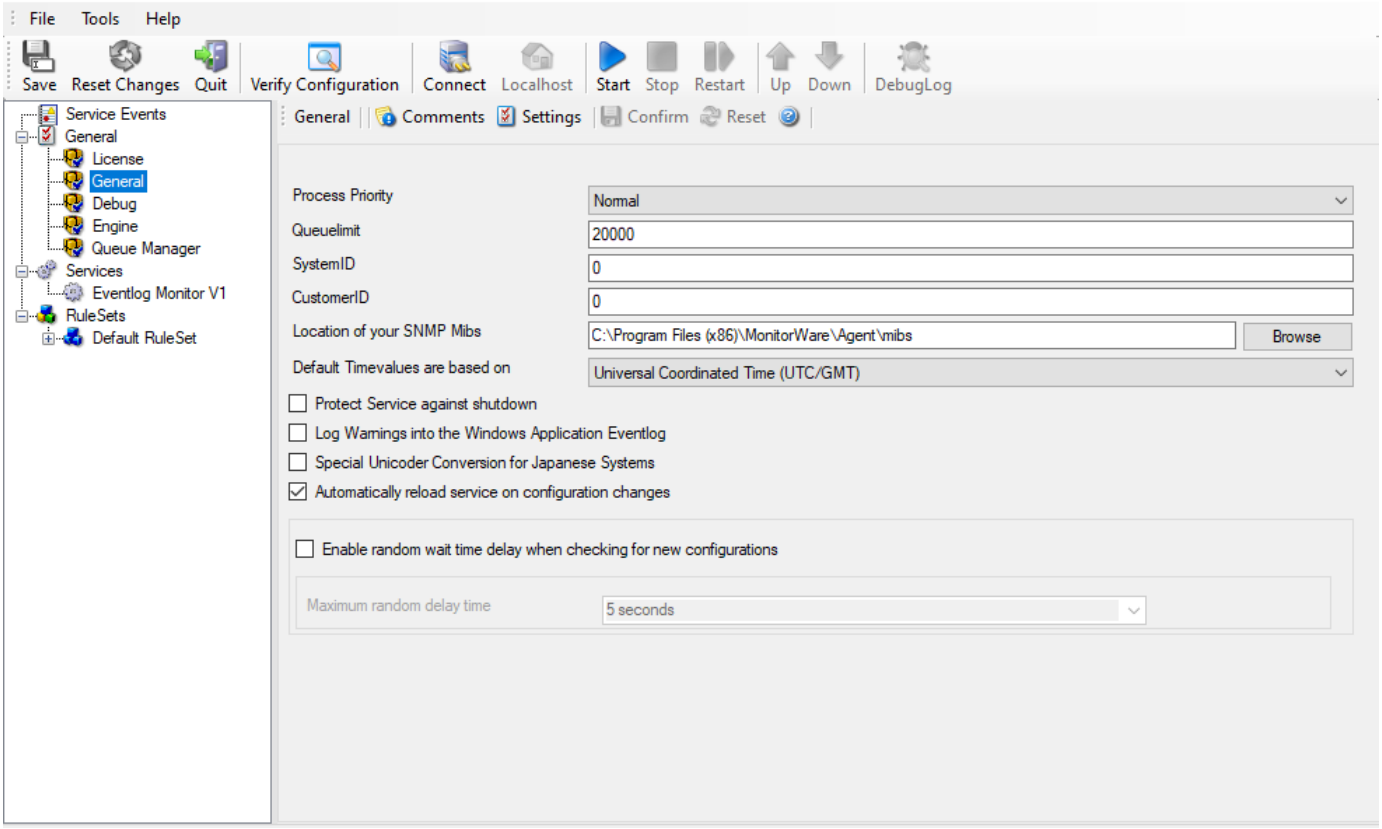
Additional languages can be easily integrated using Adiscon's XML based localization technology. We ask customers interested in an additional language for a little help with the translation work (roughly 1 hour of work). Adiscon will then happily create a new version. This service is free!



Friendly and Customizable User Interface

The Cloning feature helps to clone a Ruleset, a Rule, an Action, or a Service with one mouse click. It includes a Move up and Move down function for Actions in the Client.

With the MonitorWare Agent as many RuleSets, Rules and Actions as necessary can be defined.



- Friendly and Customizable User Interface*

Other Miscellaneous Features

Ease of use

Using the new EventReporter client interface, the product is very easy to setup and customize. We also support full documentation and support for large-scale unattended installations.

Comprehensive Windows Event Log Support

EventReporter provides complete support for all Windows Event Log formats and systems, including modern Windows event logging infrastructure. All event log information can be gathered, fully decoded, and submitted to log targets. This includes support for custom event logs and advanced event log features. EventReporter handles whatever Windows Event Log requirements you may have.

Runs on large variety of Windows Systems

This includes Windows 10, Windows 11, and Windows Server 2016/2019/2022. Workstation or Server, 32 or 64 bits – EventReporter runs on all of them. Note: For legacy systems (Windows XP, NT, Server 2003), older versions are available - contact Adiscon for details. available on request). Please note that the current builds do no longer support older platforms, but we have previous releases available for those that need them (the current software can no longer run on e.g. NT 3.5 platforms, because there are technical issues mostly with the installer).

Robustness

EventReporter is running in a large number of installations. It is written to perform robustly even under unusual circumstances. Its reliability has been proven at customers' sites since 1997.

Remote Administration

The client can be used to remotely manage EventReporter instances.

Minimal Resource Usage

EventReporter has no noticeable impact on system resources. It was specifically written with minimal resource usage in mind. In typical scenarios, it's footprint is barely traceable. This ensures it can also be installed on heavily loaded servers.

Full Windows Event Log Decoding

EventReporter can fully decode all types of Windows Event Log entries. It has the same capabilities like event viewer.

Windows Service

The log forwarding process "the engine" is implemented as a native multithreaded Windows service. It can be controlled via the control panel services applet, the computer management console and any other means for controlling services.

Double Byte Character Set Support (e. g. Japanese)

EventReporter supports characters encoded in double byte character sets (DBCS). This is mostly used with Asian languages like Japanese or Chinese. All DBCS strings are forwarded correctly to the syslog daemon or email recipient. However, the receiving side must also be able to process DBCS correctly. Adiscon's syslog daemon for Windows, WinSyslog, does so. The output character encoding is selectable and support Shift-JIS, JIS and EUC-JP for Japanese users.

Getting Started

EventReporter can be used for simple as well as complex scenarios.

This chapter provides a quick overview of EventReporter and what can be done with it.

Most importantly, it contains a tutorial touching many of the basic tasks that can be done with MonitorWare Agent as well as pointer on how to setup and configure.

Be sure to at least briefly read this section and then decide where to go from here - it is definitely a worth time spent.

Installation

Installing the product is straightforward because a familiar Windows setup program guides you through the process.

Multiple download variants are available so you can pick the package that matches your environment; always install the most recent release to benefit from the latest fixes and improvements.

Each installer automatically creates a Windows Firewall exception for the service process during setup, ensuring the service can communicate right away without additional manual steps.

Review the dedicated Installing EventReporter guide for a step-by-step walkthrough. The latest setup packages are available on the Download Versions page, and they ship as direct install sets so you can launch them immediately without extracting archives.

Information for a Mass Rollout

A mass rollout in this context means deploying an Adiscon client (for example MonitorWare Agent, EventReporter, or WinSyslog) to more than a handful of systems in an automated way. The aim is to invest the upfront effort needed to create a consistent "master" configuration once and then reuse it for every target machine. For guidance on differentiating between initial and update rollouts, see the MWAgent FAQ section.

Preparing the Baseline

1. Install the product on a single master system and configure it exactly as desired. Verify that the configuration works before continuing.
2. Export the configuration to a registry file via the configuration client (`Computer` → `Export Settings`).

3. Gather the files required for an engine-only installation:
    - For MonitorWare Agent 8.1 and newer this is typically `mwagent.exe` and `mwagent.pem`.
    - Older releases may also require the Visual C++ runtime and OpenSSL helper files (`Microsoft.VC90.CRT.manifest`, `libeay32.dll`, `ssleay32.dll`, `msvcm90.dll`, `msvcp90.dll`, `msvcr90.dll`).

Automated Rollout Example

Once the master system is prepared, copy the required files to a network share or removable media and automate the rollout with a script similar to the following:

```
copy \\server\share\mwagent.exe C:\some-local-dir
copy \\server\share\mwagent.pem C:\some-local-dir
cd C:\some-local-dir
mwagent -i
regedit /s \\server\share\configParams.reg
net start "AdisconMonitorWareAgent"
```

`configParams.reg` represents the registry export taken from the master system. Because the rollout ships only the engine files, this approach works well for DMZ environments where RPC or file sharing cannot be opened.

> **Note**
>
> `mwagent -i` (or the equivalent command-line switch for other Adiscon products) only registers the Windows service. It assumes the binaries already exist in the current directory, so copy the files before running the command.

Branch Office Rollouts

For branch offices or semi-automated deployments, distribute the prepared package and have the local administrator perform the following steps:

1. Create a directory on the target computer and copy the provided files into it.
2. Run `mwagent -i` from that directory to register the service.
3. Import the exported configuration by double-clicking the `.reg` file (or by running `regedit /s` from an elevated command prompt).
4. Start the Windows service via `net start` or the Services management console. Restarting the entire machine is not required.

> **Important**
>
> The directory that hosts the engine files **is** the installation directory. Deleting it removes the binaries and effectively uninstalls the product.

Updating Existing Rollouts

To upgrade an engine-only installation, update the master system first, export the revised configuration, and distribute the refreshed files using the same process. Uninstallation is unnecessary as long as you overwrite the files in place, but always stop the Windows service before copying the new binaries. For a walkthrough focused on update scenarios, refer to the MWAgent FAQ section.

Obtaining a Printable Manual

A printable version of the manual can be obtained at https://www.EventReporter.com/manuals.

The manuals offered on this web page are in printable (PDF format) or HTML Versions for easy browsing and printing. The manual is also included as a standard Windows help file with all installations. So if you have the product already installed, there is no need to download these documents.

The version on the web might also include some new additions, as we post manual changes frequently – including new samples and as soon as they become available. Past manual versions are also available for those customers in need of it.

EventReporter Tutorial

This tutorial provides a rough overview of EventReporter as well as some of its typical uses. It is in no way complete, but helps in understanding EventReporter and how it can be configured to suit your needs.

In the tutorial, we start by describing and focusing on the filter conditions, as these are often needed to understand specified scenarios that follow below.

EventReporter gathers network events - or “information units” as we call them - with its services. Each of the events is then forwarded to a rule base, where the event is serially checked against the different rule’s filter conditions. If such condition evaluates to true (“matches”), actions associated with this rule are carried out (e.g. storing the information unit to disk or emailing an administrative alert).

Note: The screenshots in this tutorial are made with EventReporter and MonitorWare Agent. MonitorWare Agent is used, as the user interface is similar to the one EventReporter uses.

Debug Logging

When having Debug Logging enabled, the Username used for the Service will be printed below the Version Build number now. This is useful for debugging purposes.

Filter Conditions

For every rule, filter conditions can be defined in order to guarantee that corresponding actions are executed only at certain events.

These filter conditions are defined via logical operators. Boolean operators like “AND” or “OR” can be used to create complex filter conditions.

If you are note so sure about the Boolean operators, you might find the following brush-up helpful:

AND – all operands must be true for the result to be true. Example: AND (A, B):

Only if both A and B are true, the result of the AND operation is true. In all other cases, it is false.

OR – if at least one of the operands is true, the end result is also true.

Example: OR (A, B): The end result is only false if A and B are false. Otherwise, it is true.

XOR – it yields true if exactly one (but not both) of two operands is true.

Example: XOR (A, B): The end result is false if A and B both are True or False. Otherwise, it is true.

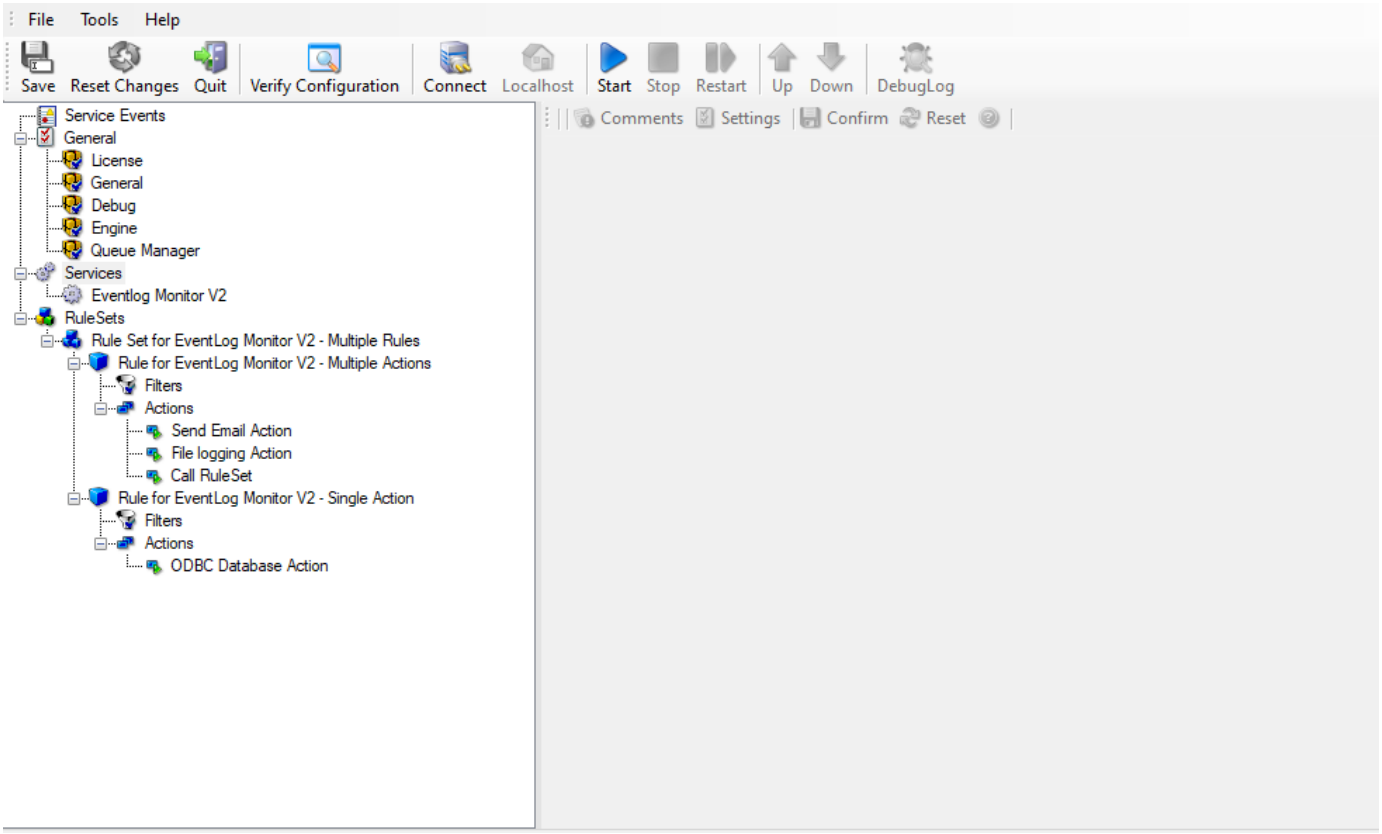
NOT – negates a value. Example: NOT A: If A is true, the outcome is false and

vice versa. There can only be a single operand for a NOT operation.

TRUE – returns true.

FALSE – returns false.

Multiple RuleSets - Rules - Actions



- Multiple RuleSets, Rules and Actions*

With EventReporter as many “RuleSets”, “Rules” and “Actions” as necessary can be defined.

You can create a separate “RuleSet” for each Service used, or just one “RuleSet” for all services. RuleSets can also be created to use them with the “callruleset action”.

RuleSets contain one or multiple rules. All actions and processing carried out is defined by the rules.

A rule consists of filter conditions and 1 to multiple actions. All actions that have to meet the same filter conditions can be combined in the same rule.

Ignoring Events

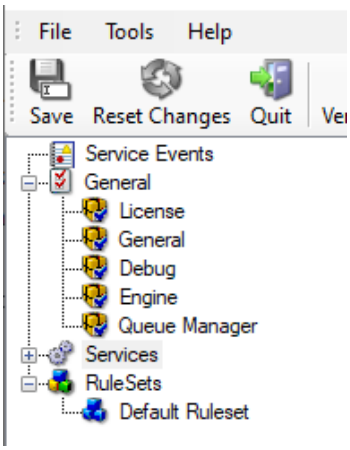
There are some events which occur often and you do not want them to be stored in your log files or either take any action on those.

We handle these events on top of our ruleset. This ensures that only minimal processing time is needed and they are discarded as soon as possible.

In this tutorial, we define a filter that discards such events. In our example, we assume that Events with the ID 105, 108, and 118 are not required. Please note that for simplicity reasons we only apply filter based on the event ID. In a production environment, you might want to add some additional properties to the filter set.

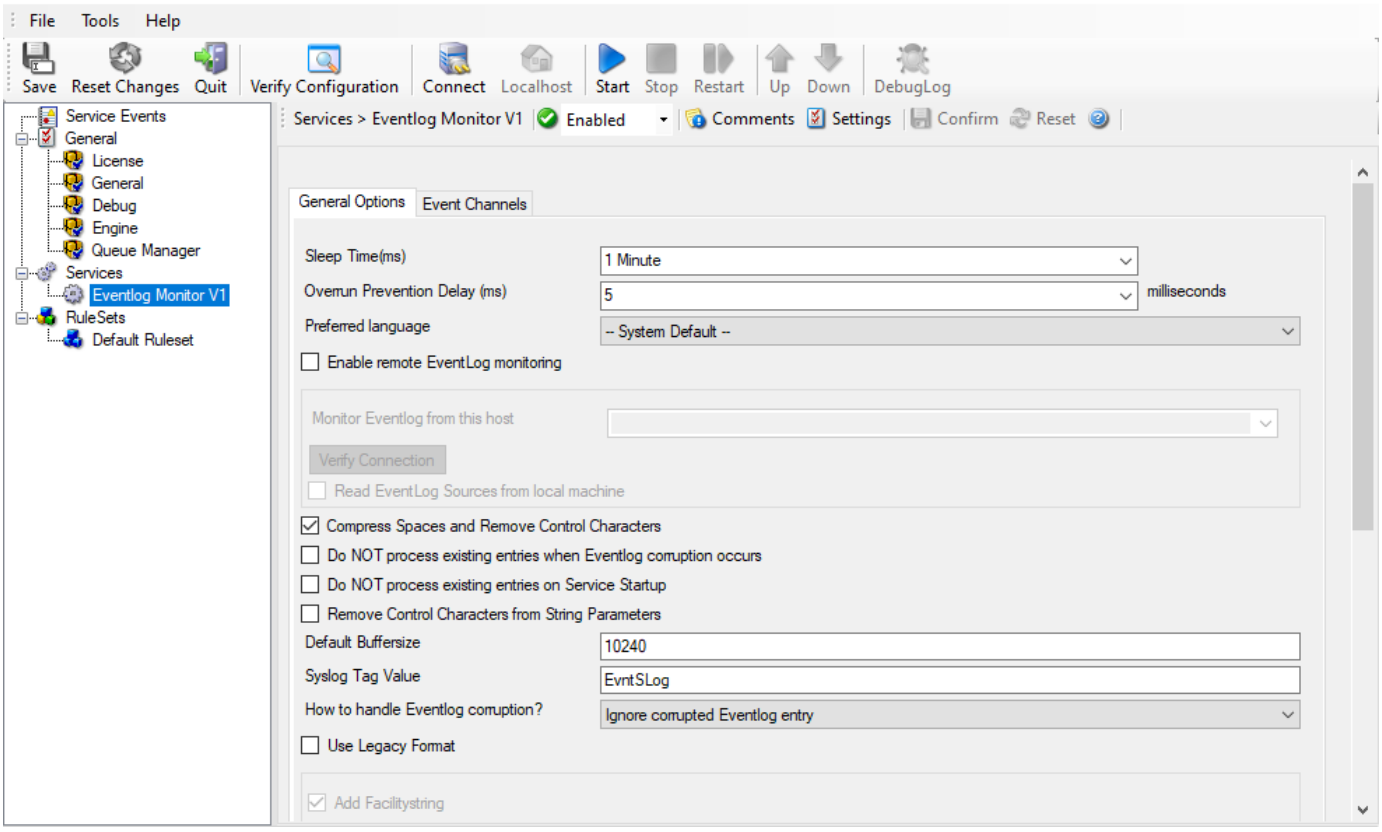
In this sample, no service or ruleset has yet been defined. It is just a “plain” system right after install.

We start by defining a ruleset. Right-click on “RuleSets” and choose “Add RuleSet” from the context menu. Enter a name of your choice. In this tutorial, we use the name “Default RuleSet”. Click on “Next”. Leave everything as it is in the next dialog. Click “Next”, then “Finish”. As can be seen in following screen shot, the ruleset “Default” has been created but is still empty.



- Ignoring Events - 1*

Of course we can only use a rule if we configure a corresponding service. To do so, right-click on “Running Services” and then select “Add Services”. Choose the desired “Service” from the context menu i.e. “Event Log Monitor” in this sample. Provide a name of your choice. In our sample, we call the service “Event Log Monitor”. Leave all defaults and click “Next”, then “Finish”. Now click on “Event Log Monitor” under “Running Services”. Your screen should look as follows:

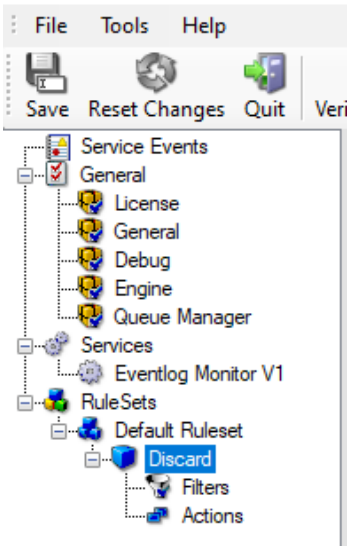


- Ignoring Events - 2*

As we had created the “Default” ruleset initially, it is shown as the rule set to use for this service. For our purposes, that is correct. To learn more on the power of ruleset assignments, see other sections of this manual.

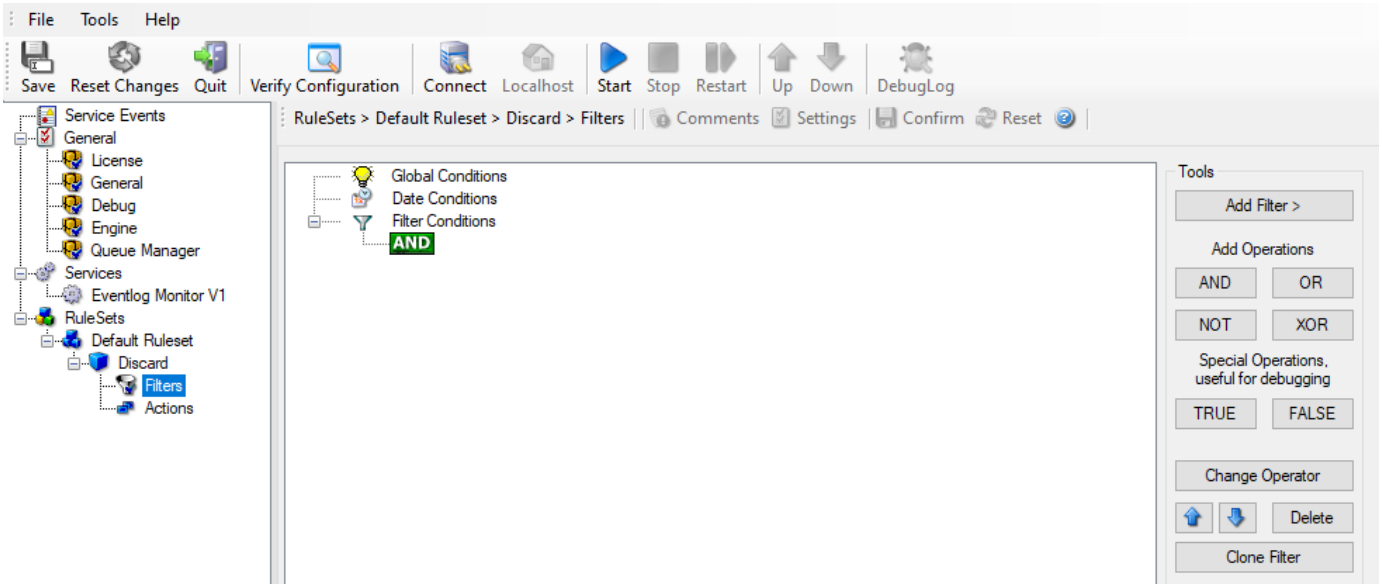
Now we do something with the data that is generated by the Event Log Monitor. To do so, we must define rules inside the ruleset.

In the tree view, right-click “Defaults” below “RuleSets”. Then, click “Rules”, select “Add Rule”. Choose any name you like. In our example, we call this rule “Discard”. Then, expand the tree view until it looks like the following screenshot:



- Ignoring Events - 3*

Click on “Filter Conditions” to see this dialog:

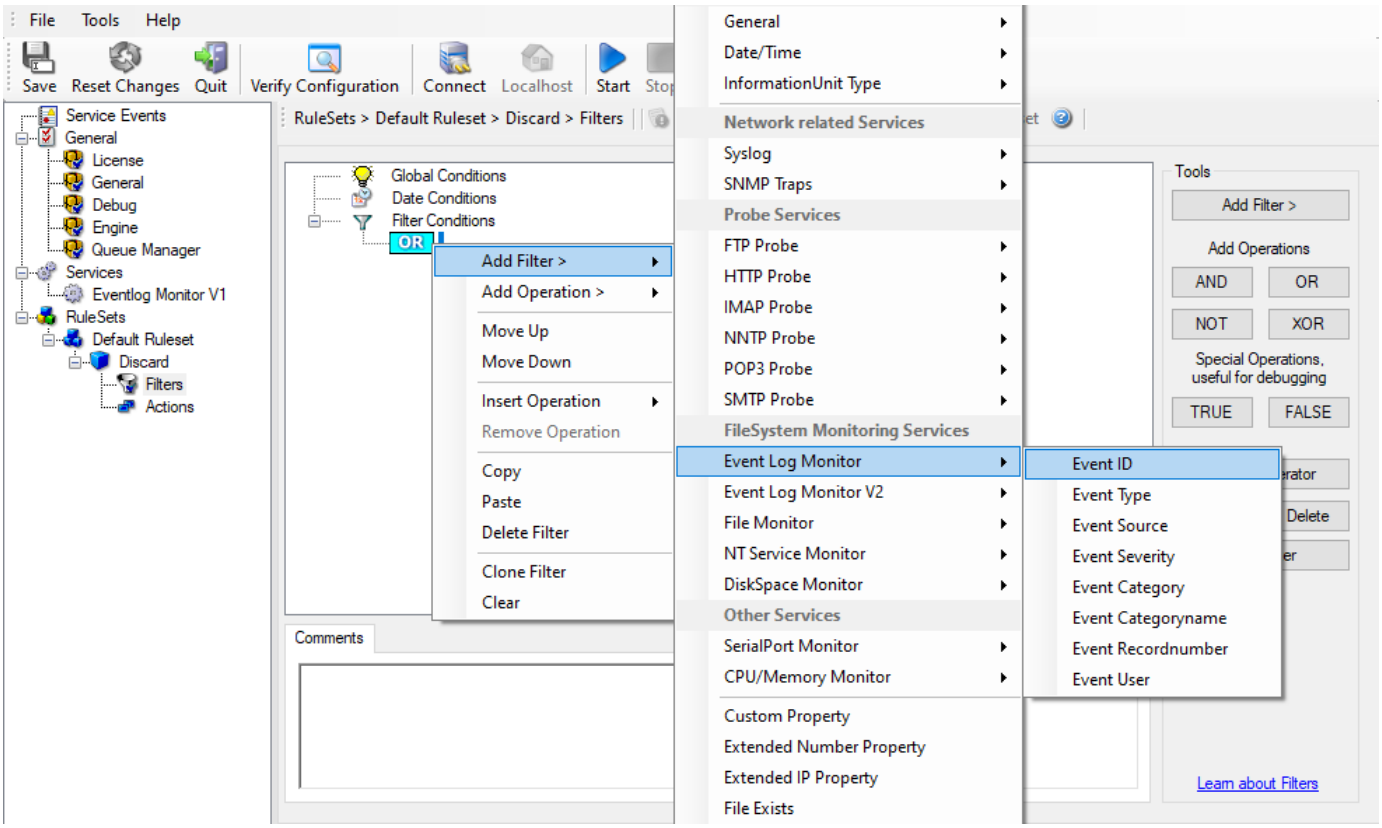


- Ignoring Events - 4*

In that dialog, we define our filter. Remember: we are about to filter those events, which we are not interested in. As we would like to discard multiple events, we need the Boolean “OR” operator in the top-level node, not the default “AND”. Thus, we need to change the Boolean operator.

There are different ways to do this. Either double-click the “AND” or click “Change Operator” - you find it below tools on the right side - to cycle through the supported operations. In any way the Boolean operation should be changed to “OR”.

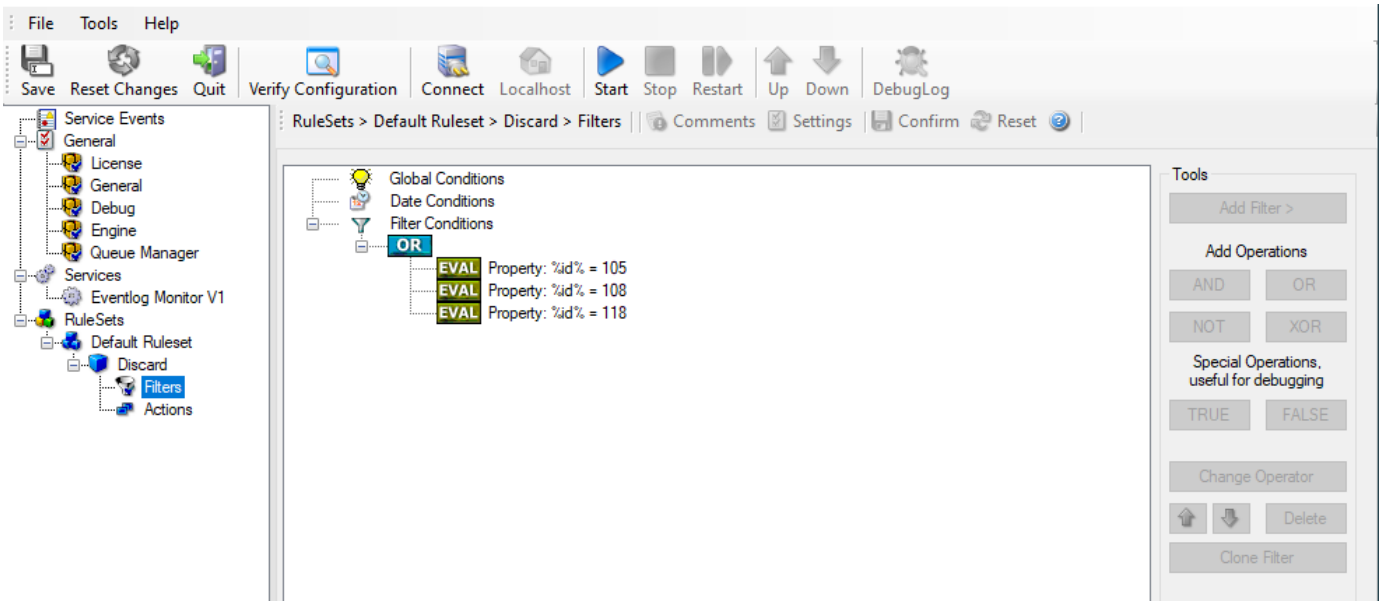
We filter out “uninteresting” events via their event id. Again, there are different ways to do this. You may use the Tools at the right side to click the “Add filter” button. In the sample, we do it via right-clicking the “OR” node and selecting “Add Filter” from the pop up menu. This can be seen in the screen shot below:



- Ignoring Events - 5*

I prefer to add all three Event ID property filters first and later on change the Event ID to the actual value I am looking for.

In order to enter the actual values, select each of the three filters. A small dialog opens at the bottom of the screen. There you enter the values you are interested in. In our sample, these are IDs 105, 108, and 118. As we are only interested in exactly these values, we do a comparison for equality, not one of the other supported comparison modes. When you have made the updates, you screen should look as follows:

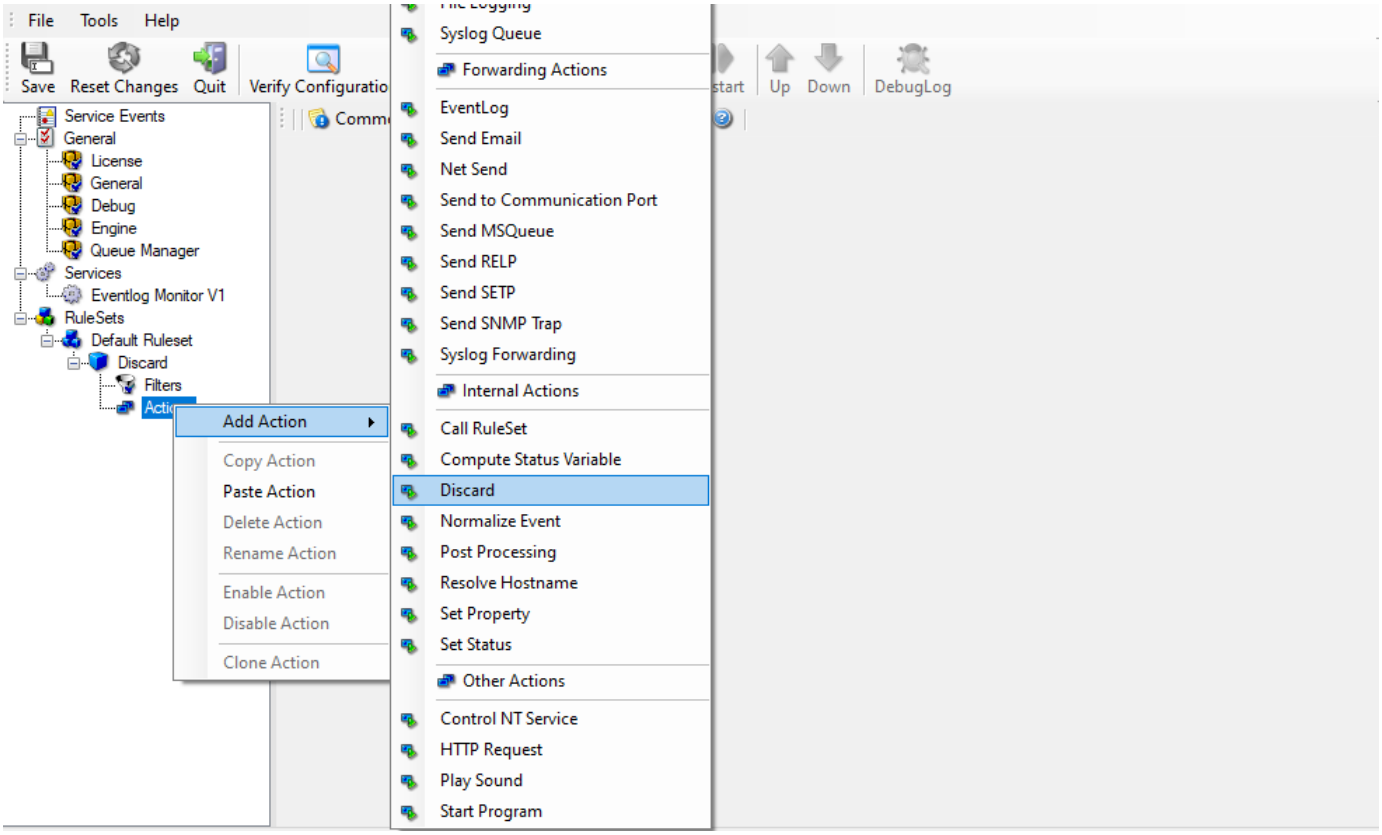


- Ignoring Events - 6*

Save the settings by clicking the (diskette-like) “Save” button. We have now selected all events that we would like to be discarded. In reality, these are often far more or a more complicated filter is needed. We have kept it simple so that the basic concept is easy to understand – but it can be as complex as your needs are.

Now let us go ahead and actually discard these events. This is done via an action. To do so, right-click on “Actions” and select “Discard.”

Again, name the action as you like in the following dialog. We use “Discard” as this is quite descriptive.



- Ignoring Events - 7*

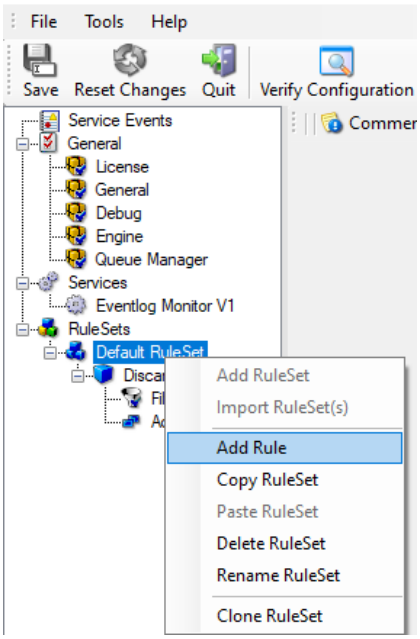
This concludes the definition of our first rule.

If we would start MonitorWare Agent service now, all events with IDs 105, 108 and 118 would be handled by this rule and thus be discarded. All other events do not cause the filter condition to evaluate to true and thus those would be left untouched. Consequently, only these other events flow down to rules defined behind the “Discard” rule. Obviously, our configuration effort is not yet completed. We just finished a first step, excluding those events that we are not interested in. And of course, in reality you need to decide which ones to discard in a real ruleset.

Logging Events

Often, a broad range of events (or information units as we call them) needs to be stored persistently so that you can review and analyze them if the need arises. As such, we are in need of a rule that persists the events. In our sample, we choose to work with a text log file (not a database, which we also could use). We now create a rule to store all those events not discarded by the previous rule into a log file.

To do so, right-click the “Defaults” ruleset as shown below. Then, select “Rules” and “Add Rule”:

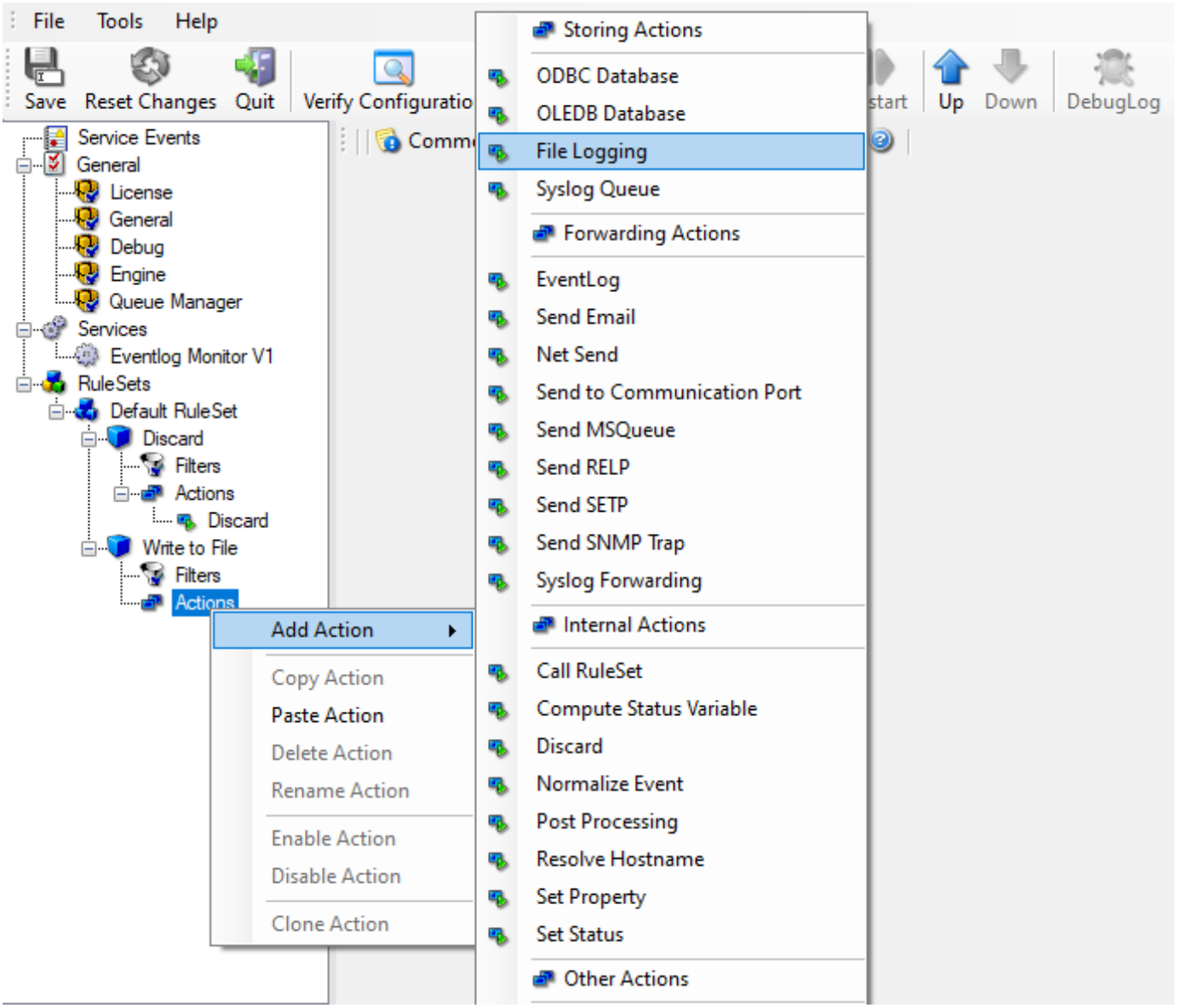


- Logging Events - 1*

Use a name of your choice. In our sample, we call this rule “Write to File”. This rule should process all events that remained after the initial discard rule. As such, we do not need to provide any filter condition (by default, the filter condition matches always).

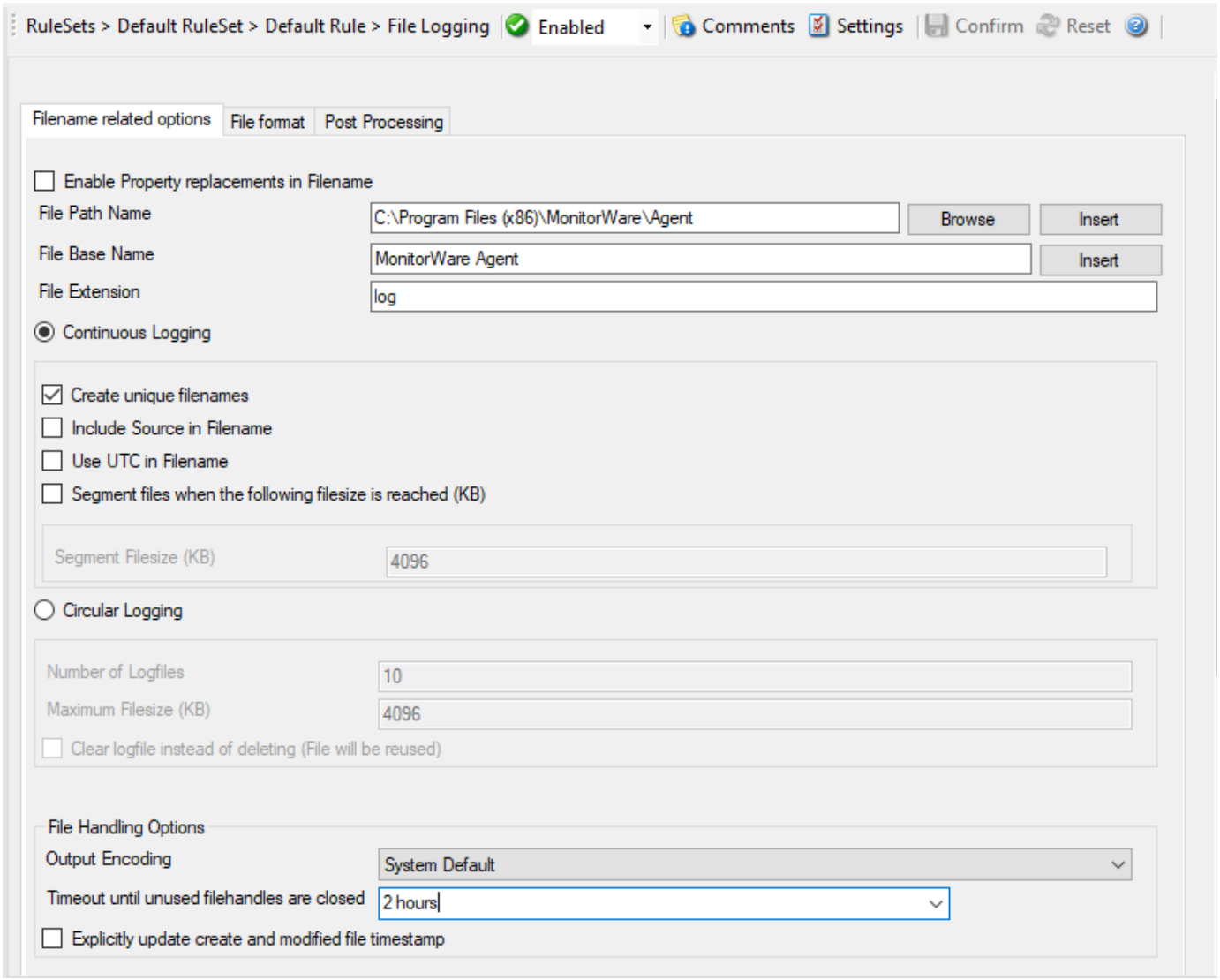
Since we want to store all still open Events with help of this rule, we do not require any filter rules here. However, a corresponding action must be defined. Therefore, we just need to define the action:

To do so, expand “Write to File” and right-click “Actions”. Select “Add Action”, then “File Logging” as can be seen below:



- Logging Events - 2*

Do not modify the defaults. In our sample, we call this action “File Logging”. Now the tree view contains a node “File Logging”, which we select:



- Logging Events - 3*

Important

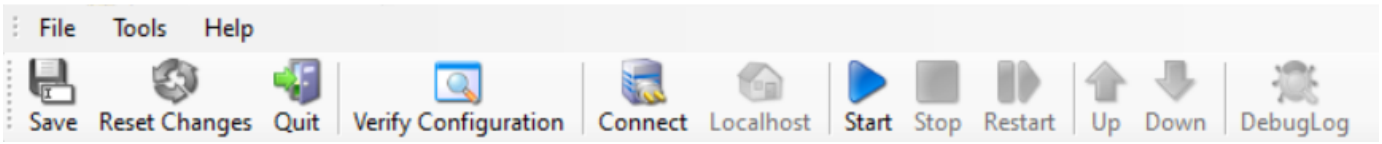
If the configured directories are missing, they are automatically created by MonitorWare Agent i.e. the folder specified in “File Path Name”.

In our sample, we change the file base name to “logdata”. This was just done out of personal preference. There is no need to do so, but it may be convenient for a number of reasons.

Summary

What did we do so far? All events from the Windows Event Log are passed through our rule engine and rule filters. Certain events are discarded and the remaining events are stored to a text file on the local disk (for later review or post-processing).

We can now do a quick test: Start MonitorWare Agent by hitting the start button seen below:



- Logging Events - 4*

The log file should be created in the path you have specified. Open it with notepad. You should see many events originating from the event log. When you re-open the log file, new events should appear (if there were any new events in the Windows Event Log). The file is not easily readable. Most probably, you have created it for archiving

purposes or to run some external scripts against it. For review, we recommend to try free Adiscon Loganalyzer open source project.

Please note that the current date is appended to the log file. This facilitates file management in archiving. The format is “logdata-YYYY-MM-DD.log”.

You have now learned to define rules and actions. The following chapters thus does not cover all details of this process. If in doubt, refer back to these chapters here.

Time-Based Filters

Time-based filters are especially useful for notifications. For example, a user login is typically a normal operation during daytime, but if there are no night shifts, it might be worth generating an alert if a user logs in during night time. Another example would be a backup run that routinely finishes during the night. If we see backup events during the day, something might be wrong.

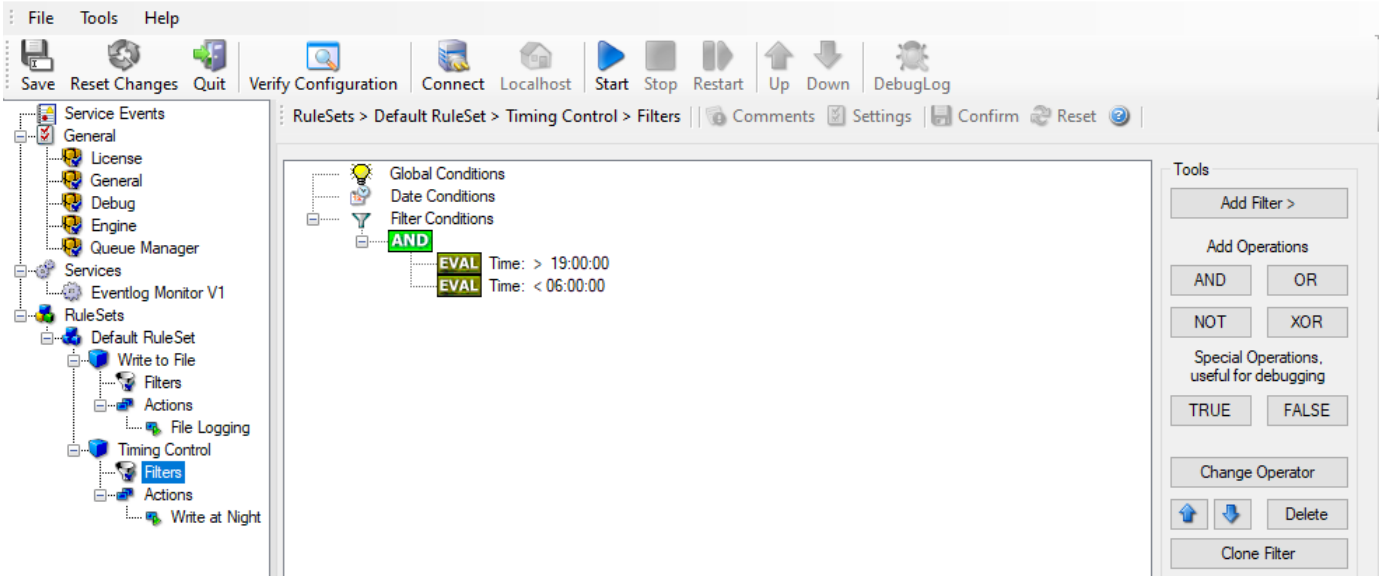
Similarly, there are a number of other good reasons why specific actions should only be applied during specific time frames. Fortunately, MonitorWare Agent allows defining complex time frames. In this tutorial, though, we focus on the simple ones.

Let us first define a sample time-based filter that applies a nightly time frame. In fact, there are many ways to do this. We have used the method below, because it is straightforward and requires the least configuration work.

To make matters easy, we use this filter condition just to write nightly event log data to a different log file. In reality, time-based filters are often combined with other conditions to trigger time based alerts. However, this would complicate things too much to understand the basics.

In the sample below, an additional rule called “Timing Control” has been added. It includes a time-based filter condition. Only if that condition evaluates to “true”, the corresponding action is executed. This action can be “Write to Database” or “Write to File”. Here we had selected “Write to File” action and renamed it as “Write at Night”.

Please note: we use the 24-hour clock system.



- Time-Based Filters - 1*

All events generated by services binding to our ruleset “Defaults” are now also be passed along the “Timing Control” ruleset. If these events come in night times between `19:00:01` and `5:59:50`, the action “Write at Night” is executed.

Please note that the use of the “OR” operator is important because either one of the time frames specified does apply. This is due to the midnight break.

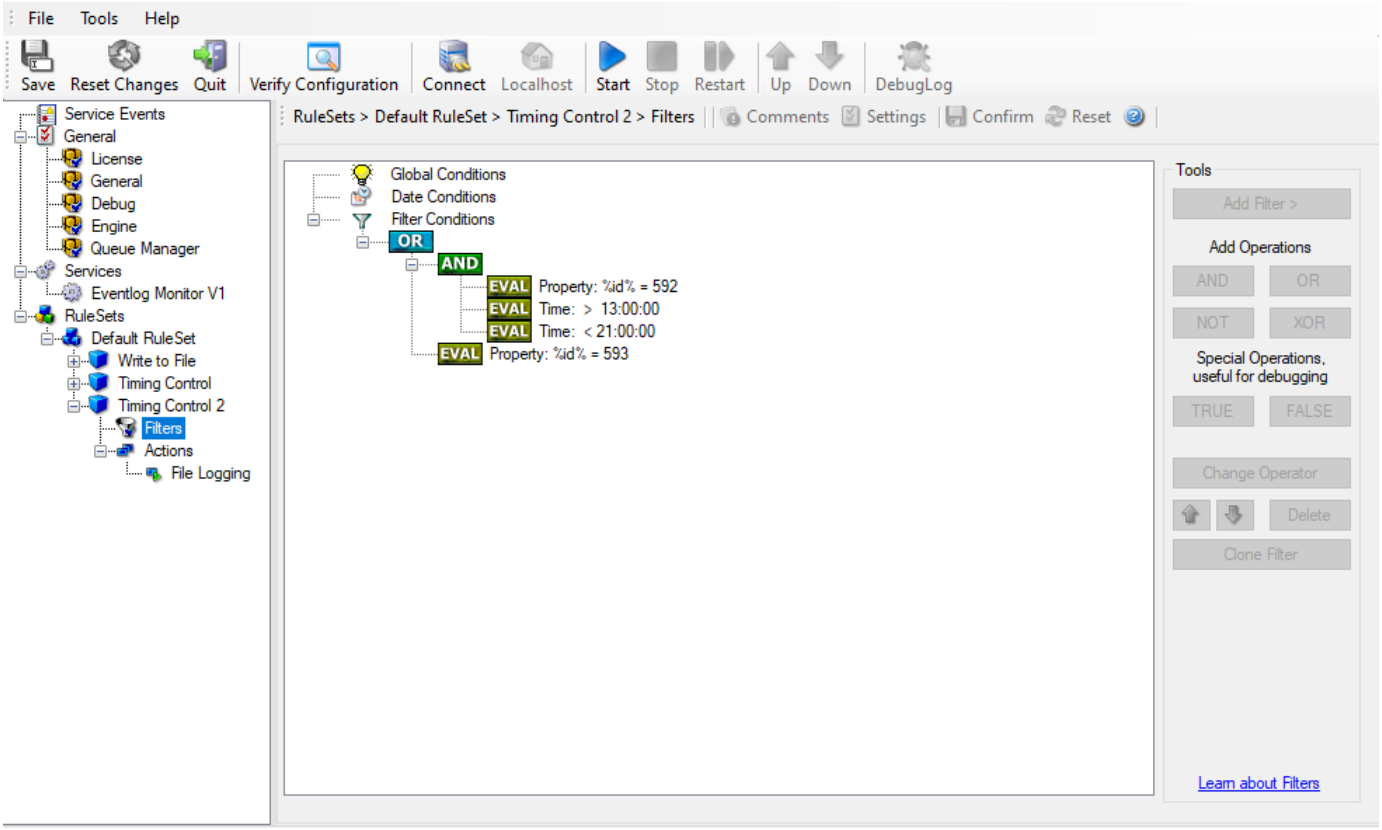
If an event comes in at `08:00:00 AM` in the morning, the action is not called – it is outside of the specified time frame:

```
08:00:00 > 19:00:00 = false
08:00:00 < 06:00:00 = false
```

If the very same event comes in at `20:00:00` it evaluates to true and the action is executed.

```
20:00:00 > 19:00:00 = true
20:00:00 < 06:00:00 = false
```

As stated earlier, time frames are most often used in combination with other filters. Here is a more complex example:



- Time-Based Filters - 2*

In this example, we call the configured actions if events with ID 592 occurs between``13:00:01`` and `20:59:59` (roughly `21:00:00`). We also execute the configured actions if event ID 593 occurs. Please note that in the case of 593 events, the time filter does not apply due to the used Boolean operations.

In this sample, you also notice that we use an “AND” condition to build the time frame. The reason is that there is no implicit midnight boundary for our time frame as was in the first sample. As such, we need to employ “AND” to make sure the events are WITHIN the specified range.

Now let us look at some sample data:

We receive a 592 event at `07:00:00` sharp:

```
Event ID = 592          = true
07:00:00 > 13:00:00     = false
07:00:00 < 21:00:00     = false
"AND" Branch            = false
Event ID = 593          = false
```

In all, the filter condition is false.

Now, the same event comes in at `14:00:00`:

```
Program start ID = 592  = true
Event ID = 592          = true
14:00:00 > 13:00:00     = true
14:00:00 < 21:00:00     = true
"AND" Branch            = true
Event ID = 593          = false
```

This time, the time frame is correct, yielding to an overall true condition from the “AND” branch. That in turn yields to the filter condition as whole to evaluate to true.

In this example still is another Event ID. All events with event ID 593 are grasped. This happens independently from the timing control when grasping the Events 592.

One last sample. At this time, event 593 comes in at `07:00:00`:

```
Program start ID = 593  = true
Event ID = 592          = false
07:00:00 > 13:00:00     = false
07:00:00 < 21:00:00     = false
"AND" Branch            = false
Event ID = 593          = true
```

This time the filter condition evaluates to true, too. The reason is that the (not matched) time frame is irrelevant as the other condition of the top-level “OR” branch evaluates to true (Event ID = 593).

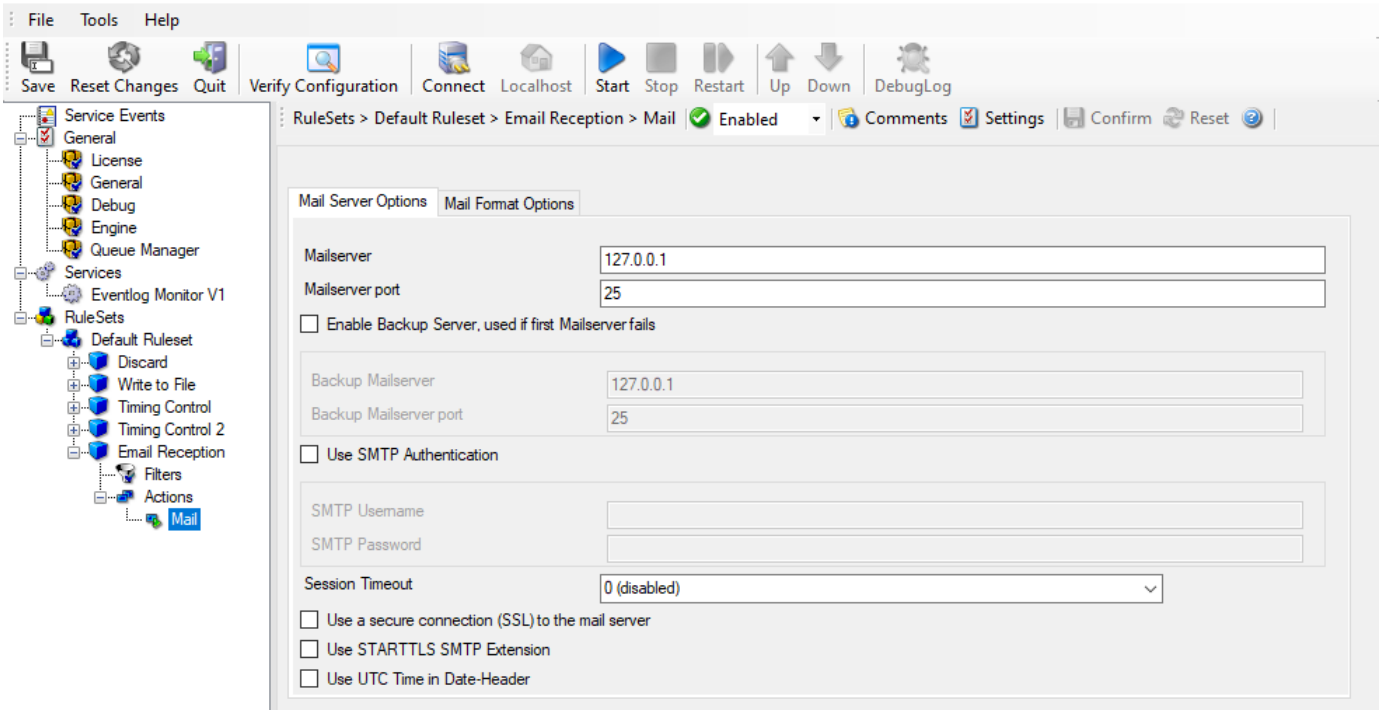
Email Notifications

In this example, we would like to receive email notifications when certain events happen.

So let us create an additional rule for that purpose: Right-click the “Default ruleset” and select “Add Rule” from the pop up menu. Provide a name. We call it “Email Reception” in this example. Then, chose below the “Forwarding Actions”: “Send Email”.

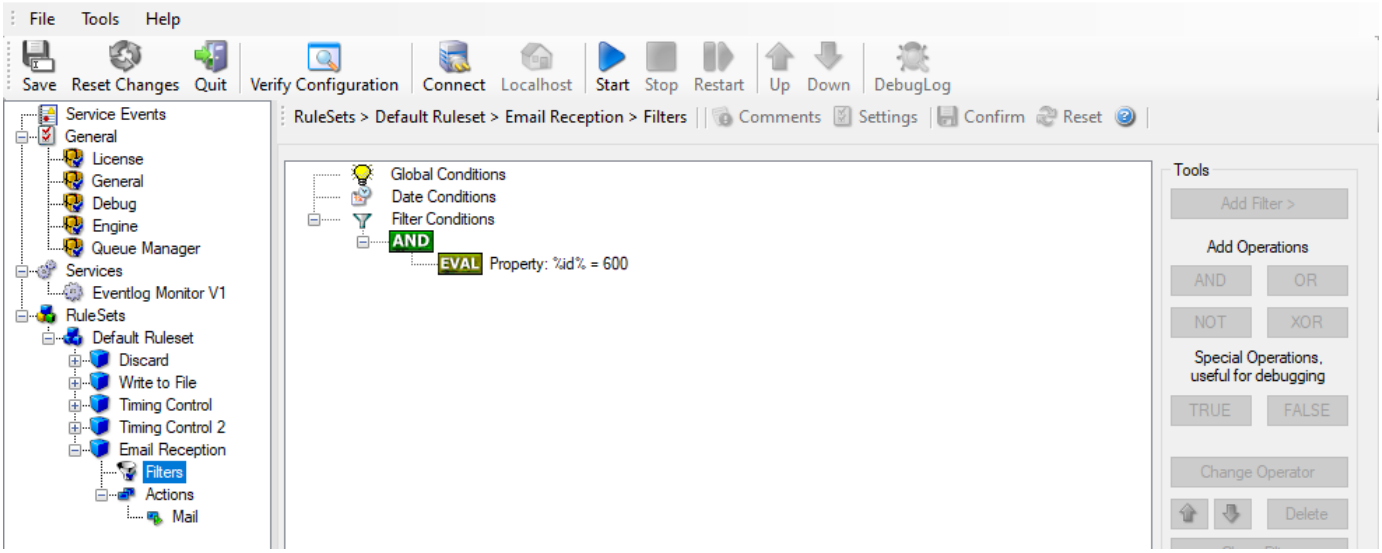
In the action details, be sure to configure at least the mail server, recipient and subject properties.

Please note that many mail servers also need a valid sender mail address otherwise they deny the delivery of the message.



Email Notifications - 1

Then, select the filter conditions. Let us assume we are just interested in events of ID 600. Then the filter conditions should look as shown below:



Email Notifications - 2

When you have finished these steps, be sure to save the configuration and restart the MonitorWare Agent service. After the restart, the newly extended ruleset is executed. In addition, the rules defined so far, the new one is carried out, emailing all events with ID 600 to the specified recipient.

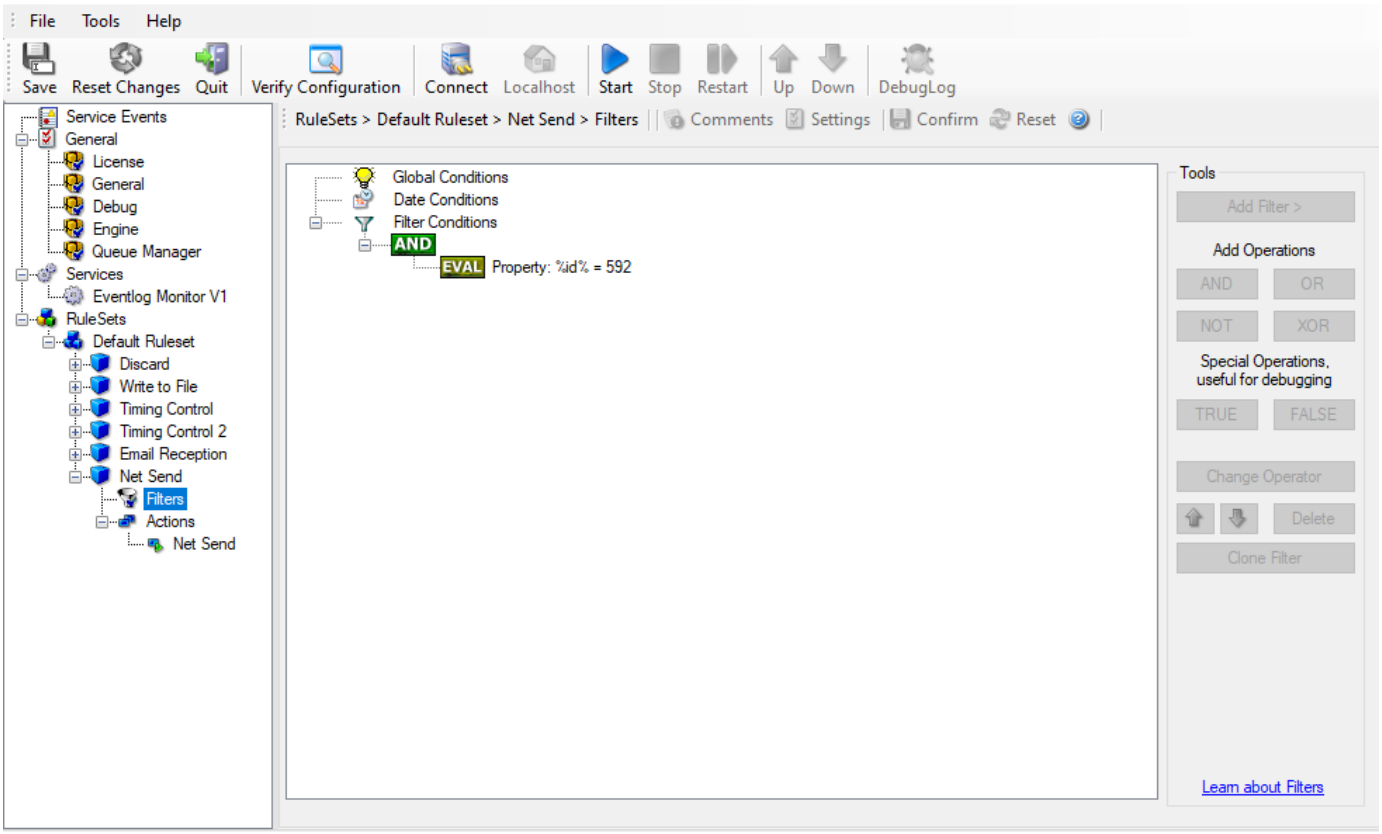
Alarming via Net Send

Attention! Due changes in all supported Windows Version, the net send functionality has been disabled by default. Using the Net Send Action will not work unless you enable certain policies on your Windows Installations.

Again, we add another rule to our ruleset. This time, we would like to receive notification via the Windows messenger service (aka “net send”).

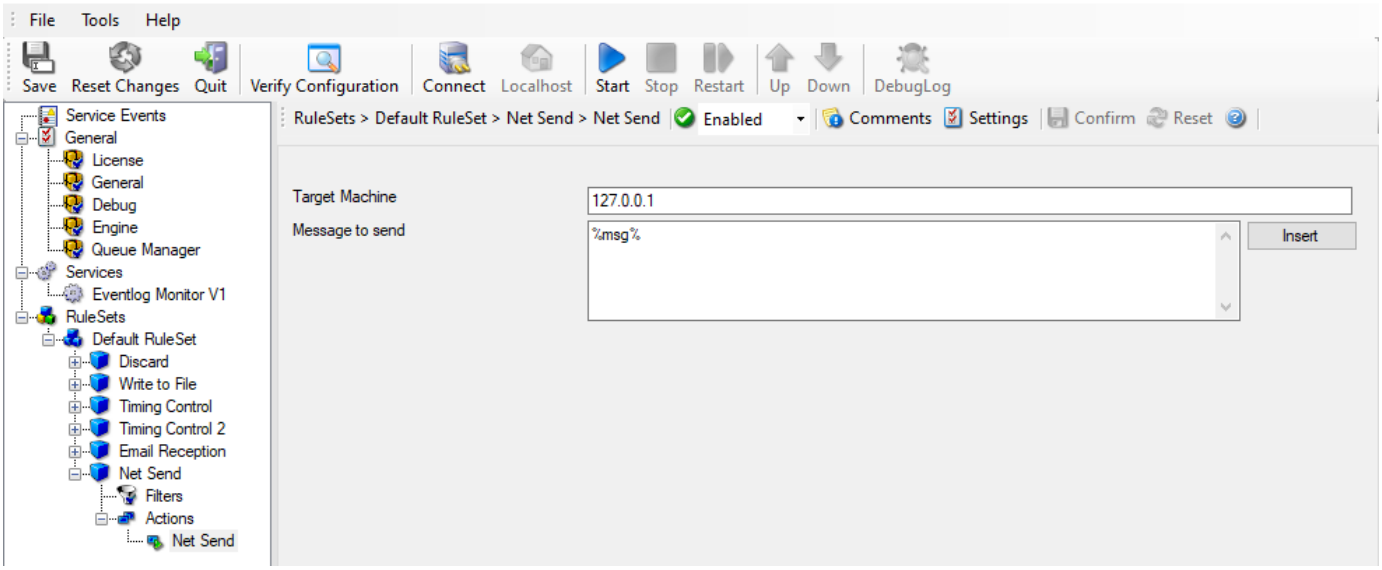
Please bear in mind that the Windows messenger service is not the instant messaging service that many people nowadays associate with it. The messenger service is meant for administrator notifications. If a windows workstation (or server) receives a message via that service, a message box pops up on that workstation and the user needs to press an “OK” button to continue. No interaction is possible.

We create a new rule in our ruleset “Defaults”. In this case, we assume that we receive messenger notifications for all events with Event ID 592. In a real use case, you make sure that this is a real important event, or chances are good you become overwhelmed with messaging windows. A better example could be a filter that checks for a server running low on disk space (using the disk space monitor).



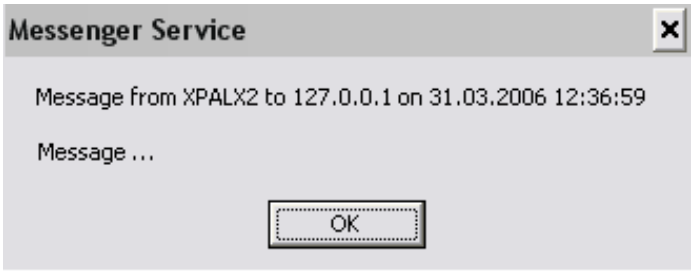
Alarming via Net Send - 1

This time, we use the “Net Send” action as can seen below. The target field holds either the name or IP-Address of the workstation this message should be sending to. The message text itself goes into “Message to send”.



Alarming via Net Send - 2

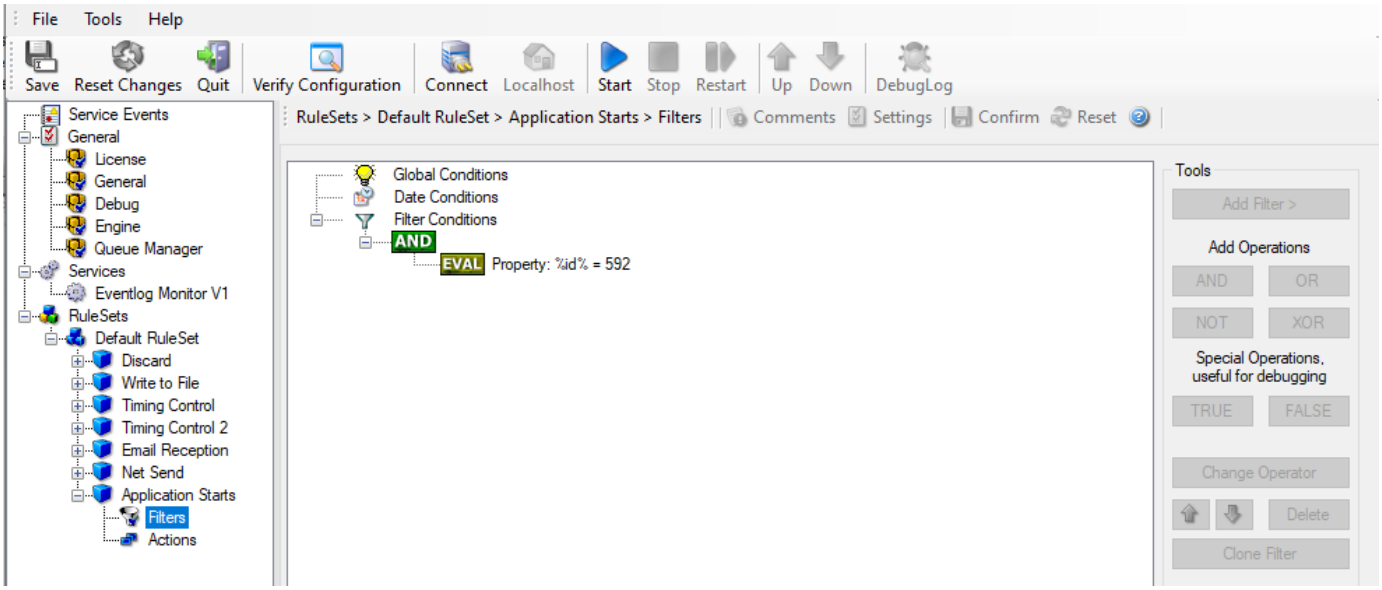
After saving the configuration and restarting the MonitorWare Agent, we receive notifications if the filter condition evaluates to true. A sample message might look like this (slightly obscured in this sample):



Starting Scripts and Applications in Response to an Event

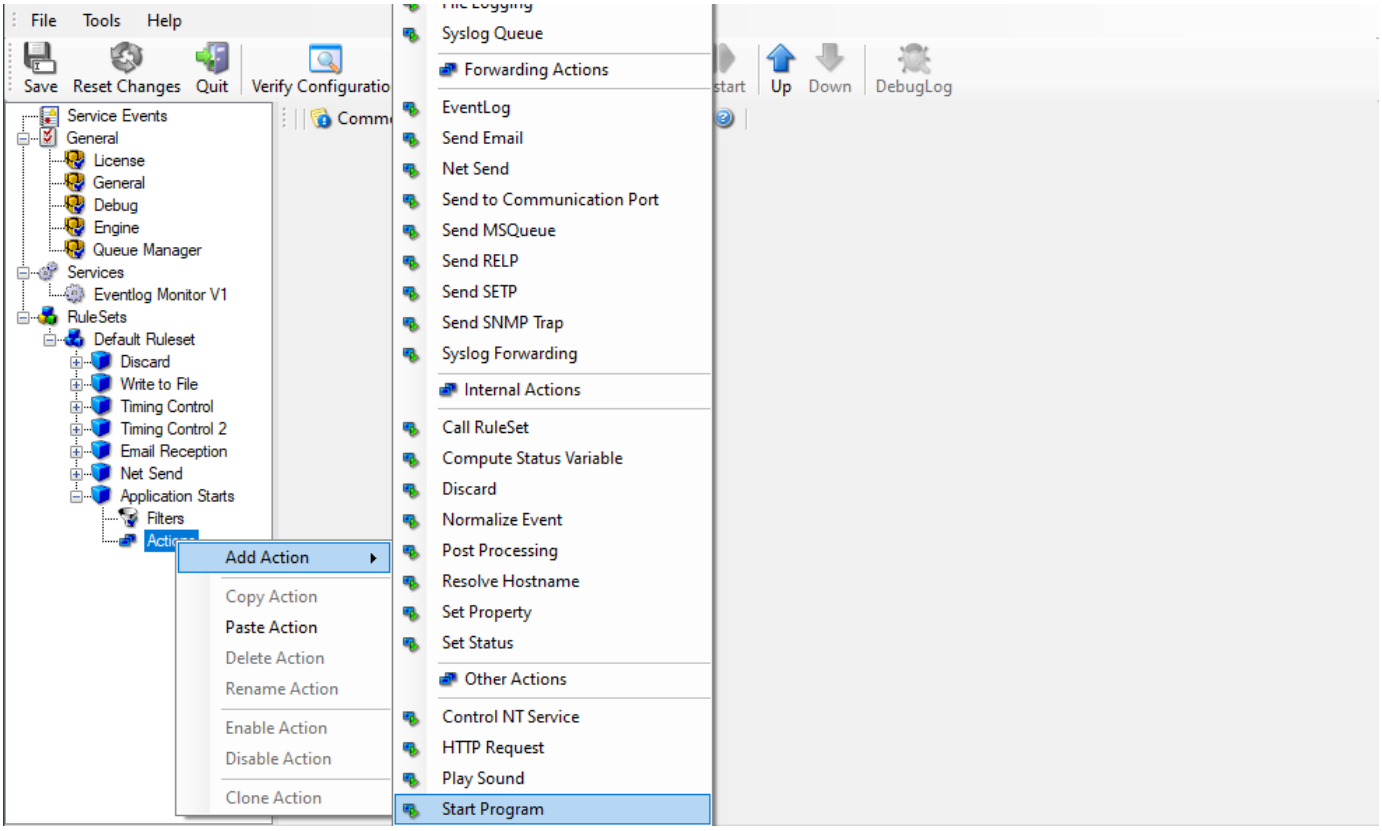
We now want to start an application or a script when certain events occur. Typically, this is done to start administrative scripts or corrective action. For example, if a disk runs low on space, you could start a script that deletes temporary files, or if a service fails, a script could restart it.

Our sample, on the other hand, is kept quite simple again. We just show how to generically start an exe file. To do so, we define a new rule named “Application starts”. Again, we use the imaginary event 592 as a filter condition. Therefore, the application starts whenever event 592 comes in.



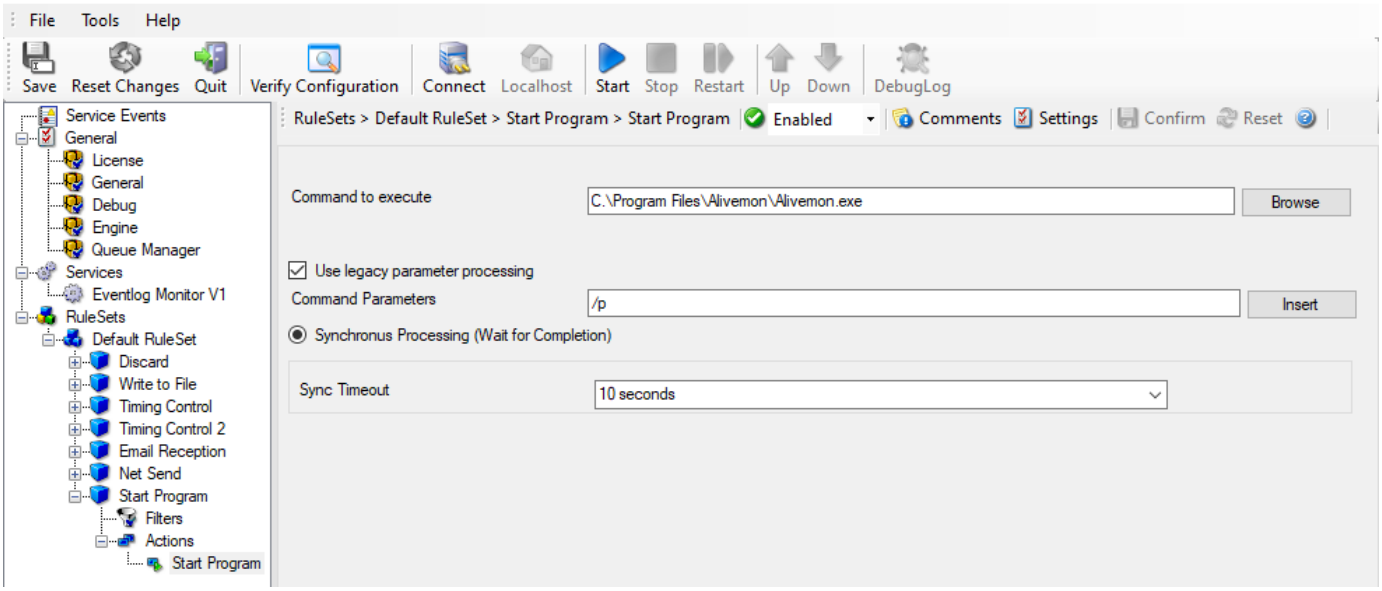
- Starting Scripts and Applications in Response to an Event - 1*

The start program action is just a “normal” action:



- Starting Scripts and Applications in Response to an Event - 2*

In the “Start Program” action’s parameters select the file to run. Also all parameters need to be supplied to it (if any):



- Starting Scripts and Applications in Response to an Event - 3*

Once this configuration is done, the program is executed as soon as an event matching the filter condition comes in.

Step-by-Step Guides

The step-by-step guides are meant to get you started quickly. They provide information on how to configure the product in common scenarios. Each section includes the information necessary to complete a specific task.

The information is presented in an easy to follow “step by step” way (hence the name). Each section begins with the intended result and then explains the steps to achieve it in the correct order. They are documented together with hardcopies, so they should be easy to follow. For best results, please be sure to follow the exact order of the steps.

The step-by-step guides do eventually not include all information that might be relevant to the situation. Please use your own judgment if the scenario described sufficiently matches your need.

In the step-by-step guides, we assume the product is already successfully installed but no configuration has been done. If it is not installed, please do so first.

All step-by-step guides assume that the client is running. This is kind of a step 0 for all the guides.

Installations and Configurations

How to enter the license information

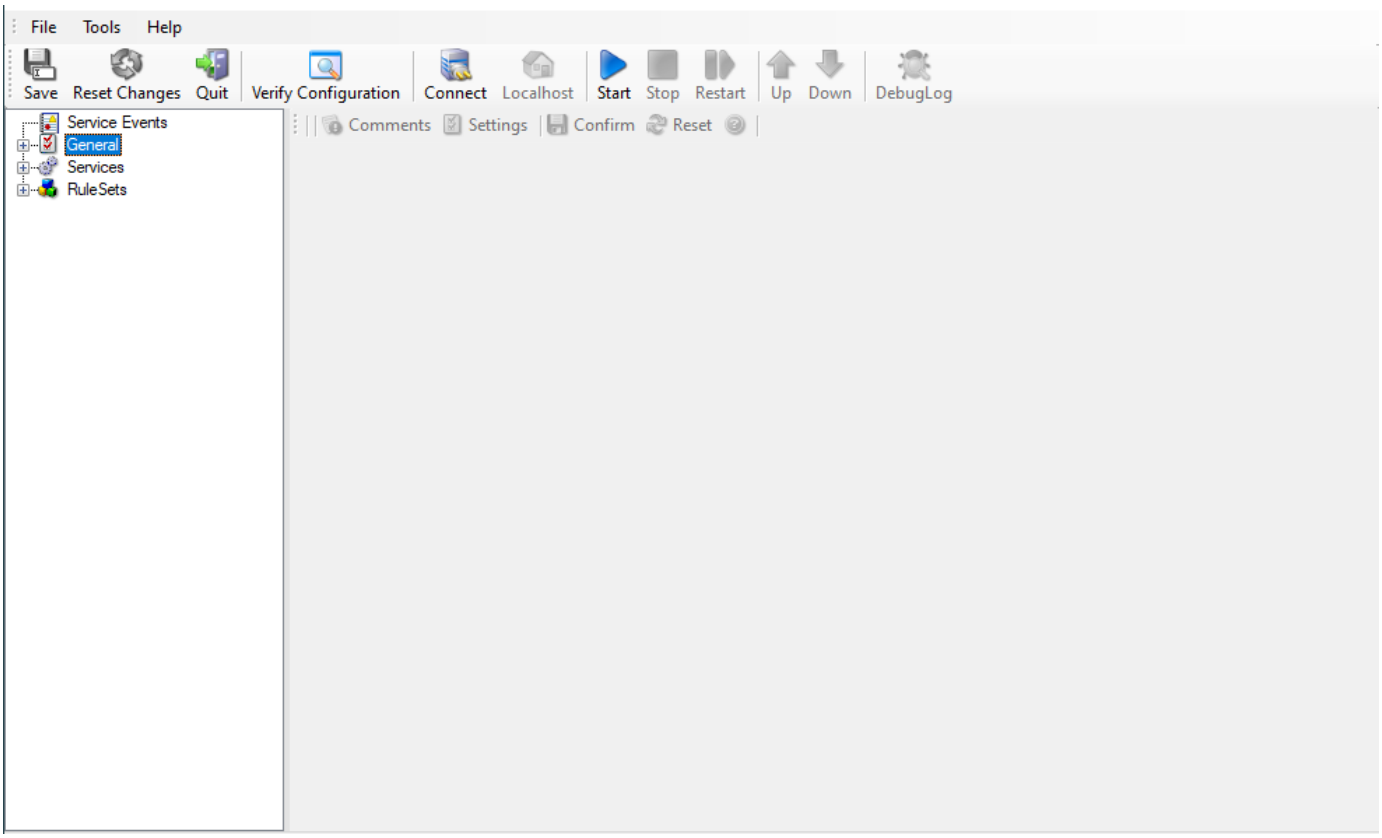
Article created 2021-10-05 by adiscon team

This article describes how to enter the license information you received via mail by buying one of our products.

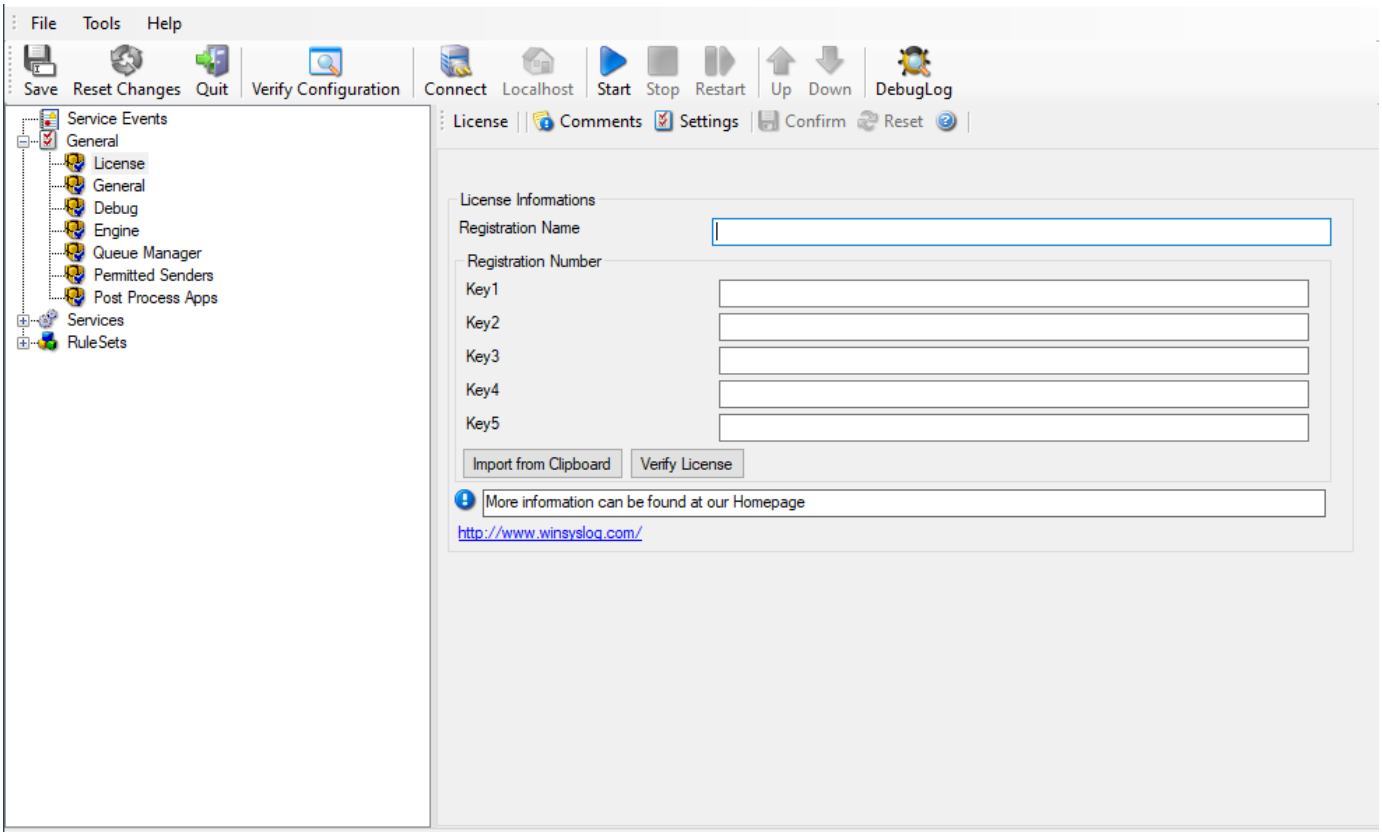
The Article is applicable to EventReporter, MonitorWare Agent, WinSyslog and Rsyslog WindowsAgent.

The license screen can be found on the left side of the client under the item General. Applying the license is very straightforward with only a few steps. After purchase, you will receive an email from us that contains the license name and key.

Under General - on the left side of the Configuration Client - you will find the menu entry: License.



If you click on it, you will find the license screen on the right.



The easiest way is to copy and paste the license name without quotation marks into the field “Registration Name” because it is case sensitive and must be entered exactly as given. Leading and trailing spaces are also part of the registration name. Be careful not to enter any.

Copy the full license key and use the button “Import from Clipboard” to paste it into the key fields. The client detects invalid registration numbers and reports the corresponding error.

Save the configuration and restart the service.

This is all that will be required to apply the license.

- forwarding filtered iis logfiles
- database logging with mssql in monitorware agent
- interactive logon/logoff filter?
- how do i apply filters in monitorware agent, winsyslog, and eventreporter?
- how to setup monitorware agent, winsyslog, and eventreporter?
- configuring windows for the eventlog monitor
- intrusion detection via the windows event log

services

- How to setup EventLogMonitor Service
- How to setup EventLogMonitor V2 Service
- Forwarding NT event logs to a Syslog Server
- Forwarding NT event logs to an SETP Server

Actions

- How to setup a Forward via Syslog Action
- How to setup a SETP Action
- How to setup a Write to File Action
- How to setup a send Mail Action
- How to setup a Set Status Action
- How to setup a Start Program Action
- How to setup a Control Windows Service Action
- Creating a Rule Set for Database Logging
- How to store custom properties of a log message in a database

Centralized Monitoring / Reporting

- How to setup Windows centralized Monitoring (MonitorWare Agent 5.x & MonitorWare Console 3.x)
- How to setup a central log server for Windows machines and syslog sending devices (MonitorWare Agent 4.x & EventReporter 8.x)
- How to setup Windows centralized Monitoring
- How to Report Log Truncation

You may also want to visit our syslog device configuration pages at https://www.adiscon.com/syslog-enabled-products/. They contain instructions on setting up several devices for syslog.

Configuring

EventReporter is easy to use and is powerful.

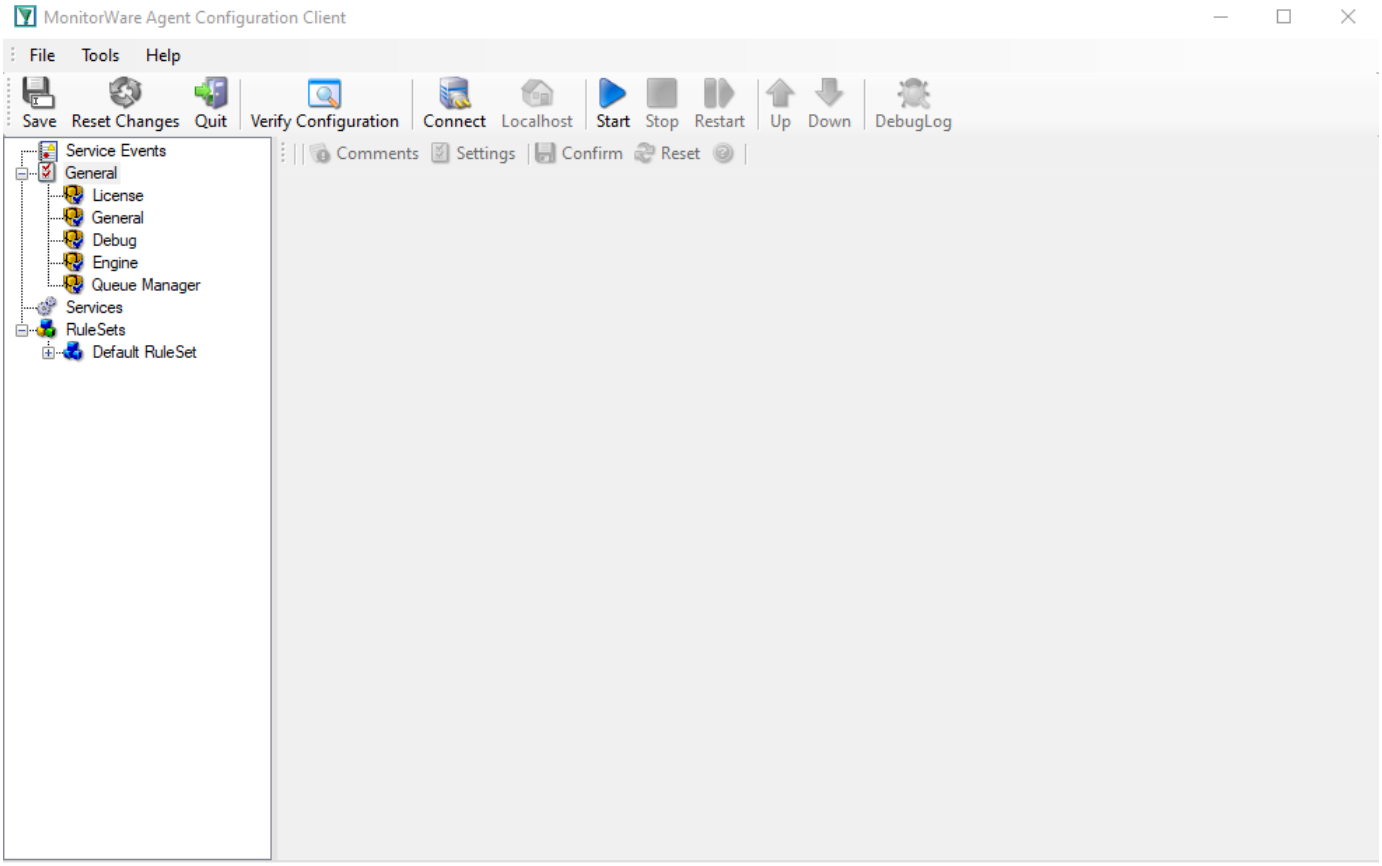
In this chapter, you will learn how to configure the EventReporter Service.

Configuring EventReporter

In this chapter, you see how to configure the EventReporter Service.

The EventReporter service runs in the background once it is configured. There is no manual intervention needed to operate it. As such, this chapter focuses on the EventReporter configuration client application. It is used to configure the service settings.

To run the EventReporter Configuration client, simply click its icon present in the EventReporter program folder located in the Start menu. Once started, a Window similar to the following one appears:



- Configuration Client*

The configuration client (“the client”) has two elements. On the left hand side is a tree view that allows you to select the various elements of the MonitorWare Agent system. On the right hand side, there are the parameters specific to the element selected in the tree view. In the sample above, the right hand side displays the specific parameters for a rule action.

The tree view has three top-level elements: General, Services, and RuleSets.

Under General, basic operational parameters as well as defaults for actions and services are defined. The default themselves do not activate anything. However, the parameters in here are used each time an actual service or action needs a configuration parameter and none is defined in that specific instance. We highly recommend putting the most common parameters into the defaults. That reduces the amount of data entry in the specific elements dramatically. Please note that each default can be overwritten in a specific service or action.

The tree view’s Running Services area lists all configured services as well as their parameters. There is exactly one service entry for each service created. Please note that there can be as many instances of a specific service type as your application requires. Typically, there can be multiple instances of the same service running, as long as their configuration parameters do not conflict. For example the Syslog service: there can be multiple Syslog servers on a given system as long as they listen to different ports. Consequently, there can be multiple instances of the Syslog service be created. For example, there could be three of them: two listen to the default port of 514, but one with TCP and one with UDP and a third one listens to UDP, port 10514. All three coexist and run at the same time. If these three services are listening to the same port then an error message is logged into Windows Event Log that more than one instance of Syslog server is running. After which MonitorWare Agent wouldn’t be able to perform the desired action.

Theoretically, you can run a few hundred services in a single service instance. However, both from a usage scenario point of view as well as concerning operating system resources, we recommend limiting the services to a maximum of 20 to 30. Of course, there are some applications where more than this limit is useful. MonitorWare Agent does not restrict this number. If there is a need for a large number of services and the hardware is capable of managing all these tasks, there is nothing in the MonitorWare Agent that limits from doing so.

The actual parameters depend on the service type. Common to all services is the capability to enable or disable a service. A service is started only if it is enabled. Otherwise, it does not run, but the configuration data can still be present. That way, it is easy to temporarily disable a service without deleting it.

Also common to all service types is the association to a ruleset seen at the bottom of the right hand configuration dialog. This specifies which of the rule sets have to apply to information units generated by this service.

To create a new service, right click on “Running Services”. Then select “Add Service” and the respective service type from the pop up menu. To delete an existing service, right click it and select “Delete Service”. This removes the service and its configuration is irrecoverable. To temporarily “Remove a service”, simply disable it in the property sheet.

The tree view’s last main element is RuleSets. Here, all rulesets are configured. Directly beneath are the individual rulesets. Each set is completely independent from each other. They are just centrally stored so they can be associated with services (see above for an explanation).

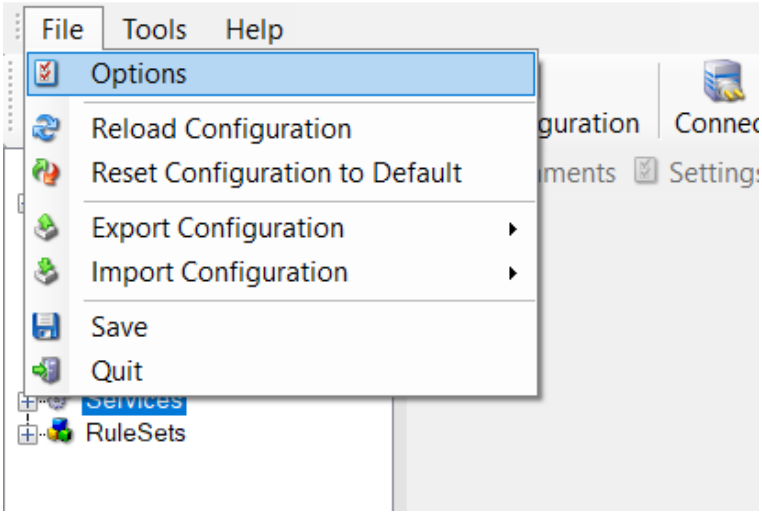
Beneath each ruleset are the individual rules. As described in Rules, a rule’s position in the list is vitally important. rules at the top of the ruleset are executed before those further down. To move a rule up or down, simply right click it and select “move up” or “move down” from the pop up menu.

In the tree view, filter conditions and actions are beneath the rule they are associated with. Finally, beneath actions are all actions to carry out.

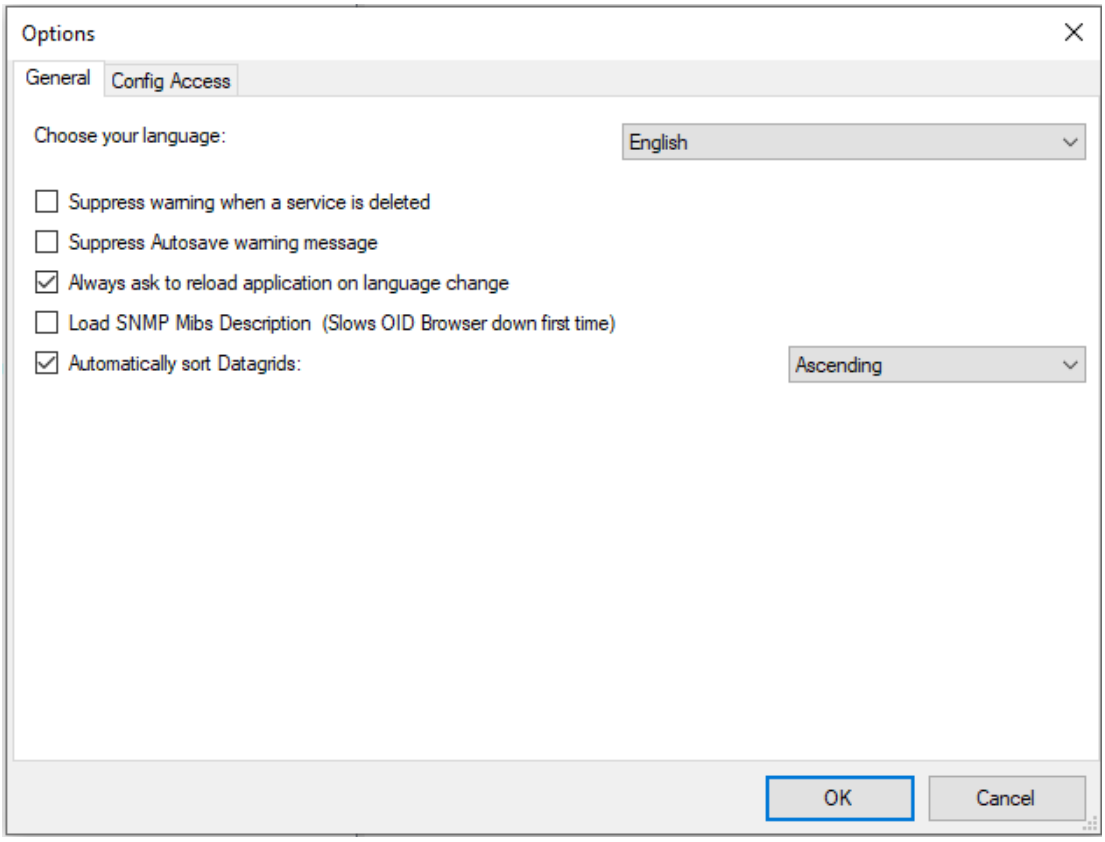
The following sections describe each element’s properties.

Client Options

There are several options, that refer to the configuration client and not to the service. These can be found under File -> Options



- Client Options*



- General Tab*

Choose your language

You can choose a language pack. “English” is the default and suggested language.

Suppress warning when a service is deleted

If this option is checked you will not get a warning when you try to delete a service and there is no other service that uses the connected ruleset.

Suppress autosave warning message

If you make changes in the configuration and switch to another component, a warning will occur if you haven’t saved the changes. This warning will also allow you to directly enable auto-saving the configuration.

Always ask to reload application after language change

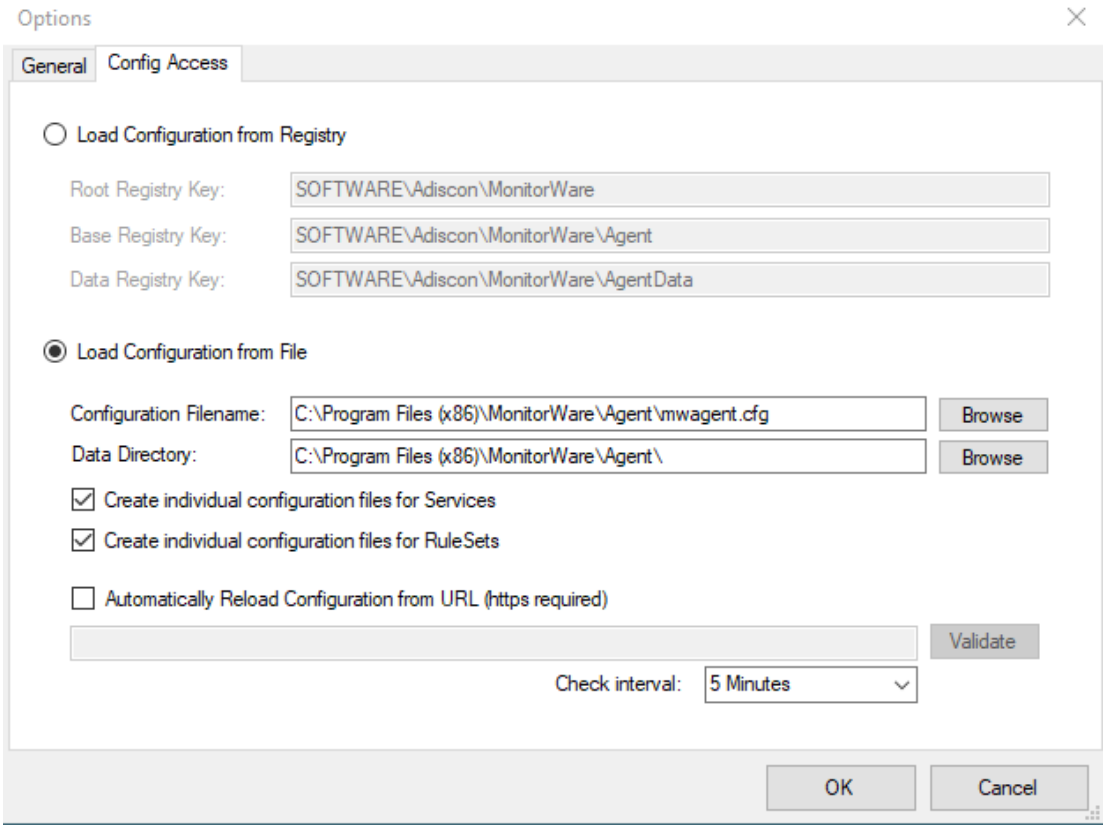
When you change the language, a popup will ask you to reload the configuration client to properly apply the changes and load with the set language.

Load SNMP Mibs Description (Slows OID Browser down first time)

If enabled, load SNMP Descriptions from MIB files (Client starts a little slower on startup).

Automatically sort Datagrids

Datagrids are used in certain areas within the configuration objects. You can change the default sorting behavior from ascending to descending here.



- Config Access Tab*

Load Configuration from Registry

The Configuration Client can be switched to a different registry path for configuration. The registry path change can be made permanent here. The changed registry path is the saved within the Parameters key of the Service.

Load Configuration from File

Alternatively, you can configure the service to load the configuration from a file. You can set the paths with the two fields below.

When enabled, the configuration will always be backed up before applying the new configuration. The backup consists of the last iteration and will be placed in the same directory.

Create individual configuration files for Services

Can only be enabled when “Load Configuration from File” is enabled. When enabled, the Services section of the configuration will be put into a separate file.

Create individual configuration files for RuleSets

Can only be enabled when “Load Configuration from File” is enabled. When enabled, the RuleSet section of the configuration will be put into a separate file.

Automatically Reload Configuration from URL (https required)

Only possible if File Configuration Mode is used.

If enabled, the configuration will be reloaded from a remote https location. Please note that a valid SSL certificate is required, or if custom certificates are used they have to be imported on the local machine properly.

If the remote configuration file can be downloaded from the configured location and differs from the current configuration, it will be installed automatically and the service will reload itself.

Check interval

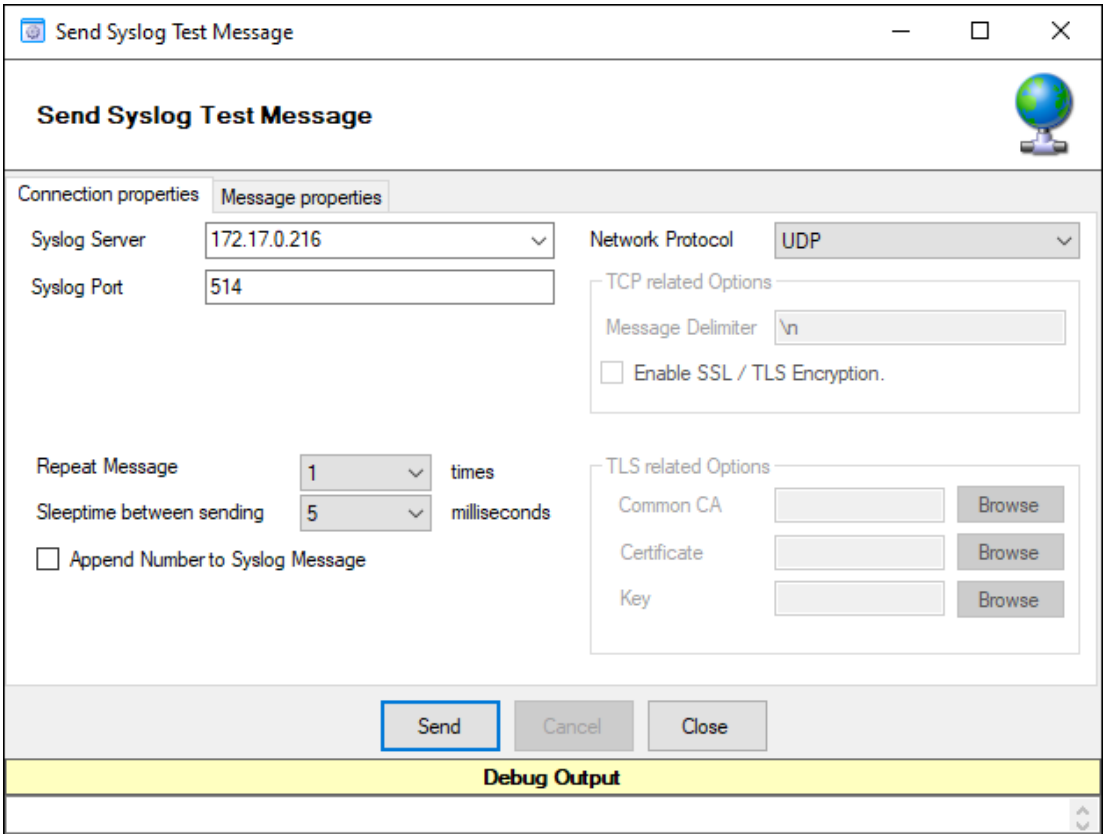
Specifies how often the service will check for remote configuration files. Please keep in mind that the configuration needs to be downloaded each time from the remote https url for comparison with the local one. We do not recommend to use a value lower then 5 minutes.

Client Tools

There are tools within the configuration client that you can use to test certain services or debug the application in general. Some can be found in the Tools menu.

Syslog Test Message

Opens a new windows which can send syslog test messages to Syslog Servers. This can also be opened within the configuration window of a Syslog service.



- Syslog Test Message Connection properties - UDP*

Syslog server

The hostname or ip address of the target Syslog server.

Syslog Port

The port that should be used to connect to the target Syslog server.

Repeat Message

How often you want to repeat the test message. Can be configured from 1 to 1000.

Sleeptime between sending

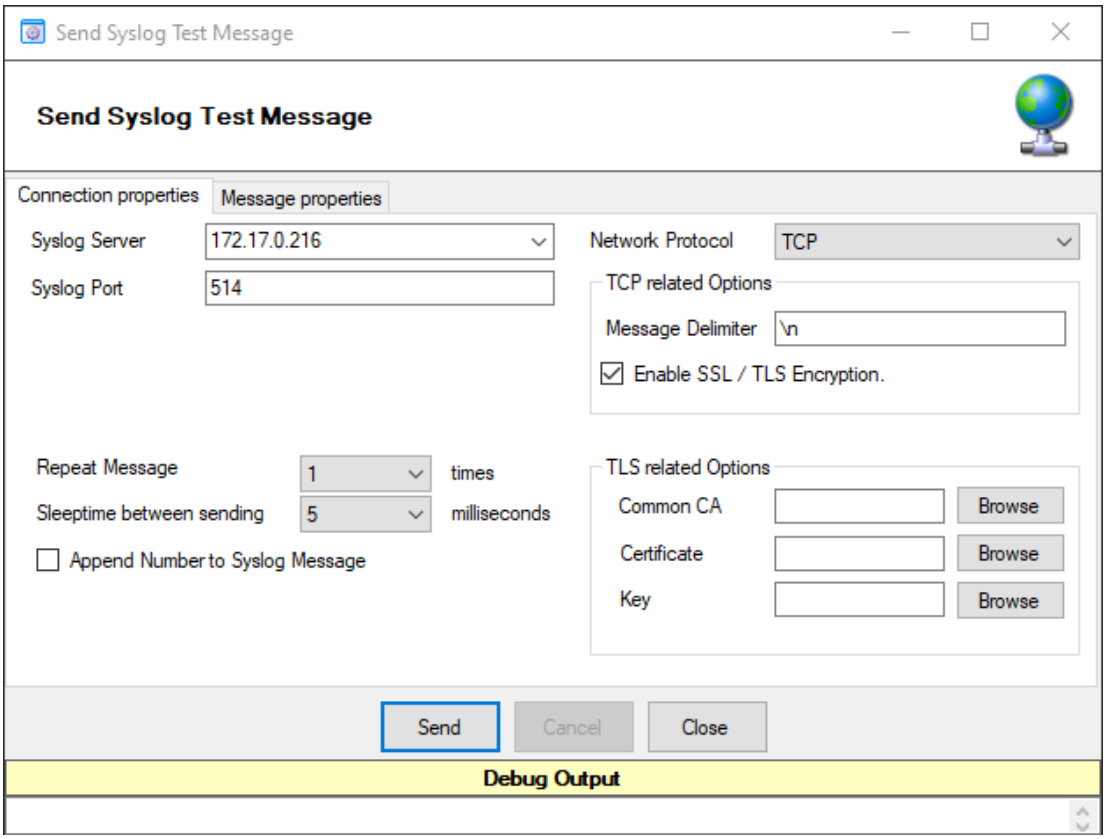
When using TCP, you can use 0ms. For UDP we recommend 1-5ms as sleeptime between sending syslog messages. Otherwise package loss can happen.

Append Number to Syslog Message

If sending multiple messages, enable this option in order to add a syslog number at the end of the message.

Network Protocol

Which network protocol should be used, either UDP or TCP can be selected.



- Syslog Test Message Connection properties - TCP*

Message Delimiter (TCP related Options)

When using TCP protocol, a message delimiter (separator) can be configured which is a simple linefeed by default.

Enable SSL/TLS Encryption (TCP related Options)

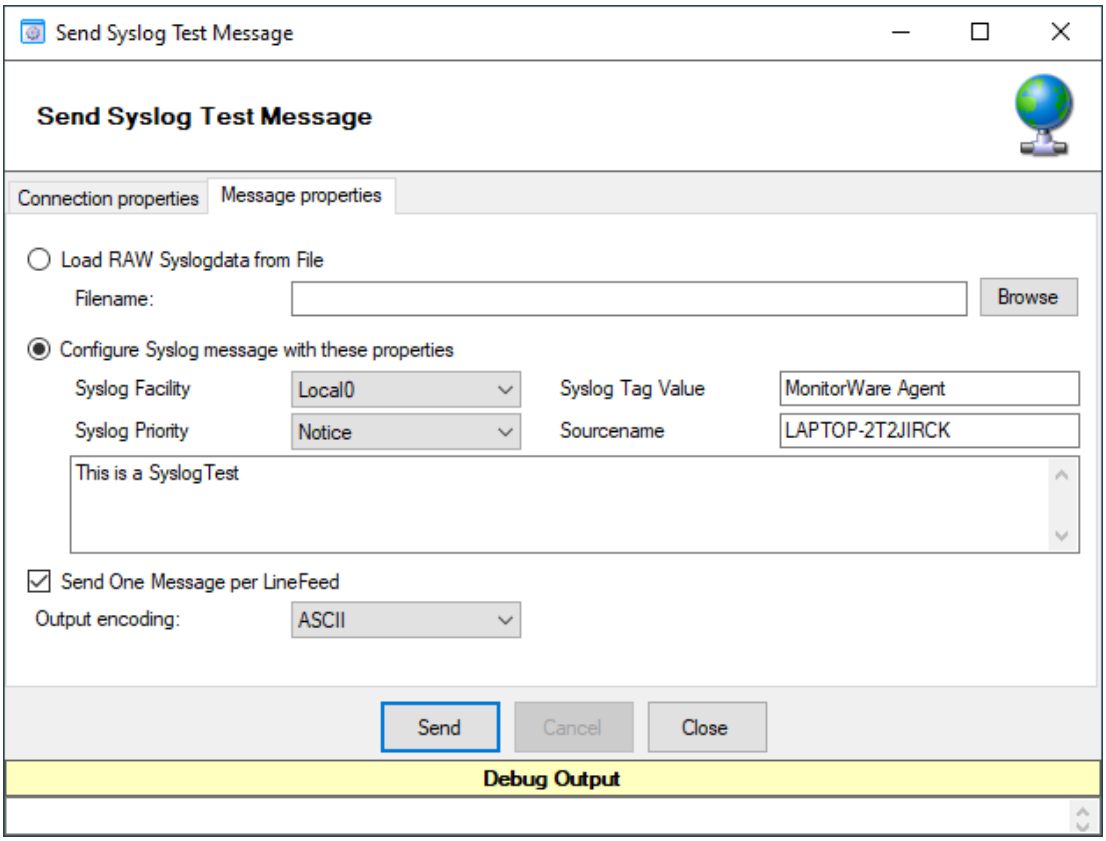
Check this option to enable the TLS related Options.

TLS related Options (TCP related Options)

Select common CA: Select the certificate from the common Certificate Authority (CA), the syslog receiver should use the same CA.

Select Certificate: Select the client certificate (PEM Format).

Select Key: Select the keyfile for the client certificate (PEM Format).



- Syslog Test Message Message properties*

Load RAW Syslogdata from File

You can choose to load raw syslogdata from file using this option. When loading UTF8 data make sure to set the Output encoding format from ASCII to UTF8. And if your file contains multiple syslog messages make sure that - Send One Message per LineFeed - is checked.

Configure Syslog message with these properties

Choose this if you want to configure all properties of the syslog message manually.

Send one Message per LineFeed

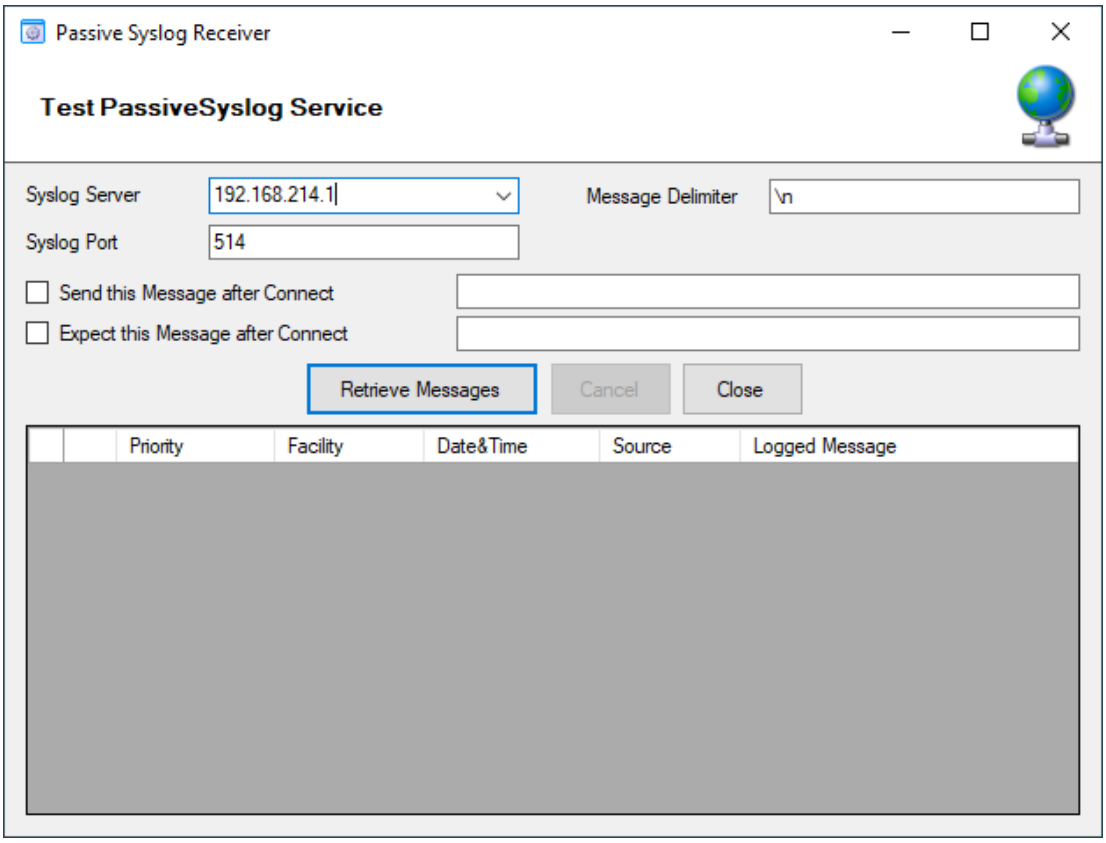
Check if your syslogdata contains multiple syslog messages divided by line feeds

Output encoding

Select the Output encoding you wish to use. When using UTF8, the UTF8 BOM is automatically prepended.

Passive Syslog Receiver

Opens a new windows to test Passive Syslog Servers. This can also be opened within the configuration window of a Passive Syslog service.



- Test Passive Syslog Service*

Syslog server

The hostname or ip address of the target passive Syslog server.

Syslog Port

The port that should be used to connect to the target passive Syslog server.

Message Delimiter

The message delimiter (separator) used to split syslog messages which is a simple linefeed by default.

Send this Message after Connect

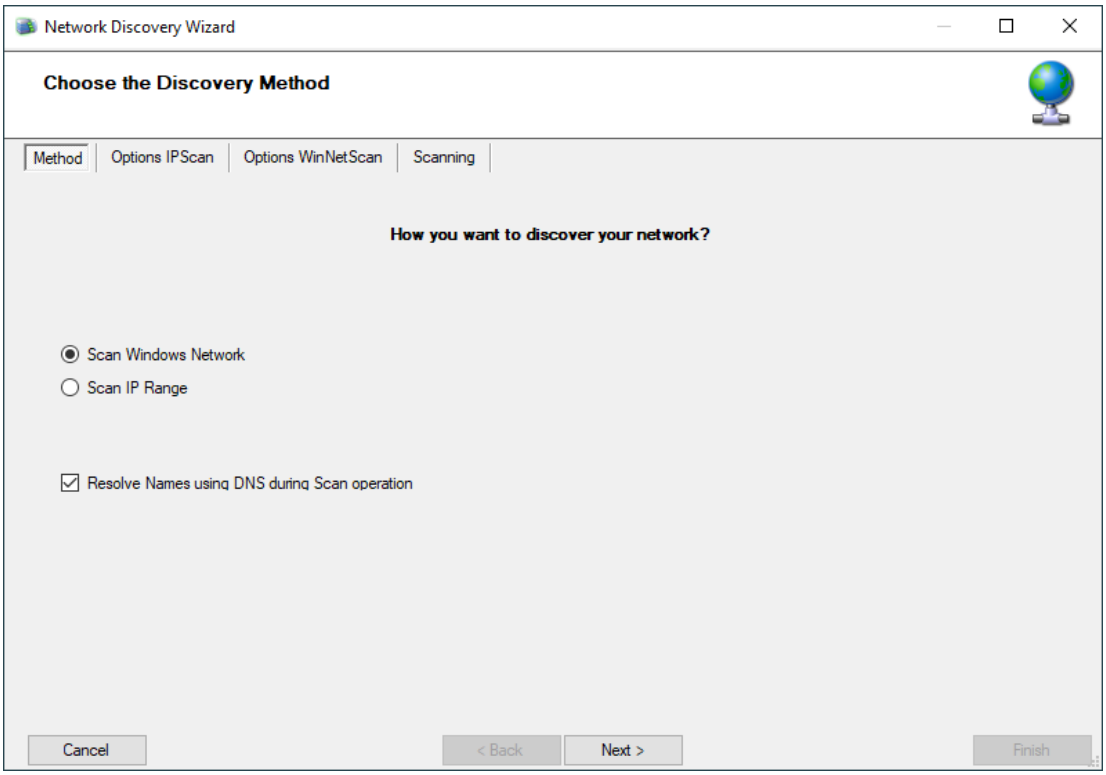
If required, configure a custom message that is send to the server after connect.

Expect this Message after Connect

If required, configure a custom message that is expected by the sender when the server response to our custom message.

Network Discovery

Opens up a Wizard that will help you discover devices in your local network. Once the wizard has scanned your network, it will show Windows compatible devices it has found. Please note that this will require Windows Management Instrumentation (WMI) access to the remote machines which may be disabled in Windows Firewalls by default.



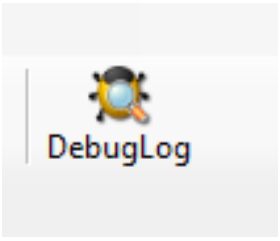
- Network Discovery - Choose the discovery Method*

Kill Service

When stopping a service, and it does not shutdown in the time period, you can use this function to forcefully stop the service. The service process will be killed if possible.

DebugLog

The **DebugLog** Button will be available if Debug Logging is enabled in your Debug Options



- DebugLog*

When clicked, a new Logviewer window will be opened. The Debug Logviewer can load, parse, and analyze debug log-files from the service.

- Logviewer (Debug)*

Debugfile

Will automatically be set to your configured debug file. You can also choose other saved debug-files for analysis.

Load

When Load is clicked, the Logviewer will load lines as configured in the initial debug-lines field. When loading all log-lines on a large debug log-file, this may take a while. While the Load button is grayed out, the Logviewer will continue to read data from the debug log as it is being written.

Stop

Stop continuous loading of the debug log.

Reset

Will reset all loaded log-lines from memory and clear the debug data-grid.

Init Debuglines

The amount of log-lines you want to read the first time.

Show debug messages for

Once the debug-log is processed, the Logviewer will automatically add filters for objects like services, rulesets, rules, and actions. You can use this select box to filter by them.

Filtering (bottom bar)

At the bottom of the Logviewer window, you can filter the debug-log for Thread (ID), Priority, internal Facility, and Functions. You can also filter for words or word sequences. The view will automatically be refreshed once you changed a filter.

Using File based configuration

Working with File based Configurations

Support for running the Service from file based configuration may be interesting for environments where you want to minimize registry access to a minimum or you want to manually edit the configuration without using the configuration client every time.

The Adiscon Configuration format is quiet simple. In the following description, all the configuration options will be explained in detail.

Adiscon Configuration format explained

Our configuration format is something between JSON and XML but hold at a very simple level.

Variables

All variables start with a dollar ($). Name and Value of a variable are separated by the FIRST space character. Everything else behind the first space will be considered as the Value. A line feed terminates the value. If your configuration value contains line feeds, you have to replace them with “\n” or “\r\n”. A single backslash can be used to escape brackets ({ and }).

Comments

All lines starting with a sharp (#) at the beginning will be ignored.

File Includes

Sample

```
includeconfig my-subconfigfiles-*.cfg*
```

The includeconfig statement will include either a single file or many files based on a filename pattern. In this sample all Files starting with “my-subconfigfiles-” and ending with “.cfg” will be included into the configuration. It is possible to create your own custom file structure with includes. The configuration client will be able to load and show your custom file structure, however it will not be able to maintain (save) it. We support a maximum include depth of up to 10 levels when using the includeconfig statement.

General Options

Sample

```
general(name="[name]") {
 $nOption 1
 ...
}
```

All options between the brackets will be loaded as variables into the general configuration object. The name attribute field specifies the general configuration block name. The brackets start and end an object block.

Services

All possible configuration parameters are named within the detailed services documentation.

Sample Service configuration:

```
input(type="[ID]" name="[name]") {
 $var1 Value1
 $var2 Value2
 ...
}
```

The brackets start and end a service block. All variables between the brackets will be loaded into the service configuration. The name attribute specifies the service display name. The type attribute contains the service type ID. It can be one of the following types:

```
1       = Syslog
2       = Heartbeat
3       = EventLog Monitor V1 (Win 2000 / XP / 2003 )
4       = SNMP Trap Listener
5       = File Monitor
8       = Ping Probe
9       = Port Probe
10      = NTService Monitor
11      = Diskspace Monitor
12      = Database Monitor
13      = Serialport Monitor
14      = CPU Monitor
16      = MonitorWare Echo Request
```

```
17      = SMTP Probe
18      = FTP Probe
19      = POP3 Probe
20      = IMAP Probe
21      = IMAP Probe
22      = NNTP Probe
23      = EventLog Monitor V2 (Win VISTA/7/2008 or higher)
24      = SMTP Listener
25      = SNMP Monitor
26      = RELP Listener
27      = Passive Syslog Listener
1999998 = MonitorWare Echo Reply
1999999 = SETP Listener
```

RuleSets

All possible configuration parameters are named within the detailed actions documentation.

Sample

```
ruleset(name="[name]" expanded="[on/off]") {
 rule(name="[name]" expanded="[on/off]" actionexpanded="[on/off]"
 ThreatNotFoundFilters="[on/off]" GlobalCondProperty="[on/off]"
 GlobalCondPropertyString="" ProcessRuleMode="[0/1/2]"
ProcessRuleDate="[uxtimestamp") {
       action(type="[ID]" name="[name]") {
               $var1 Value1
               $var2 Value2
               ...
       }
       filter(nTabSelection="0") {
               $nOperationType AND
               $PropertyType NOTNEEDED
               $PropertyValueType NOTNEEDED
               $CompareOperation EQUAL
               $nOptionalValue 0
               $nSaveIntoProperty 0
               $szSaveIntoPropertyName FilterMatch
       }
    }
}
```

The brackets start and end a ruleset block. The attributes of a Ruleset are self-explainable. Within a RuleSet, you can have Rules. The attributes of Rules are also self-explainable and partially Global Conditions that are equal to the options found in the Filter dialog. Within a Rule you can one Basefilter. This Basefilter again can have child filters it and these child filters can have child filters again. All “expanded” settings are optional and only important for the client treeview.

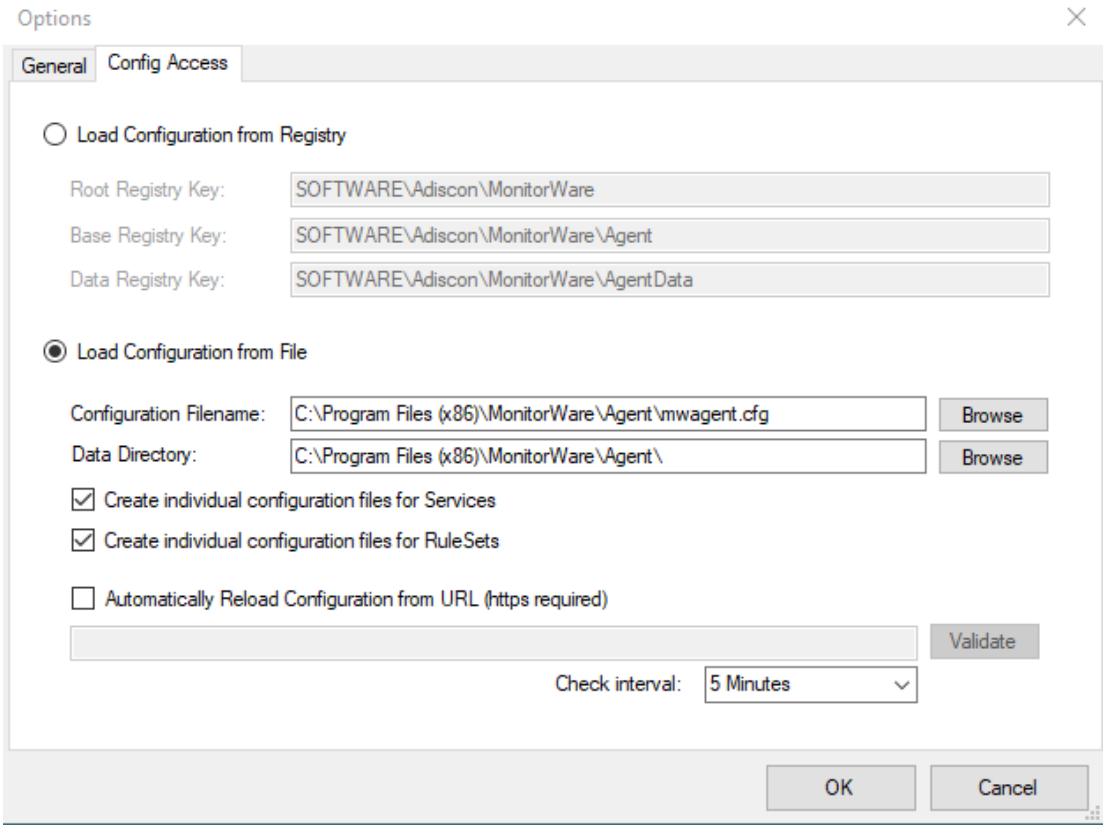
Within a Rule you can have Actions. The brackets start and end an action block. All variables in an action block between the brackets will be loaded into the action configuration. The name attribute specifies the service display name. The type attribute contains the action type ID. It can be one of the following types:

```
1000 = ODBC Database
1001 = Send Syslog
1008 = Net Send
1009 = Start Program
1011 = Send SETP
1012 = Set Property
1013 = Set Status
1014 = Call RuleSet
1015 = Post Process
1016 = Play Sound
1017 = Send to Communication Port
```

```
1021 = Send SNMP
1022 = Control NT Service
1023 = Compute Status Variable
1024 = HTTP Request
1025 = OleDB Database
1026 = Resolve Hostname
1027 = Send RELP
1028 = Send MS Queue
1029 = Normalize Event
1030 = Syslog Queue
```

How to enable file based configuration?

To switch from registry to file configuration mode, all you need to do is to go the “Config Access” tab in the Configuration “Client Options” and switch from “Load Configuration from Registry” to “Load Configuration from File” mode. Once you accept the change, the Client will ask you if you want to export the current loaded configuration into the file. Hit YES if you want to do so and NO if already have an existing configuration file. The configuration client will reload itself automatically after this.



- Client Options Configure File Based Configuration*

Create individual configuration files for Services

When enabled, the configuration client will create separated configuration files for each configured service. The main configuration file will then use the includeconfig statement to include all these configuration files by using a pattern. When deleting a service, its configuration file will be deleted as well.

Create individual configuration files for RuleSets

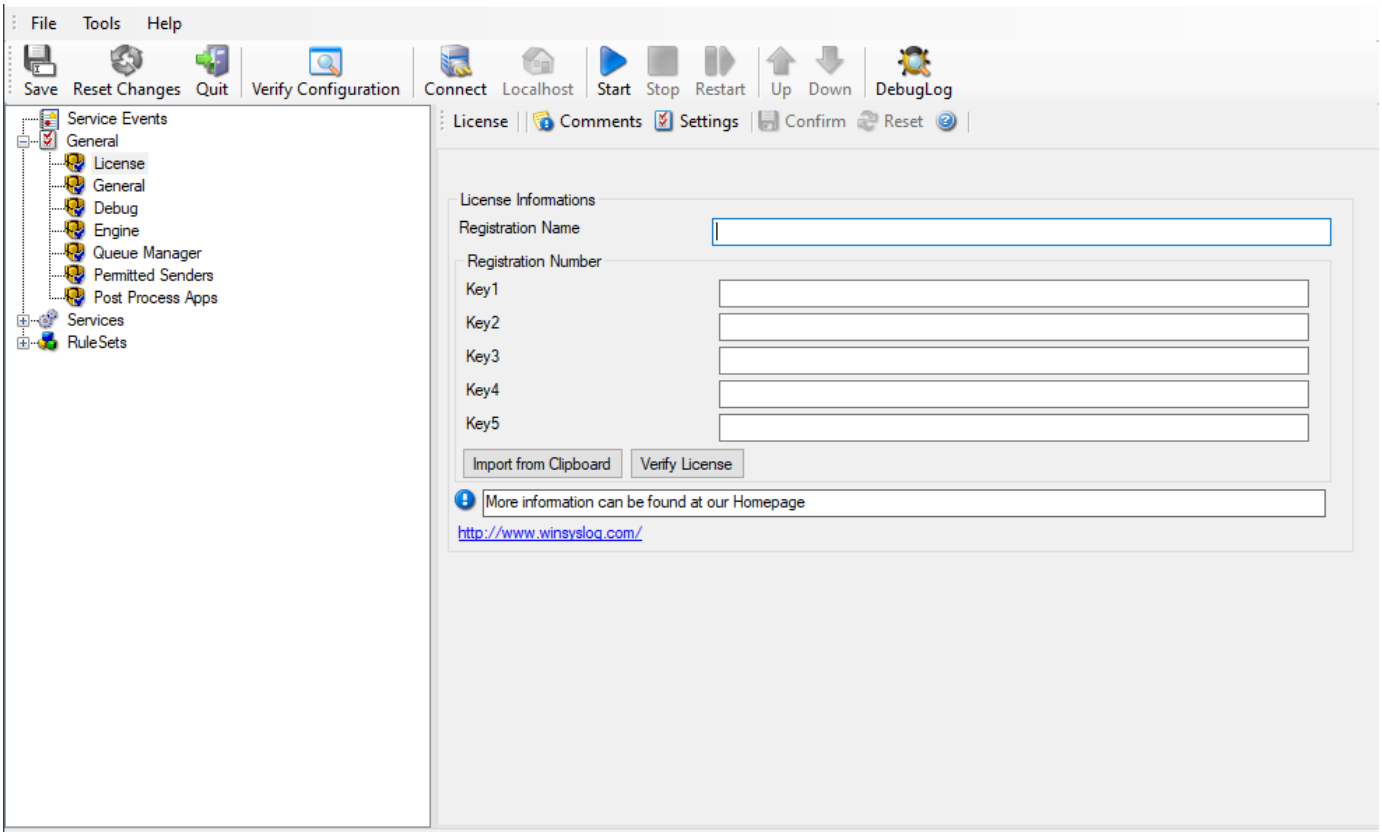
When enabled, the configuration client will create separated configuration files for each configured ruleset. The main configuration file will then use the includeconfig statement to include all these configuration files by using a pattern. When deleting a ruleset, its configuration file will be deleted as well

General Options

In this chapter, you find the general option settings.

License

After the purchase, the licensing information can be entered here.



Registration Name

File Configuration field:

szlicense

Description

The user chooses the registration name. It should correspond to your organization name, e.g. a company called “AA Carpenters, Inc.” should not choose “AA” as registration name. This can easily be mistaken and most probably be rejected by Adiscon for that reason. With the above scenario, we recommend using the full company name “AA Carpenters, Inc.”.

Please note: The registration name is case sensitive. It must be entered exactly as given. Leading and trailing spaces are also part of the registration name, so be sure to enter none.

Registration number

File Configuration field:

nLicenseKey1, nLicenseKey2, nLicenseKey3, nLicenseKey4, nLicenseKey5

Description

Adiscon provides this number. It is valid for a specific registration name. Be sure to enter the correct registration number. Each block of the license key must be filled into one of the key fields. Alternatively, you can use the “Import from Clipboard” button. The client detects invalid registration numbers and reports the corresponding error.

Import from Clipboard

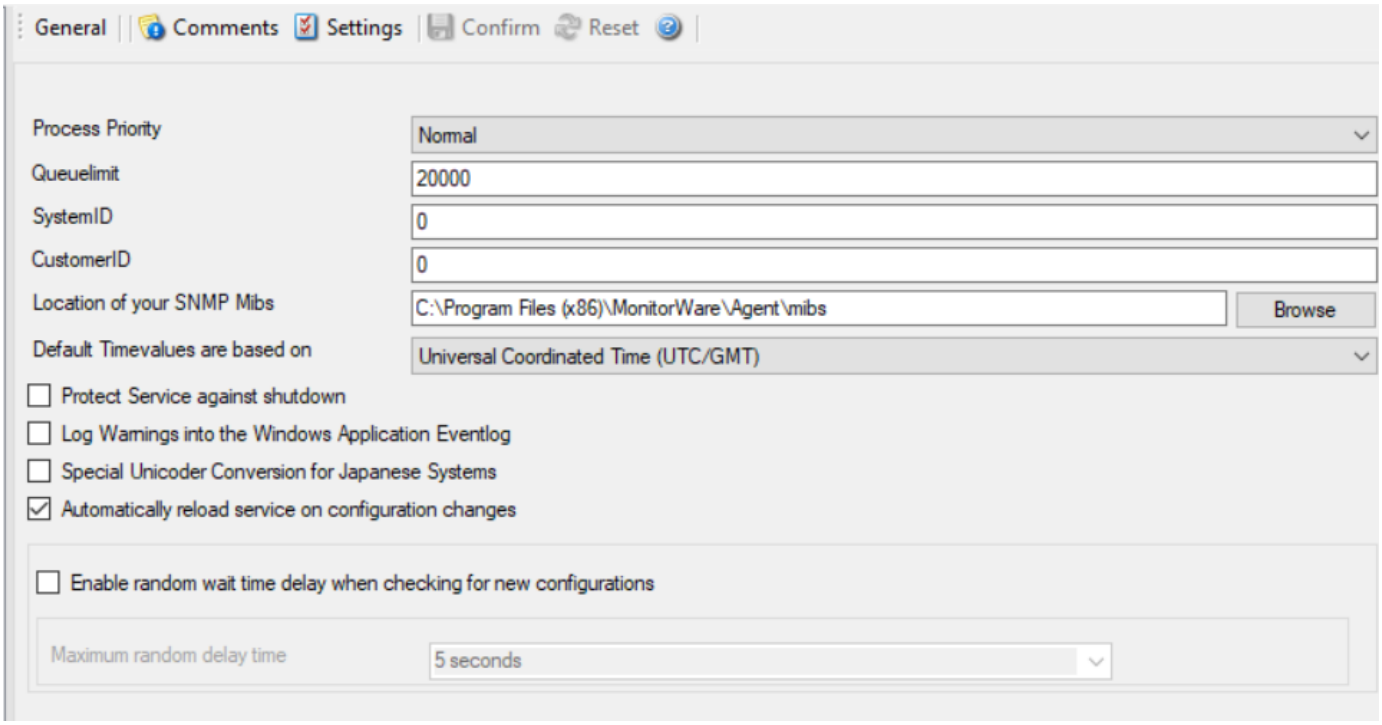
If the key has been copied to the clipboard it can be imported with this button.

Verify License

Here it can be verified if the license is valid.

General

The General Options available on this form are explained below:



Process Priority

File Configuration field:

nProcessPriority

Description

Configurable Process Priority to fine-tune application behavior.

Queuelimit

File Configuration field:

nQueueLimit

Description

The applications keeps an in-memory buffer where events received but not yet processed are stored. This allows the product to handle large message bursts. During such burst, the event is received and placed in the in-memory queue. The processing of the queue (via rulesets) itself is de-coupled from the process of receiving. During traffic bursts, the queue size increases, causing additional memory to be allocated. At the end of the burst, the queue size decreases and the memory is freed again.

Using the queue limit, you can limit that maximum number of events that can be in the queue at any given time. Once the limit is reached, no further enqueuing is possible. In this case, an old event must first be processed. In such situations, incoming events might be lost (depending on the rate they come in). A high value for the queue size limit (e.g. 200,000) is recommended, because of the risk of message loss.

It is also possible to place no limit on the queue. Use the value zero (0) for this case. In this case, the queue size is only limited by virtual memory available. However, we do not recommend this configuration as it might cause the product to use up all available system memory, which in turn could lead to a system failure.

SystemID

File Configuration field:

nSystemID

Description

SystemID is of type integer to be used by our customer. In addition, it is user configurable.

CustomerID

File Configuration field:

nCustomerID

Description

CustomerID is of type integer provided for customer ease. For example if someone monitors his customer's server, he can put in different CustomerIDs into each of the clients. Let us say someone monitors servers A and B. A has 5 servers all of them with CustomerID = 1 and B has 2 servers all of them with CustomerID = 2. Both A and B happen to have a server named "SERVER". Together with the customerID, these machines are now uniquely identifiable. This is user configurable.

Location of your SNMP MIBs

File Configuration field:

szMIBSPath

Description

Click the Browse button to search for your MIBs location or enter the path manually. The Client and Service will read all files from this directory automatically on startup.

Default Timevalues are based on

File Configuration field:

nTimeMode

Description

The general options of each product (EventReporter, MonitorWare Agent and WinSyslog) contain a setting for the "Default Timevalues are based on". This setting can be set to Localtime and UTC (Universal Coordinated Time) which is default. This setting has an effect on:

- Send Email Action: The date in the email header is affected
- Start Program Action: Time parameters in the command line are affected
- Write File Action: Time properties in the file name are affected
- Filter Engine: If you filter by weekday or time fields, localtime does affect the filter result

For information about "How can I get localtime output" please see default timevalues setting in EventReporter/MonitorWare Agent/WinSyslog explained.

Protect Service against shutdown

File Configuration field:

nProtectAgainstShutdown

Description

When enabled, the Agent will not stop processing the internal queue when it is stopped. **Please note that it will remain in the stopping state then.**

Log Warnings into the Windows Application Eventlog

File Configuration field:

nEnableEventlogWarnings

Description

The Service will also log Warnings into the Windows Application Eventlog, and so be more verbose for troubleshooting. Default is disabled.

Special Unicoder Conversion for Japanese Systems

File Configuration field:

nJapanStringHandling

Description

This is a historical option for older multibyte systems from the time when UTF8 was not known yet. If enabled, whenever text is being converted from 16 Bit wide character to 8 Bit character, the conversion is done with bit masking in order to avoid broken encoding. **For today modern systems, we do NOT recommend to enable this option.**

Automatically reload service on configuration changes

File Configuration field:

nEnableAutoConfigReload

Description

When enabled (default), the service will detect configuration changes and reload its core automatically. This feature only works if the latest Client Application is used for configuration. It will also work if you are using the file based configuration method and update the configuration file. It will not work if you are using the service in console mode unless you send any input to the console.

Enable random wait time delay when checking for new configurations

File Configuration field:

bAutoReloadRandomDelay

Description

When enabled, a random delay (with the configured maximum) will be added between new configuration checks.

Maximum random delay time

File Configuration field:

nAutoReloadDelayTime

Description

The maximum for this random delay is 24 hours. The random delay has no affect on the service control anymore.

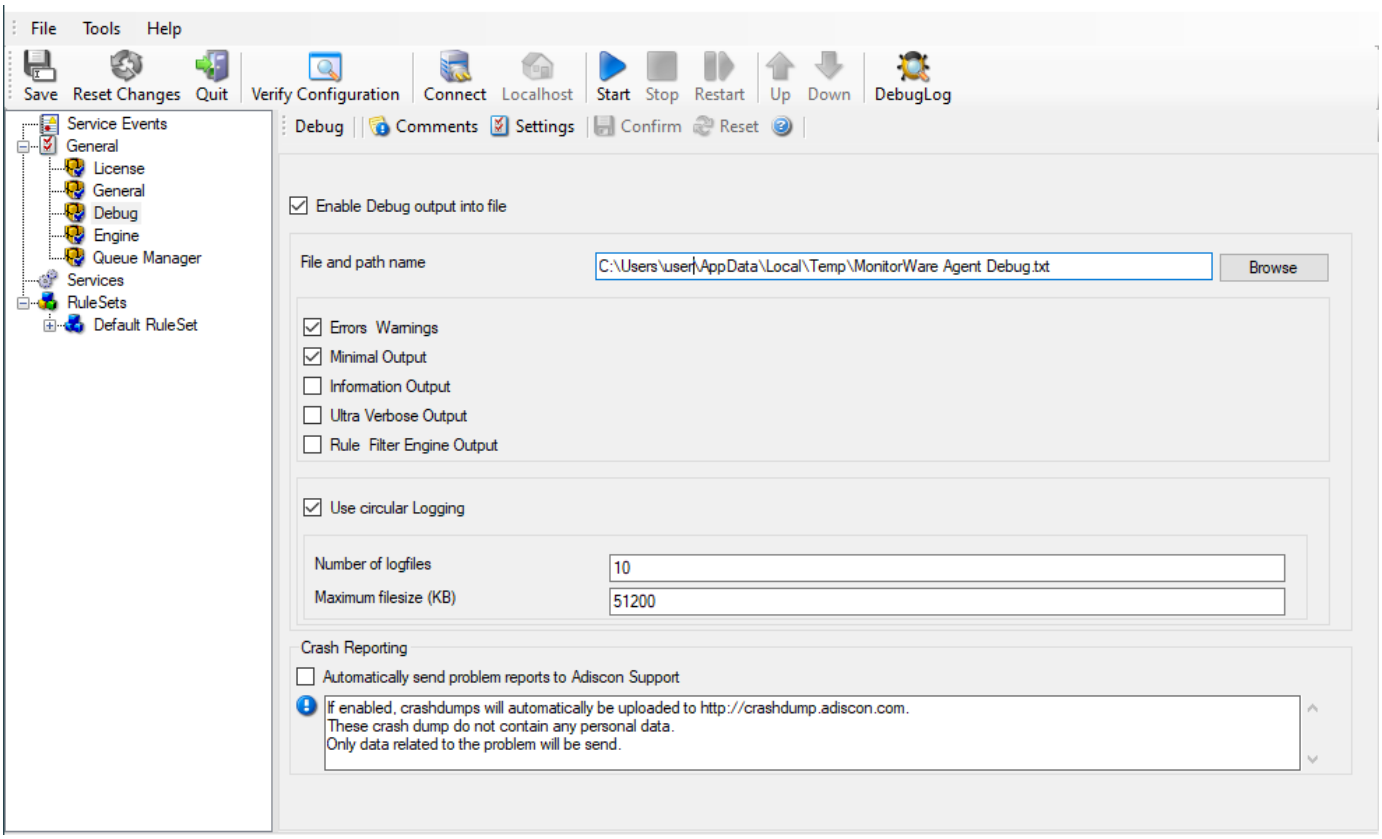
Debug

This tab can be used to debug rule bases. Especially with complex bases, it might be necessary to learn what application is internally doing while it is processing them. With the debug log, the service tells you some of these internal workings.

Other than rule basis testing, the debug log is also helpful when contacting Adiscon support. An Adiscon support engineer might ask you to set the debug log to a specific level while doing troubleshooting.

Note

Debug logging requires considerable system resources. The higher the log level, the more resources are needed. However, even the lowest level considerable slows down the service. As such, we highly recommend turning debug logging off for normal operations.



Enable Debug output into file

File Configuration field:

nEnableDebugOutput

Description

If checked, the debug log is enabled and written as the service operates. If unchecked, no debug log is written. For performance reasons, it is highly recommended that this box is unchecked during normal operations.

File and path name

File Configuration field:

szDebugFileName

Description

The full name of the log files to be written. Please be sure to specify a full path name including the drive letter. If just the file and/or path name is specified, that information is local to the service default directory. As this depends on a number of parameters, it might be hard to find the actual log file. So for consistency purposes, be sure to specify a fully qualified file name including the drive.

Note: If the configured directories are missing, they are automatically created by application i.e. the folder specified in “File and Path Name”.

Debug Levels

File Configuration field:

nDebugErrors, nDebugMini, nDebugInternal, nDebugUltra, nDebugRuleEngine

Description

These checkboxes control the amount of debug information being written. We highly recommend only selecting “Errors & Warnings” as well as “Minimum Debug Output” unless otherwise instructed by Adiscon support.

Use circular Logging

File Configuration field:

nCircularLogging

Description

Support for circular debug logging has been added as the debuglog can increase and increase over time. This will avoid an accidental overload of the hard disk. Of course you can also customize the amount of files used and their size or disable this feature.

Automatically send problem reports to Adiscon Support

File Configuration field:

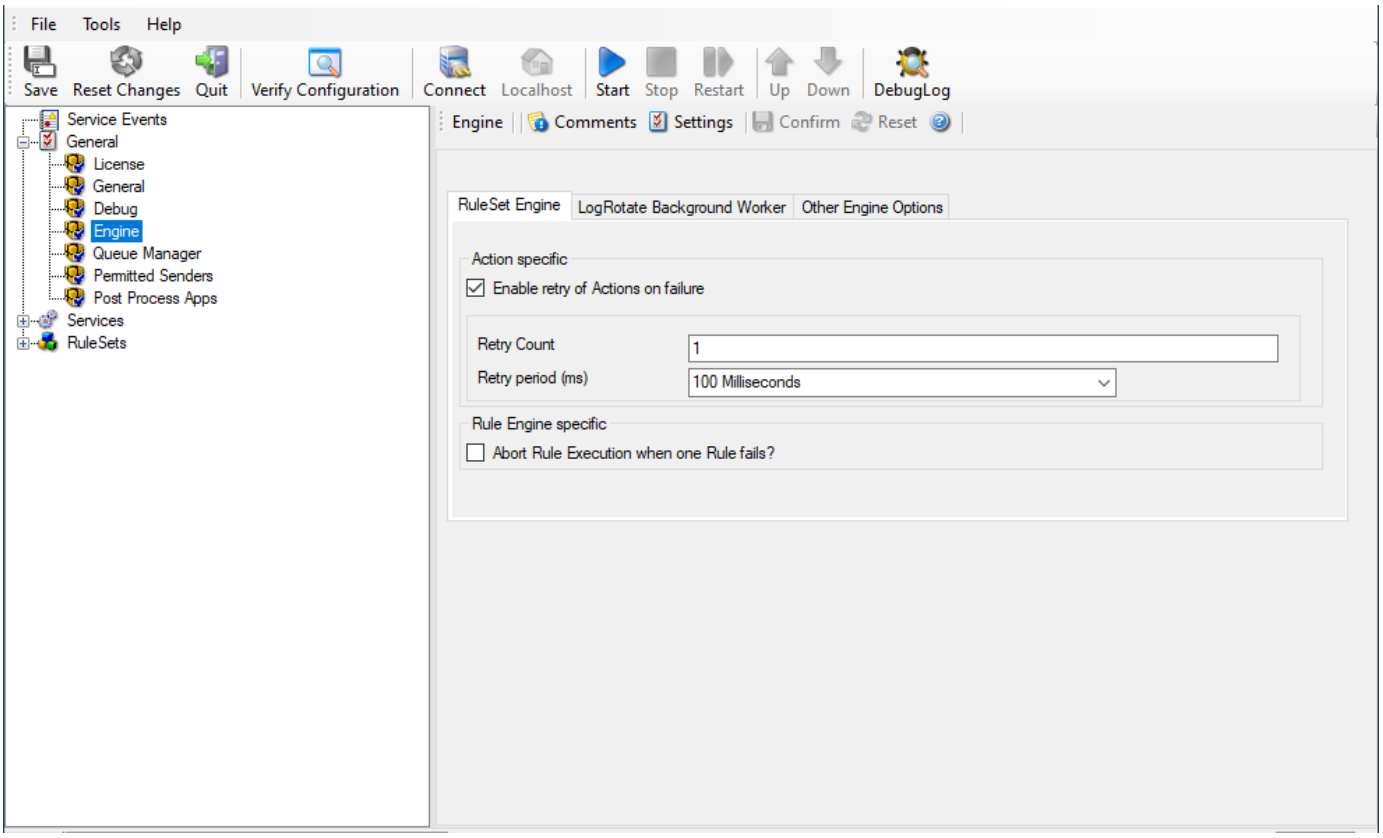
nReportCrash

Description

If enabled, problem reports will automatically be uploaded to `http://crashdump.adiscon.com`. A problem report is generated if the service internally stops working for some unknown reason. The reports are small dumpfiles which do not contain any personal data and will help us find and fix the problem. Also the dumpfiles are very small and do not exceed 256 Kbyte. In most cases only 32Kbyte data is send.

Engine

The Engine specific Options are explained below:



- RuleSet Engine Tab*

Action specific

Enable retry of Actions on failure

File Configuration field:

nEnableRetry

Description

If enabled, the Agent retries Actions on failure (until the retry counter is reached). Note that the Event error 114 will only be written if the last retry failed, previous error's will only be logged in the debug log (with the error facility). Note that you can customize the Retry Count and the Retry Period in ms as well.

Rule Engine specific

Abort Rule Execution when one Rule fails?

File Configuration field:

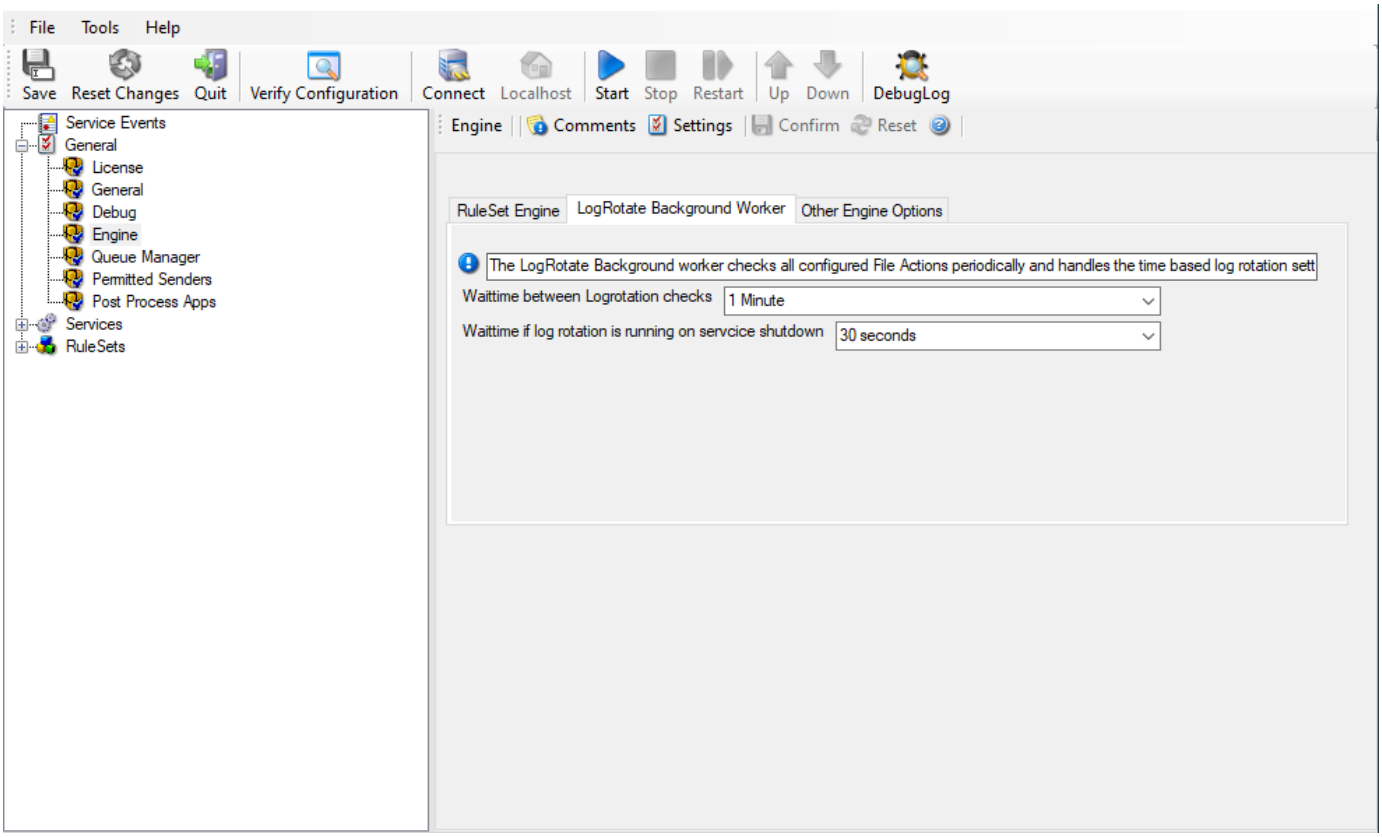
bAbortRuleOnFailure

Description

If checked, and an action fails, the execution will be aborted. If unchecked, and an action fails, simply the next action in this rule will be executed.

LogRotate Background Worker

The LogRotate Background worker checks all configured File Actions periodically and handles the time based log rotation settings, if enabled.



- LogRotate Background Worker Tab*

Wait time between Logrotation checks

File Configuration field:

nLogRotateWorkerSleepTime

Description

Defines how often the logrotate background worker thread checks all configured actions to see if any logfiles need to be rotated based on time related rotate conditions.

Wait time if log rotation is running on service shutdown

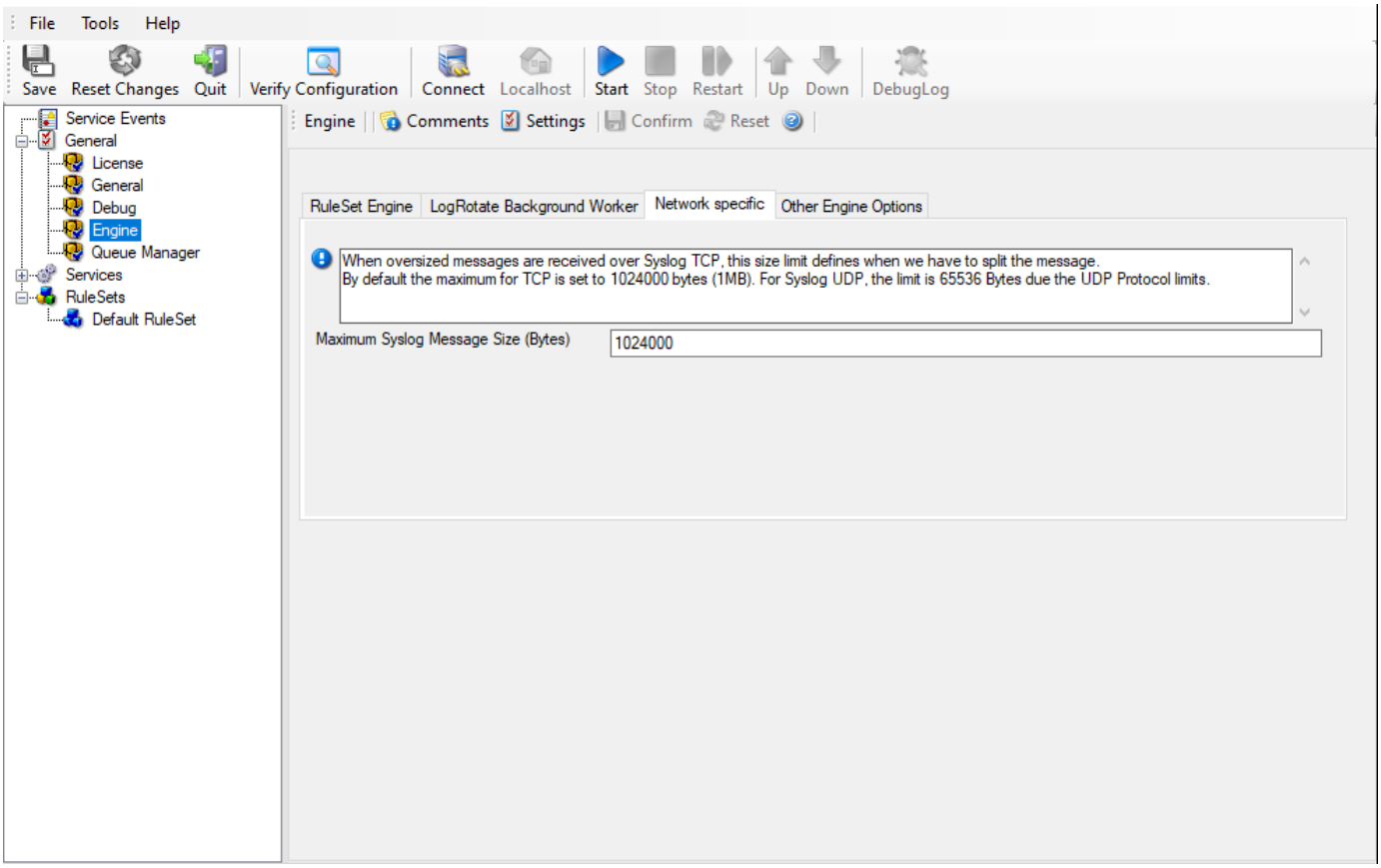
File Configuration field:

nLogRotateWorkerStopWaitTimeout

Description

When service is being shutdown, this defines how much time the logrotate background worker thread has left to finish its log rotations before a forceful termination.

Network specific Options



- Network specific Options Tab*

Maximum Syslog Message Size (Bytes)

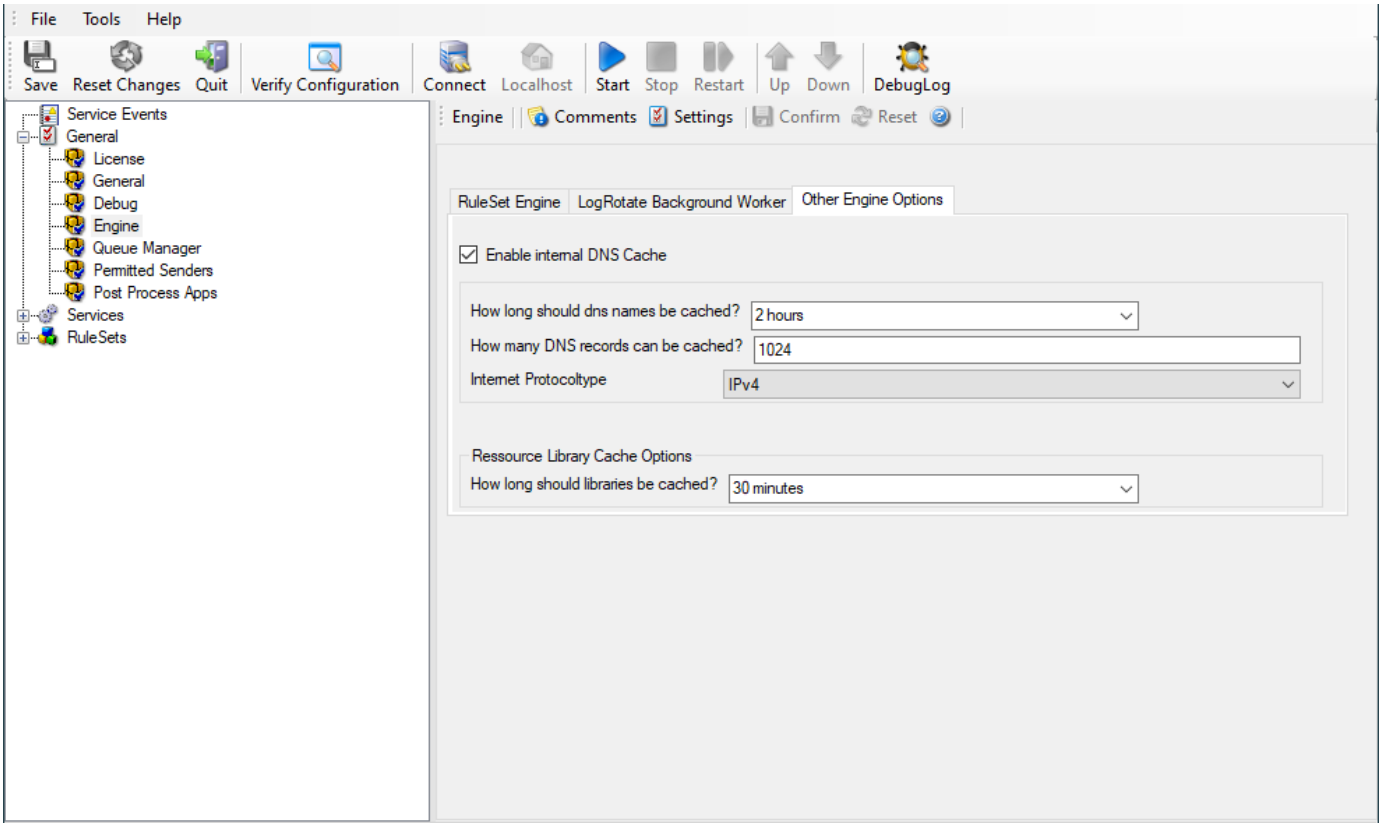
File Configuration field:

nSyslogMaxMessageSize

Description:

Configurable message size limit for Syslog TCP messages. The default is 1MB which is far more as defined in Syslog RFC's. If a syslog message exceeds the size limit, it will be split into multiple messages.

Other Engine Options



- Other Engine Options Tab*

Enable internal DNS Cache

File Configuration field:

nEnableDNSCache

Description

The DNS cache is used for reverse DNS lookups. A reverse lookup is used to translate an IP address into a computer name. This can be done via the resolve hostname action. For each lookup, DNS needs to be queried. This operation is somewhat costly (in terms of performance). Thus, lookup results are cached. Whenever a lookup needs to be performed, the system first checks if the result is already in the local cache. Only if not, the actual DNS query is performed and the result then stored to the cache. This greatly speeds up reverse host name lookups.

However, computer names and IP addresses can change. If they do, the owner updates DNS to reflect the change. If we would cache entries forever, the new name would never be known (because the entry would be in the cache and thus no DNS lookup would be done). To reduce this problem, cache records expire. Once expired, the record is considered to be non-existing in the cache and thus a new lookup is done.

Also, cache records take up system memory. If you have a very large number of senders who you need to resolve, more memory than you would like could be allocated to the cache. To solve this issue, a limit on the maximum number of cache records can be set. If that limit is hit, no new cache record is allocated. Instead, the least recently used record is overwritten with the newly requested one.

How long should DNS names be cached?

File Configuration field:

nDNSCacheTime

Description

This specifies the expiration time for cache records. Do not set it too high, as that could cause problems with changing names. A too low-limit results in more frequent DNS lookups. As a rule of thumb, the more static your IP-to-hostname configuration is, the higher the expiration timeout can be. We suggest, though, not to use a timeout of more than 24 to 48 hours.

How many DNS records can be cached?

File Configuration field:

nDNSCacheLimit

Description

This is the maximum number of DNS records that can be cached. The system allocates only as many memory, as there are records required. So if you have a high limit but only few sending host names to resolve, the cache will remain small. However, if you have a very large number of host names to resolve, it might be useful to place an upper limit on the cache size. But this comes at the cost of more frequent DNS queries. You can calculate about 1 to 2 KBytes per cache record.

Internet Protocoltype

File Configuration field:

nDNSInetProtocol

Description

Select if you wish to prefer IPv4 or IPv6 addresses for name resolution. Note that this only has an effect on names which return both, IPv4 and IPv6 addresses.

Resource Library Cache Options

How long should libraries be cached?

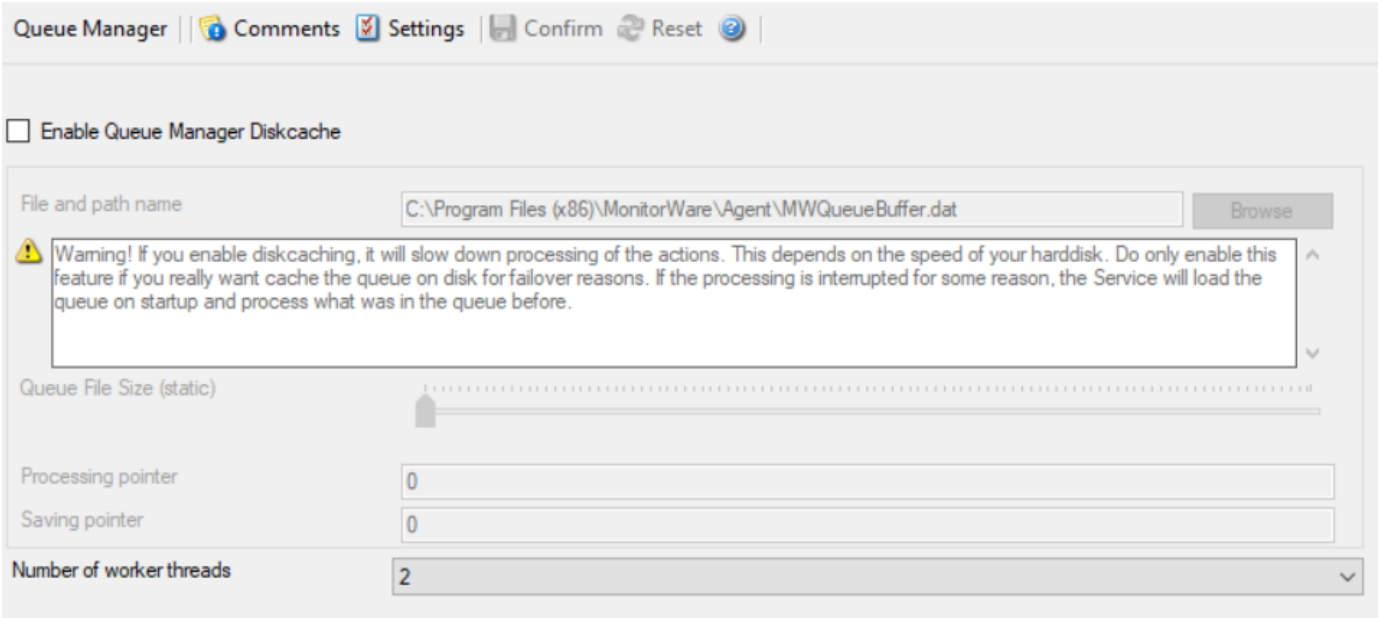
File Configuration field:

nLibCacheTimeOut

Description

This feature will be mainly useful for EventLog Monitor. For events with the same recurring event sources, this will be a great performance enhancement. The cache will also work for remote system libraries (requires administrative default shares). All libraries will be cached for 30 minutes by default.

Queue Manager



- Queue Manager*

Enable Queue Manager DiskCache

This feature enables the Agent to cache items in its internal queue on disk using a fixed data file.

Warning

Only use this feature if you really need to!

Depending on the speed of your hard disks, it will slow down processing of the actions, in worst case if the machine cannot handle the IO load, the Queue will become full sooner or later. The DiskCache is an additional feature for customers, who for example want to secure received Syslog messages which have not been processed yet.

The diskcache will not cache infounits from services like EventLog Monitor, as this kind of Service only continues if the actions were successfully. All other information sources like the Syslog server will cache its messages in this file. If the Service or Server crashes for some reason, the queue will be loaded automatically during next startup of the Agent. So messages which were in the queue will not be lost. Only the messages which was currently processed during the crash will be lost.

Enable Queue Manager Diskcache

File Configuration field:

nEnableRingBuffer

Description

Enable the disk based queue manager. Please read the description about the Queue Manager DiskCache first!

File and Pathname

File Configuration field:

szRingBufferFile

Description

As everywhere else, you can define here, where the queue file should be stored.

Queue File Size

File Configuration field:

nRingBufferSize

Description

With this slider, the queue size can be set from 1 MB to 2048 MB.

Processing pointer

File Configuration field:

nProcessingLow

Description

Points to the current processing position within the queue file.

Saving pointer

File Configuration field:

nSavingLow

Description

Points to the last processed position within the queue file.

Queue Manager specific

Number of worker threads

File Configuration field:

nWorkerThreads

Description

Defines the number of worker background threads that the core engine uses to process its queue.

Services

Services gather events data. For example, the Syslog server service accepts incoming Syslog messages and the Event Log Monitor extracts Windows Event Log data. There can be unlimited multiple services. Depending on the service type, there can also be multiple instances running, each one with different settings.

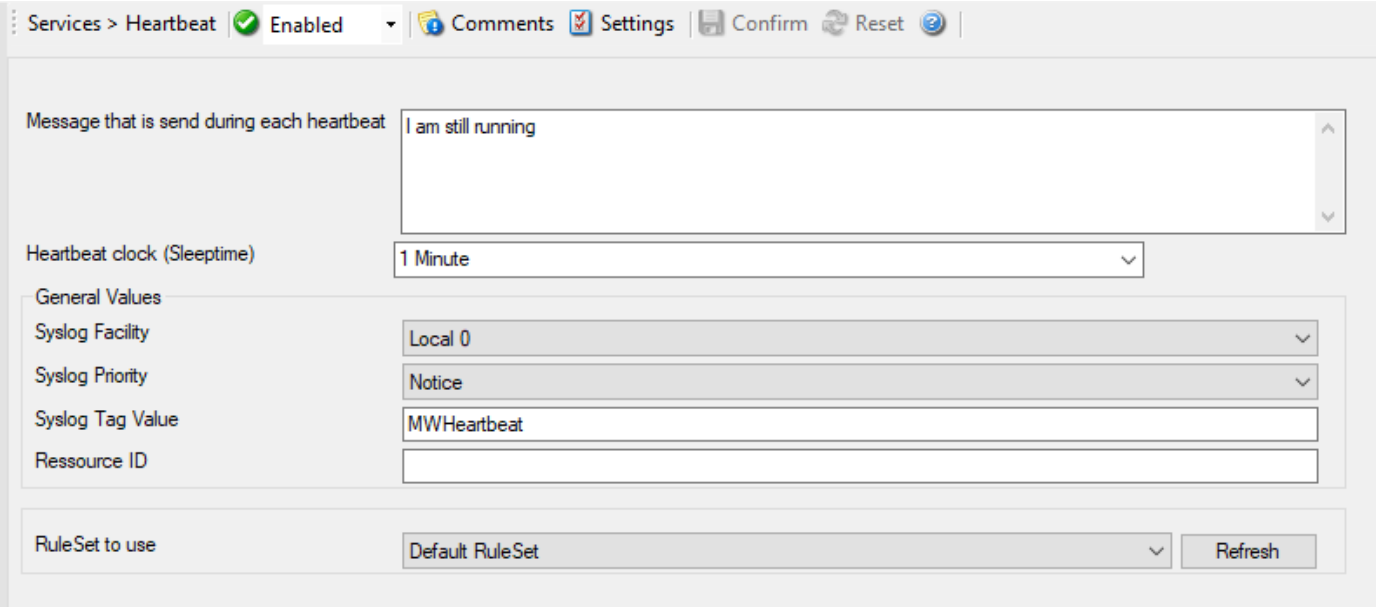
You must define at least one service, otherwise the product does not gather event data and hence does not perform any useful work at all. Sometimes, services are mistaken with service defaults those are pre-existing in the tree view. Service defaults are just the templates that carry the default properties assigned to a service, when one of the respective type is to be created. Service defaults are NOT executed and thus cannot gather any data.

Added a test mode for Services, currently EventLog Monitor V1 & V2 and File Monitor are supported. When enabling the testmode for a certain service, it will process it's Events/Files over and over again. So only use this setting for testing purposes.

Basic Services

Heartbeat

The heartbeat process can be used to continuously check if everything is running well. It generates an information unit every specified time interval. That information unit can be forward to a different system. If it does not receive additional packets within the configured interval, it can be assumed that the sender is either in trouble or already stopped running.



- Service - Heartbeat*

Message that is send during each heartbeat

File Configuration field:

szMessage

Description:

This is the message that is used as text inside the information unit. Use whatever value is appropriate. The message text does not have any special meaning, so use whatever value you seem fit.

Heartbeat clock (Sleeptime)

File Configuration field:

nSleepTime

Description:

This is the interval, in milliseconds, that the heartbeat service generates information units in. Please note that the receiving side should be tolerant. The interval specified here is the minimum time between packets. Under heavy load, the interval might be slightly longer. It is good practice to allow twice this interval before the service is considered suspect by the system monitoring the services health.

General Values (Common settings for most services)

Syslog Facility

File Configuration field:

nSyslogFacility

Description:

The syslog facility to be assigned to events created by this service. Most useful if the message is to forward to a Syslog server.

Syslog Priority

File Configuration field:

nSyslogPriority

Description:

The Syslog priority to be assigned to events created by this service. Most useful if the message is to forward to a Syslog server.

Syslog Tag Value

File Configuration field:

szSyslogTagValue

Description:

The Syslog tag value to be assigned to events created by this service. Most useful if the message is to forward to a Syslog server.

Resource ID

File Configuration field:

szResource

Description:

The resource id to be assigned to events created by this service. Most useful if the message is to forward to a Syslog server.

RuleSet to Use

File Configuration field:

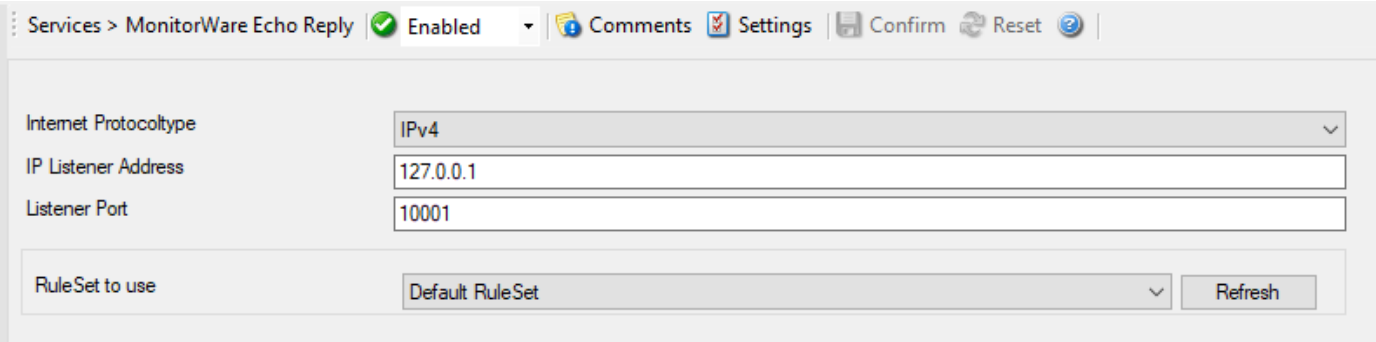
szRuleSetName

Description:

Name of the ruleset to be used for this service. The RuleSet name must be a valid RuleSet.

MonitorWare Echo Reply

The Echo Reply service is used on each of the installed EventReporter/ MonitorWare Agent. A central agent running the MonitorWare Agent is using the echo request and instructs to poll each of the other EventReporter/MonitorWare Agent services. When the request is not carried out successfully, an alert is generated. The MonitorWare echo protocol ensures that always a fresh probe of the remote EventReporter/MonitorWare Agent Service is done.



- Service - MonitorWare Echo Reply*

Internet Protocoltype

File Configuration field:

nInetType

Description:

Select the desired protocol type. IPv4 and IPv6 are available. The IPv6 protocol needs to be properly installed in order to be used. Note that one Service can only handle IPv4 or IPv6, so if you want to use both protocols, you will need to create two separate services.

IP Listener Address

File Configuration field:

szMyIPAddress

Description:

The MonitorWare Echo Reply service can be bound to a specific IP Address. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address. This feature is useful for multihome environments where you want to run different Syslog Servers on different IP Addresses. Please note that the default IP Address `0.0.0.0` means ANY IP Address.

Listener Port

File Configuration field:

nListenPort

Description:

Specify the listener port here.

RuleSet to Use

File Configuration field:

szRuleSetName

Description:

Name of the ruleset to be used for this service. The RuleSet name must be a valid RuleSet.

file system monitoring

Event Log Monitor V1

This dialog configures the Windows Event Log Monitor service.

This service was initially introduced by Adiscon's EventReporter product. To allow previous EventReporter customers seamless upgrades, there are a number of compatibility settings to support older message formats.

Newer Windows versions come with a considerably changed event logging system. In theory, the Event Log Monitor works with them, too. However, we know of some incompatibilities. For best results, we recommend using the event log monitor v2 service, which was specifically written for Windows Vista and newer. The Event Log Monitor described

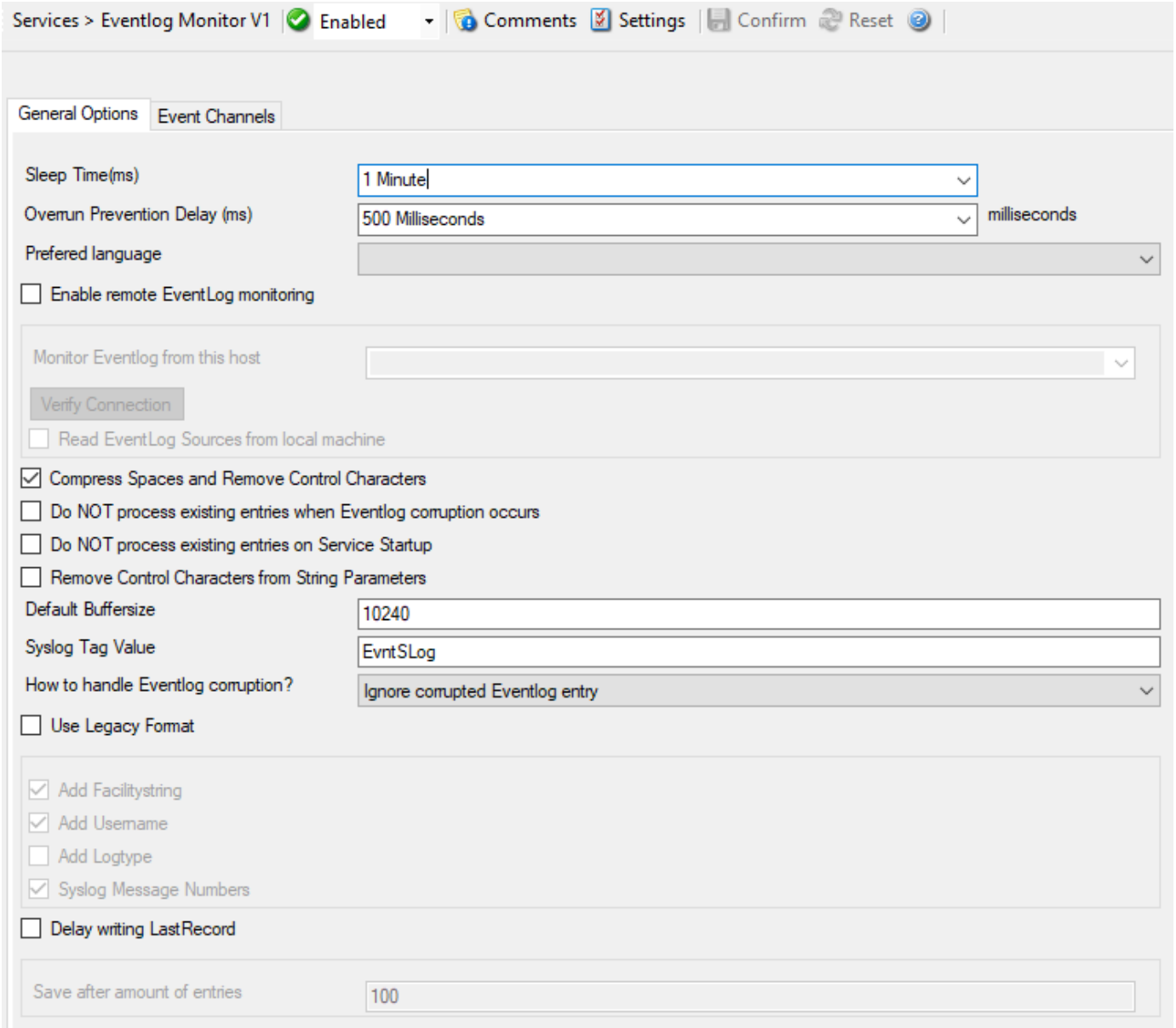
here is applicable for legacy Windows systems, and XP (where the new event logging system is not available). The Client will automatically detect and load available Event Log types during the first startup of the Event Log Monitor.

Event Log Monitor V1

- Windows XP
- Windows 2003

Event Log Monitor V2

- all modern Windows versions (Windows 10, 11, Server 2016, 2019, 2022, and newer).



- Service - Event Log Monitor V1*

General Options Tab

Sleep Time(ms)

File Configuration field:

nSleepTime

Description:

The Event Log Monitor periodically checks for new event log entries. The “Sleep Time” parameter specifies how often this happens. This value is in milliseconds.

We suggest a value of 60,000 milliseconds for the “Sleep Time”. With that setting, the Event Log Monitor checks for new events every 60 seconds. Larger periods can be specified for occasionally connected systems or if email delivery with few emails per day is intended.

Very security-aware environments might use a shorter interval. The event log monitor service is specifically designed to limit the burden on the monitored system. As such, resource usage is typically low, even with frequently run event log checks. However, we recommend not running the Event Log Monitor more often than once a second.

Overrun Prevention Delay (ms)

File Configuration field:

nPreventOverrunDelay

Description:

This property allows configuring a delay after generating an event. The time is the delay in milliseconds.

If run at a value of zero, the Event Log Monitor service generates events as fast as the machine permits. We have seen scenarios where routers and receivers are not able to keep up with this rate, resulting in packet loss. In addition, the CPU of the reporting machine is run at 100% - which is not a problem because the service runs at a low priority. However, with even a 1-millisecond delay, there is no noticeable CPU activity even when large bursts of events are forwarded. At one millisecond, the service can still generate 1000 events per second.

The default setting is an overrun protection of five millisecond, which allows roughly 200 events per second. This should be sufficient for even very busy servers.

Preferred language

File Configuration field:

nLanguageLCID

Description:

You can select a preferred language and the Eventlog Monitor will send the message in this language. It will only work if these languages are installed and message libs are available with the preferred language. If this fails, it will automatically fall back to the system default language.

Enable remote Event Log monitoring

File Configuration field:

nEnabledRemote

Description:

If enabled, the Event Log Monitor will read and process the EventLog from a remote machine. Use the verify button to make sure that the network connection is working correctly.

Please make sure that the machine, which you are going to monitor, does have File and Print Services enabled, and is accessible by this machine.

This is important as the Event Log Service will read the message libraries on the remote machine by using the default administrative shares. For this reason, the Service must be configured to run with a user who has administrative privileges/permissions on the local and remote machine. If File and Print services remain disabled, the local message libraries will used automatically instead. Note that you may experience a lot of missing message libraries in this case.

Additionally you have an option to read the Event Log Sources from the local machine. If enabled, the local message libraries will be used instead of the remote machines ones. Sometimes local Event Sources are more reliable or are required for third-party EventLog implementations.

Compress Spaces and Remove Control Characters

File Configuration field:

nCompressControlChars

Description:

This option allows you to control the control character removal and space compression. If checked, control characters (e.g. CR, LF, TAB - non printable characters in general) are removed. Also multiple spaces are compressed to a single one. By default this is checked. We recommend keeping it checked for most cases as it provides better display.

Please note that it should be unchecked if events should be forwarded via email. And it MUST be turned off if double-byte character sets are being processed (e.g. Japanese).

Do NOT process existing entries when Event Log corruption occurs

File Configuration field:

nDoNotProcessLastRecord

Description:

When this option is checked, it prevents from reprocessing of the whole Windows Event Log when it is corrupted or truncated . So EventReporter / MonitorWare Agent do not process all entries again.

Do NOT process existing entries on Service Startup

File Configuration field:

szSyslogTagValue

Description:

When this option is checked, it prevents from reprocessing of the whole Windows Event Log when the EventReporter / MonitorWare Agent service is restarted.

Remove Control Characters from String Parameters

File Configuration field:

nRemoveControlCharsFromParams

Description:

Enable this option to remove control characters like carriage return or line feeds from parameter strings and category names in Windows Events.

Default Buffersize

File Configuration field:

nDefaultBuffer

Description:

The default Buffersize is 10k. This value will be increased or decreased dynamically if necessary. If you want to use third-party applications like NetApp you must increase the Buffersize manually (minimum 65k), because dynamic adjusting is not possible with them.

SyslogTag Value

File Configuration field:

szSyslogTagValue

Description:

The SyslogTag Value determines the SyslogTag that is used when forwarding Events via syslog. This is useful, if you want to determine later, what kind of syslog message this is, perhaps because you log Event Logs and syslog into the same database.

How to handle Eventlog corruption

File Configuration field:

nEventLogCor

- 0 = Retry processing from beginning
- 1 = Ignore corrupted Eventlog entry

- 2 = Clear all events from Eventlog

Description:

Sometimes it can occur that Event Log messages are corrupted and cannot be read correctly. This usually happens if someone tampered with the Event Log or if you are processing the Eventlog for the first time. In cases like this, you can automatically handle the situation with this option. You have the following options:

- Retry processing Event Log from the beginning: in this case the complete Eventlog will be processed again.
- Ignore corrupted Event Log entry (default): the affected Eventlog entry will be ignored and processing will continue.
- Clear all Events from the Event Log: the Event Log will be cleared completely and new Events hopefully don't get corrupted before they are processed.

Use Legacy Format

File Configuration field:

bUseLegacyFormat

Description:

This option enhances compatibility to scripts and products working with previous versions of EventReporter. The legacy format contains all Windows Event Log properties within the message itself.

The new format includes the plain text message only. The additional information fields (like event ID or event source) are part of the XML formatted event data. If the new format is used, we highly recommend sending or storing information in XML format. This is an option in each of the action properties (of those actions that support it – the write to database option for example always stores the fields separated, so there is no specific option to do so).

Legacy format is meant to support existing parser scripts. We encourage you to use the new, XML-bound format for new implementations. Legacy format will be maintained in future releases to support backward compatibility, but it is no longer actively being developed. There are some shortcomings in legacy format, which can lead to issues when building or operating a log parser. These shortcomings are by design. We will not change this in legacy format - the solution is to use the new format. After all, the new format was created in order to address the issues with legacy format.

Add Facility String

File Configuration field:

bAddFacilityString

Description:

If checked, facility identification is prepended to the message text generated. This parameter enhances compatibility with existing Syslog programs and greatly facilitates parsing the generated entries on the Syslog server. We strongly encourage users to use this enhancement.

This setting only applies if the "Use Legacy Format" option is checked. Otherwise, it does not have any meaning and consequently cannot be configured in that case.

Add Username

File Configuration field:

nAddUserName

Description:

If checked, the Windows user that generated the event log entry is transmitted. If unchecked, this information is not forwarded. This is a configurable option for customers who have written parsing scripts for a previous format which did not contain Usernames. This option must also be unchecked if MoniLog is used.

This setting only applies if the "Use Legacy Format" option is checked. Otherwise, it does not have any meaning and consequently cannot be configured in that case.

Add Logtype

File Configuration field:

nAddLogType

Description:

If checked, then log types e.g. system, security etc. is prepended to the generated message.

This setting only applies if the “Use Legacy Format” option is checked. Otherwise, it does not have any meaning and consequently cannot be configured in that case.

Syslog Message Numbers

File Configuration field:

bShowSyslogMsgNbr

Description:

If checked, a continuously advancing message number is prepended to the generated message. This is useful for Syslog delivery to make sure that all messages have been received at the remote server

This setting only applies if the “Use Legacy Format” option is checked. Otherwise, it does not have any meaning and consequently cannot be configured in that case.

Delay writing LastRecord

File Configuration field:

nEnableLastRecordDelay

Description:

Enables the LastRecord writeback delay to the configured properties below.

Save after amount of entries

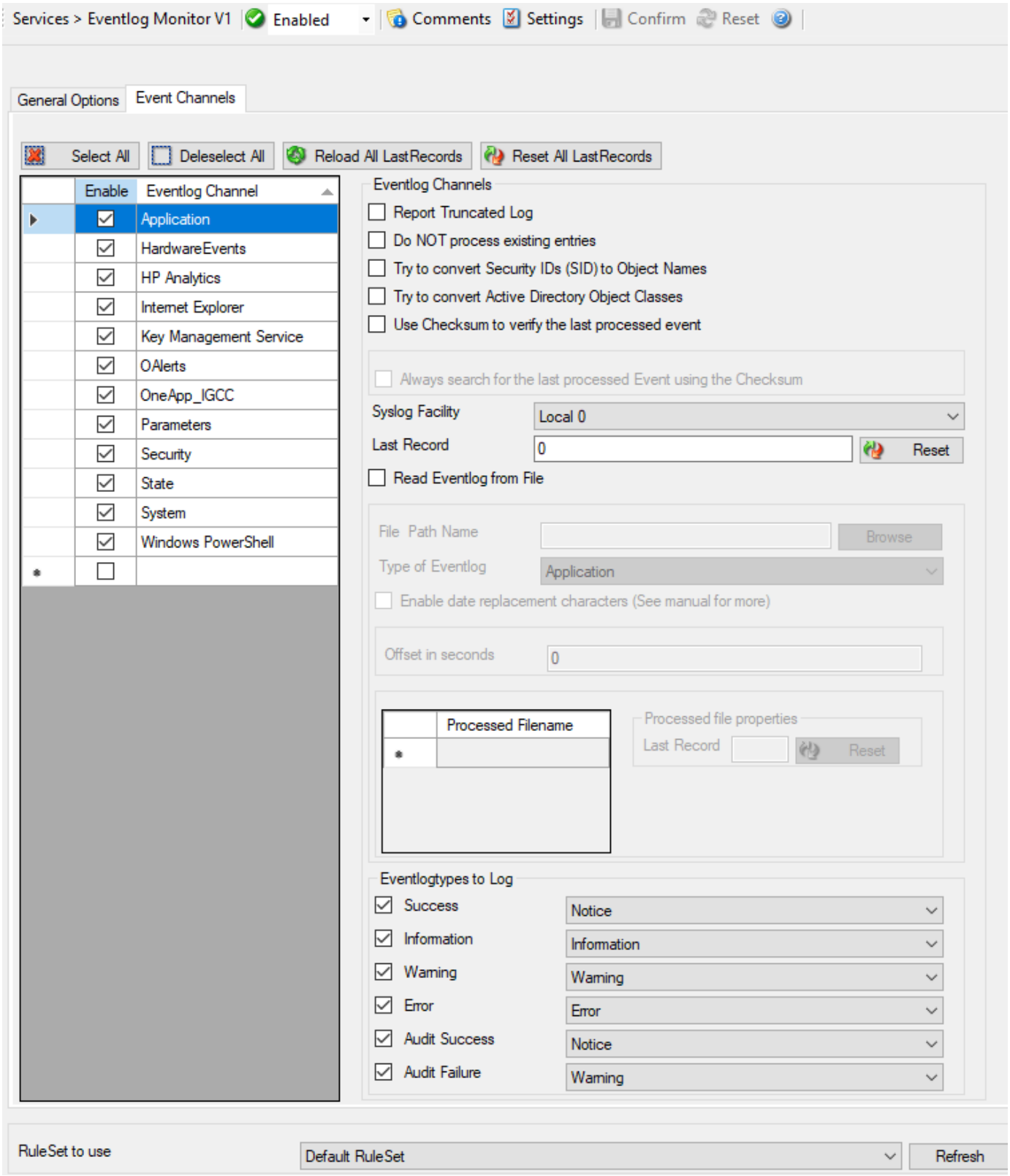
File Configuration field:

nLastRecordDelayCount

Description:

The LastRecord will be written after the amount of processed event log entries that are specified here.

Event Channels Tab



- Service - Event Log Monitor V1 Channels Tab*

Event Log Channels

They are basically a list of the different log types. The corresponding log is only be processed if the respective “Enable” checkbox is checked. The parameters are common to all logs and are explained only once.

Report Truncated Log

File Configuration field:

bReportTruncatedLog

Description:

Windows event logs can be truncated programmatically or via the Windows Event Viewer program. When a log is truncated, all information is erased from it. Any entries not already processed by the service are lost.

The service detects event log truncation. If “Report Truncated Log” is checked, it generates a separate message stating the truncation. This option is most useful in environments where truncation is not expected and as such might be an indication of system compromise.

If you regularly truncate the Windows Event Logs as part of your day-to-day operation, we suggest you turn this option off. In this case, we also recommend using a short sleep period (for example 10,000 which is 10 seconds) to avoid losing log entries.

Do NOT process existing entries

File Configuration field:

nNoExistingEntries

Description:

If you do not want to get a dump of an existing specific Windows Event Log then use this option. When MonitorWare Agent / EventReporter are restarted it will start processing after that last entry and do not look for the previous entries.

Try to convert Security IDs (SID) to Object Names

File Configuration field:

nTryConvSIDtoObj

Description:

With this option you can convert Security ID’s (SIDs) to object names. You can enable this feature in the advanced configuration of each event log type in the Event Log Monitor service. Simple check the “Try to convert Security IDs (SID) to Object Names” option.

Note that only the Security event log has this feature enabled by default. For all other event log types this feature is disabled by default.

Try to convert Active Directory Object Classes

File Configuration field:

nTryConvertDsClasses

Description:

With this option you can convert ActiveDirectory Schema GUID’s from Security Events on Domain Controllers to object names. For example Event 565, which usually has a lot of these Schema GUID’s! The GUID’s are internally cached to speed up EventLog processing operations.

Note that only the Security event log has this feature enabled by default. For all other event log types this feature is disabled by default.

Use Checksum to verify the last processed event

File Configuration field:

nEventUseChecksum

Description:

A checksum of the last processed Event will be stored along with the LastRecord of an event log. This checksum is checked during each iteration. If the checksum does not match , we consider the EventLog has been altered, cleared, or something else happened. In this case the EventLog is being reprocessed from the beginning.

Please note: This option will prevent you from modifying the LastRecord value. If you do, the whole EventLog will be reprocessed! Please note that this behavior is by design and cannot be avoided. So we recommend to use this feature only if you intend to double check if the LastRecord value is valid.

Always search for the last processed Event using this Checksum

File Configuration field:

nEventScanLastEventByChecksum

Description:

Usually, the last processed Event is determined by the LastRecord value. If the Checksum to verify the last processed Event is activated, then the option to always search for the last processed Event using the Checksum is available. When activated, the last processed Event will also be always determined by the Checksum, not the LastRecord value.

Syslog Facility

File Configuration field:

nFacility

Description:

The syslog facility to map information units stemming from this log to. Most useful if the message is to forward to a Syslog daemon.

Last Record

File Configuration field:

nLastRecord

Description:

Windows Event Log records are numbered serially, starting at one. The service records the last record processed. This textbox allows you to override this value.

Use it with caution!

If you would like a complete dump of a specific Windows Event Log, reset the “Last Record” to zero with the reset button. If you missed some events, simply reset it to some lower value than currently set. It is possible to set “Last Record” to a higher value. This suspends event reporting until that record has been created. We strongly discourage to use this feature unless definitely needed.

Read Eventlog from File

File Configuration field:

nReadFromFile

Description:

When enabled, the Eventlog is read from a file instead from the system.

File Path Name

File Configuration field:

szLogFileName

Description:

It defines that which file to be read, only available when “Read Eventlog From File” is enabled.

Type of Eventlog

File Configuration field:

szLogType

Description:

It defines as which type of event log from file is handled. This is important to read the correct message libs from the system.

Enable date replacement characters

File Configuration field:

nEnableDateReplacements

description:

Allow the use of dynamic files/paths when using evt files. The same replacement characters as in the FileMonitor apply to this feature. A configured filename may look like this: `C:\temp\evt_%Y%m%d.evt` and would be replaced with `C:\temp\evt_20130101.evt`.

To support changing log file names, there are replacement characters available within the file name. These are:

- %y Year with two digits (e.g. 2002 becomes “02”)
- %Y Year with 4 digits
- %m Month with two digits (e.g. March becomes “03”)
- %M Minute with two digits
- %d Day of month with two digits (e.g. March, 1st becomes “01”)
- %h Hour as two digits
- %S Seconds as two digits. It is hardly believed that this ever be used in reality.
- %w Weekday as one digit. 0 means Sunday, 1 Monday and so on.
- %W Weekday as three-character string. Possible values are “Sun”, “Mon”, “Tue”, “Wed”, “Thu”, “Fri”, “Sat”. This replacement character is most useful for DHCP log files.
- %generatedfilename% It contains the fully generated filename (Can be useful for filtering).
- %msgsep% Only available if enable in the advanced option of the File Monitor. This value contains the current used message separator. This is useful if you want to reconstruct messages where the separator is part of the message.
- %msgseplast% Only available if enable in the advanced option of the File Monitor. This value contains the last used message separator. This is useful if you want to reconstruct messages where the separator is part of the message.

Offset in seconds

File Configuration field:

nEnableDateReplacementsOffset

Description:

When “Enable date replacement characters” is enabled, the current date will be used to generate the filenames. However in certain cases, there is a need to generate filenames with past or future dates. Negative values will generate past filenames, while positive values will generate filenames in the future. For example if you want to generate filenames from yesterday (24 hours back), use -84600 as offset.

Event Types to Log

These checkboxes allow local filtering of the event log. Filtering is based on the Windows event type. There is a checkbox corresponding to each Windows event type. Only checked event types will be processed. Unchecked ones will be ignored.

Filtering out unnecessary log types at this level enhances system performance because no information units will be generated and passed to the rule engine. Thus, Adiscon strongly recommends dropping unnecessary log types.

Ruleset to use

File Configuration field:

szRuleSetName

Description:

Name of the ruleset to be used for this service. The RuleSet name must be a valid RuleSet.

Note

If you intend to make the Event ID part of the actual Syslog message while forwarding to a Syslog server then you have to make some changes in the Event Log Monitor Settings.

Click here **to know the settings.**

The Event Log Monitor caches messages libraries. This greatly speeds up processing, but causes memory consumption for the cached libraries. By default, libraries are cached for 30 minutes. If memory consumption is too high, you may consider to lower the cache period. The cache is global to all event log monitors. As such, its size must be changed in the Engine specific Options Tab. Here you find the Resource Library Cache Options

Event Log Monitor V2

This dialog configures the Windows Event Log Monitor V2 service.

Eventlog Monitor V2

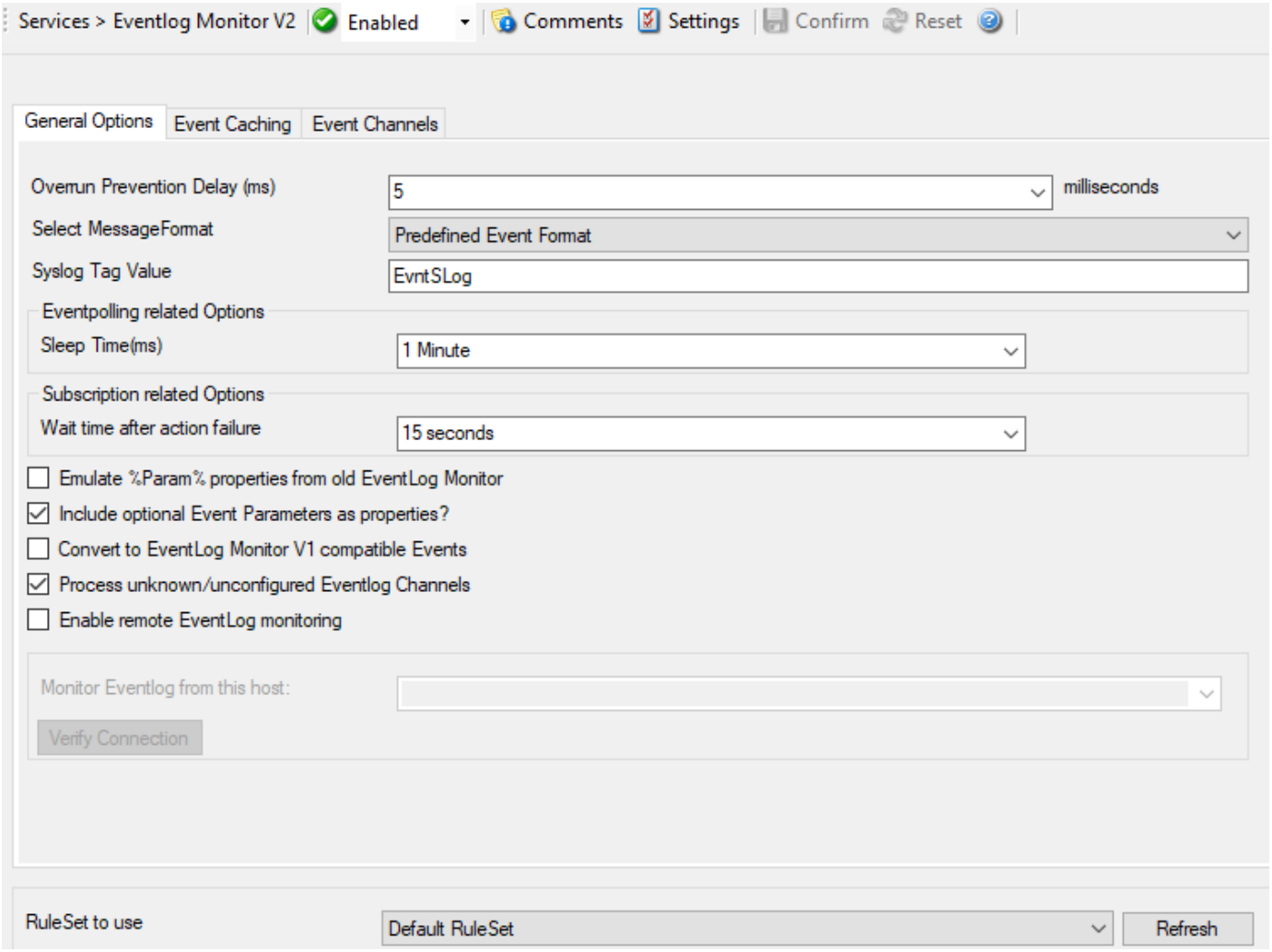
- all modern Windows versions (Windows 10, 11, Server 2016, 2019, 2022, and newer).

Eventlog Monitor

- Windows XP
- Windows 2003

The V2 Eventlog Monitor provides the ability to monitor so-called “log channels”. Each channel can work either in polling or subscription mode. In subscription mode, we are automatically notified by the operating system whenever a new event is logged. In traditional polling mode, we periodically check the channel. In both cases, it is possible for a user to re-set the channel reporting to an older event (parameter “Last Record”).

Both of these functionalities are implemented by periodically iterating over the configured channels. The frequency is controlled by the “Sleep Time” parameter.



- Service - Event Log Monitor V2 - General Options*

General Options Tab

Overrun Prevention Delay (ms)

File Configuration field:

nPreventOverrunDelay

Description:

This property allows configuring a delay after generating an event. The time is the delay in milliseconds.

If run at a value of zero, the Event Log Monitor service generates events as fast as the machine permits. We have seen scenarios where routers and receivers are not able to keep up with this rate, resulting in packet loss. In addition, the CPU of the reporting machine is run at 100% - which is not a problem because the service runs at a low priority. However, with even a 1-millisecond delay, there is no noticeable CPU activity even when large bursts of events are forwarded. At one millisecond, the service can still generate 1000 events per second.

The default setting is an overrun protection of five millisecond, which allows roughly 200 events per second. This should be sufficient for even very busy servers.

Select Message Format

File Configuration field:

nFormatType

- 0 = XML Format
- 1 = Predefined EventFormat

Description:

With this option you can choose whether the Events will be extracted in “Raw XML Format” or in the “Predefined Event Format”.

The XML format is the exact representation of the XML Stream returned by the EventLog System. **Please note that it only contains EventLog data and not a formatted message.**

The “Predefined Event Format” is what the Event in the event viewer looks like.

Copy Format into Property

If enabled, a second message format can be stored into a custom property.

Select Message Format

File Configuration field:

nCopyFormatIntoProperty

Description

Select which message format should be stored into the custom property.

Store into Property

File Configuration field:

szCopyFormatIntoProperty

Description

The custom message property, for the “Copy Format into Property” Option.

SyslogTag Value

File Configuration field:

szSyslogTagValue

Description:

The SyslogTag Value determines the SyslogTag that is used when forwarding Events via syslog. This is useful, if you want to determine later, what kind of syslog message this is, perhaps because you log EventLogs and syslog into the same database.

Eventpolling related Option: Sleep Time(ms)

File Configuration field:

nSleepTime

Description:

As said in the overview, this controls iteration over the configured channels. The value is specified in milliseconds.

For channels configured to use Polling Mode, the “Sleep Time” parameter specifies how often they are processed. Note that when multiple channels are set to polling mode, they are processed one after another. So there is a somewhat larger delay in processing than given by the “Sleep Time” parameter. The total frequency depends on how busy all polling channels are.

For channels configured to subscription mode, the “Sleep Time” interval will only influence how often a potential reset of “Last Record” is checked. Other than that, it has no effect on the event delivery rate.

We suggest a value of 60,000 milliseconds for the “Sleep Time”. With that setting, the Event Log Monitor checks for new events every 60 seconds. Larger periods can be specified for occasionally connected systems or if email delivery with few emails per day is intended.

Very security-aware environments might use a shorter interval. The event log monitor service is specifically designed to limit the burden on the monitored system. As such, resource usage is typically low, even with frequently run event log checks. However, we do NOT recommend running the Event Log Monitor more often than once a second.

Note that it is almost always an error to use a “Sleep Time” value of 0. The main processing loop of the EventLog Monitor V2 would re-run without any delay and would cause a very high CPU usage, close to 100%.

For these reasons, newer versions of the product will no longer permit to use a “Sleep Time” of zero and automatically change it to one

Subscription related Option: Wait time after action failure

File Configuration field:

nSubscriptionSleepTime

Description:

Adds some extra wait time (Delay) if an action failed to process. Without the delay, the subscription would immediately process the last event again. In some cases a reasonable delay before a retry is needed.

Emulate %Param% properties from old EventLog Monitor

File Configuration field:

nEmulateParameters

Description:

This option emulates the %Param% properties, which were often used in the old EventLog Monitor. The new EventLog implementation does not support them in the same way anymore. The Event Log Monitor V2 is still able to provide parameters in the “old style” format, what means that log analysis scripts can receive a consistent stream of data for both new style and old style Windows events.

Include optional Event Parameters as properties?

File Configuration field:

nIncludeEventParameters

Description:

If enabled, the < EventData> node from the raw XML stream (Eventlog entry) will be searched for variables. If variables with names are found, they will be set as Properties with their variable name automatically. If the variable does not have a name, it will be set to a common name like “Param1, Param2 …. ParamX”.

Convert to EventLog Monitor V1

File Configuration field:

nConvertToEventLogMonitorV1

Description:

This option maps the EventID’s from the Security EventLog back to V1 (Windows 2000/2003). The internal InforUnitID is also changed to V1. This option helps postprocessing EventLog V1 and V2 events equally.

Process unknown/unconfigured Eventlog Channels

File Configuration field:

nEnableUnknownChannels

Description:

If enabled, unconfigured Eventlog Channels (Those found on the system, but not listed in the Eventlog Channels list) will automatically be processed.

Enable remote EventLog monitoring

File Configuration field:

nEnabledRemote

Description:

If enabled, EventLog Monitor will connect to a remote machine to process the EventLog. Please note that the RPC Service needs to be installed on the remote machine, and the Service has to be configured with a network user that has sufficient access rights.

Monitor Eventlog from this host

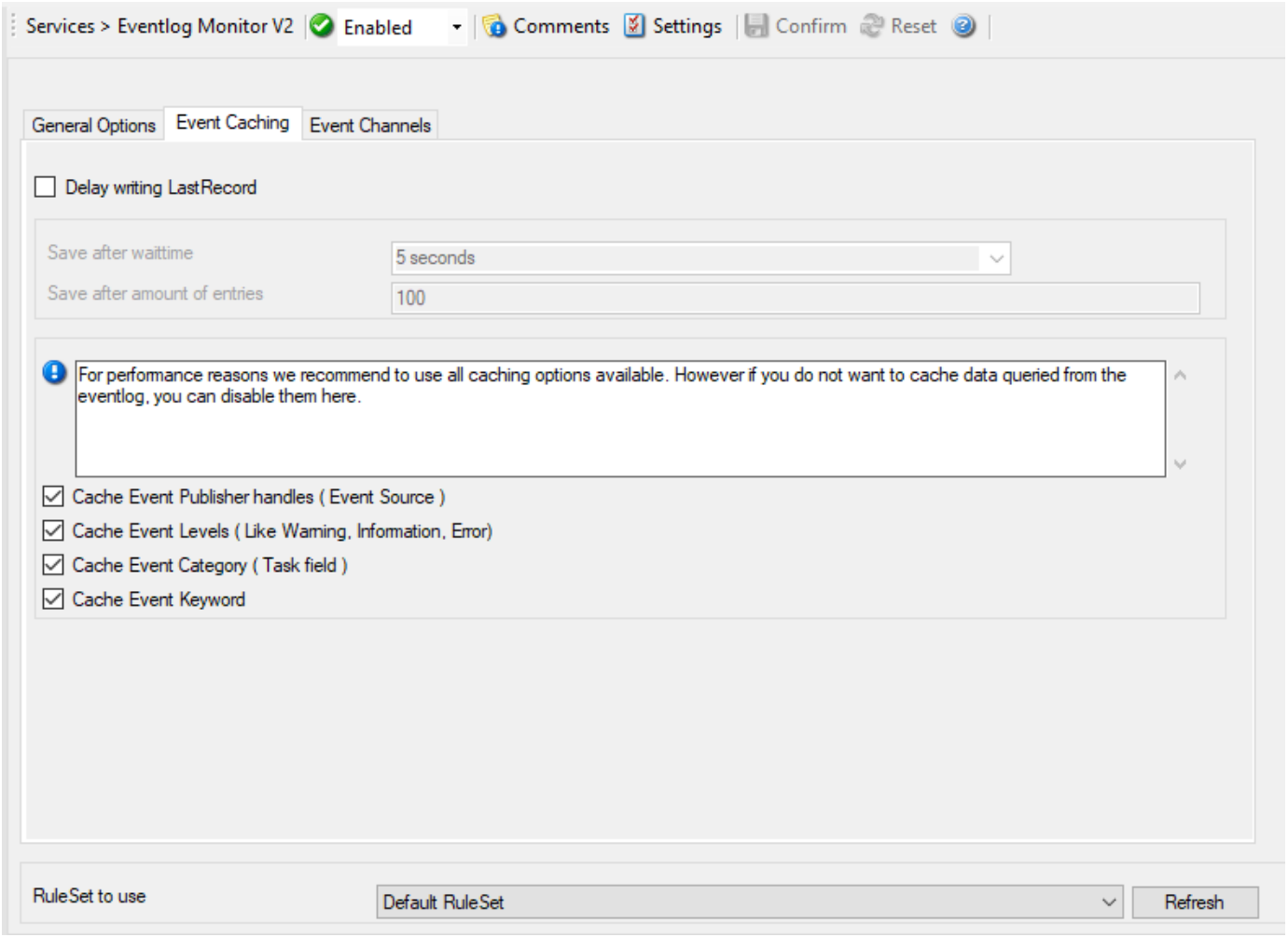
File Configuration field:

szServerName

Description:

The hostname of the remote server to be monitored by EventLog Monitor.

Event Caching Tab



- Service - Event Log Monitor V2 - Event Caching*

Delay writing LastRecord

File Configuration field:

nEnableLastRecordDelay

Description:

Enables the LastRecord writeback delay to the configured properties below.

Save after waittime

File Configuration field:

nLastRecordDelayTime

Description:

Regardless of the amount of processed event log entries, the lastrecord value will delayed for this waittime period.

Save after amount of entries

File Configuration field:

nLastRecordDelayCount

Description:

Regardless of the configured waittime period, the LastRecord will be written after the amount of processed event log entries that are specified here.

Cache Event Publisher handles (Event Source)

File Configuration field:

nCacheEventPublisher

Description:

All publisher sources will be kept open once loaded until the application / service is stopped. This increases processing speed for events from same sources.

Cache Event Levels (Like Warning, Information, Error)

File Configuration field:

nCacheEventLevel

Description:

If enabled the textual representations for event levels will be cached.

Cache Event Category (Task field)

File Configuration field:

nCacheEventCategory

Description:

If enabled, all textual representations for event categories will be cached.

Cache Event Keyword

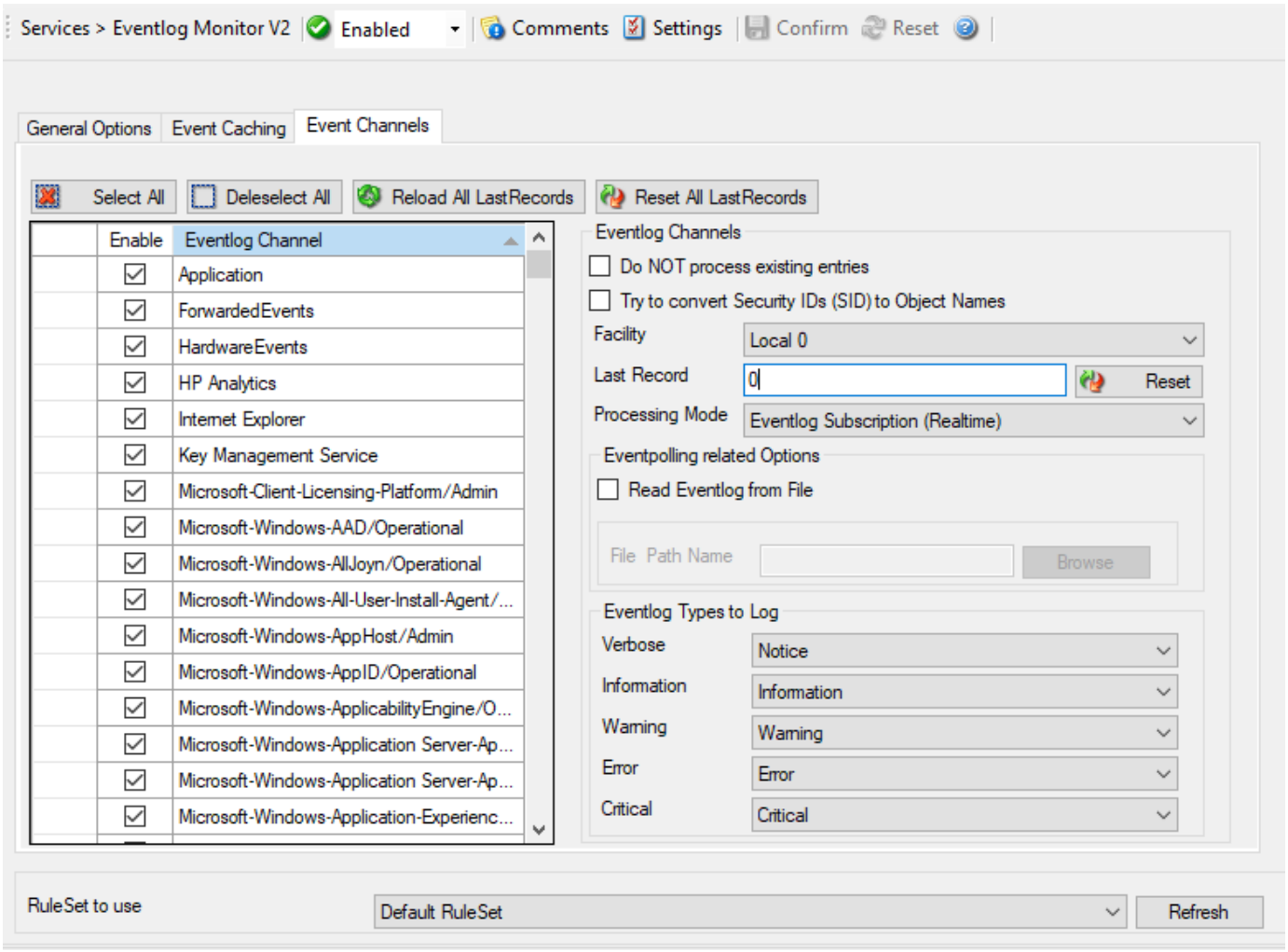
File Configuration field:

nCacheEventKeyword

Description:

If enabled, all textual representations for event keywords will be cached.

Event Channels Tab



- Service - Event Log Monitor V2 Event Channels*

The most important part of this dialog is the treeview of available Channels. It specifies which event logs are to be monitored. The monitor is set to all Channels that are currently available. There happen to be custom Channels, too, due to Applications creating them on their own. They will be added to the treeview automatically every time you re-open this configuration window.

Here you can adjust the syslog facility and the event log types. You are also able to overwrite all existing custom advanced channel configurations with your new ones.

Channels which are checked in the table will be processed. Channels which are unchecked are kept in the configuration, but are not processed.

Do NOT process existing entries

File Configuration field:

nNoExistingEntries

Description:

If you do not want to get a dump of an existing specific Windows Event Log then use this option. When MonitorWare Agent / EventReporter are restarted it will start processing after that last entry and do not look for the previous entries.

Try to convert Security IDs (SID) to Object Names

File Configuration field:

nTryConvSIDtoObj

Description:

With this option you can convert Security ID's (SIDs) to object names. You can enable this feature in the advanced configuration of each event log type in the Event Log Monitor service. Simple check the "Try to convert Security IDs (SID) to Object Names" option.

Note that only the Security event log has this feature enabled by default. For all other event log types this feature is disabled by default.

Facility

File Configuration field:

nFacility

Description:

The syslog facility to map information units stemming from this log to. Most useful if the message is to forward to a Syslog daemon.

Last Record

File Configuration field:

szXMLBookmark

Description:

Windows Event Log records are numbered serially, starting at one. The service records the last record processed. This textbox allows you to override this value. **Use it with caution!**

If you would like a complete dump of a specific Windows Event Log, reset the "Last Record" to zero. If you missed some events, simply reset it to some lower value than currently set. It is possible to set "Last Record" to a higher value. This suspends event reporting until that record has been created. We strongly discourage to use this feature unless definitely needed.

Processing Mode

File Configuration field:

nApiReadMode

- 0 = Subscription Readmode (Real-time)
- 1 = Polling Readmode (Sleeptime)

Description:

There are two processing modes available, first the default processing mode is "EventLog Subscription" which processes Events in real time. This means events are send to MonitorWare Agent by the OS as they happen, there is no delay at all. The other processing mode called "Eventlog Polling" and is similar to the method used in the old EventLog Monitor. The EventLog is checked and processed periodically controlled by the sleeptime. However using the polling method, you enable the "Read EventLog From File" option.

Eventpolling related Options

Read Eventlog from File

File Configuration field:

nReadFromFile

Description:

When enabled, the Eventlog is read from a file instead from the system.

File Path Name

File Configuration field:

szLogFileName

Description:

It defines which is file to be read, only available when "Read Eventlog From File" is enabled.

Event Types to Log

These checkboxes allow local filtering of the event log. Filtering is based on the Windows event type. There is a checkbox corresponding to each Windows event type. Only checked event types will be processed. Unchecked ones will be ignored.

Filtering out unnecessary log types at this level enhances system performance because no information units will be generated and passed to the rule engine. Thus, Adiscon strongly recommends dropping unnecessary log types.

RuleSet to use

File Configuration field:

szRuleSetName

Description:

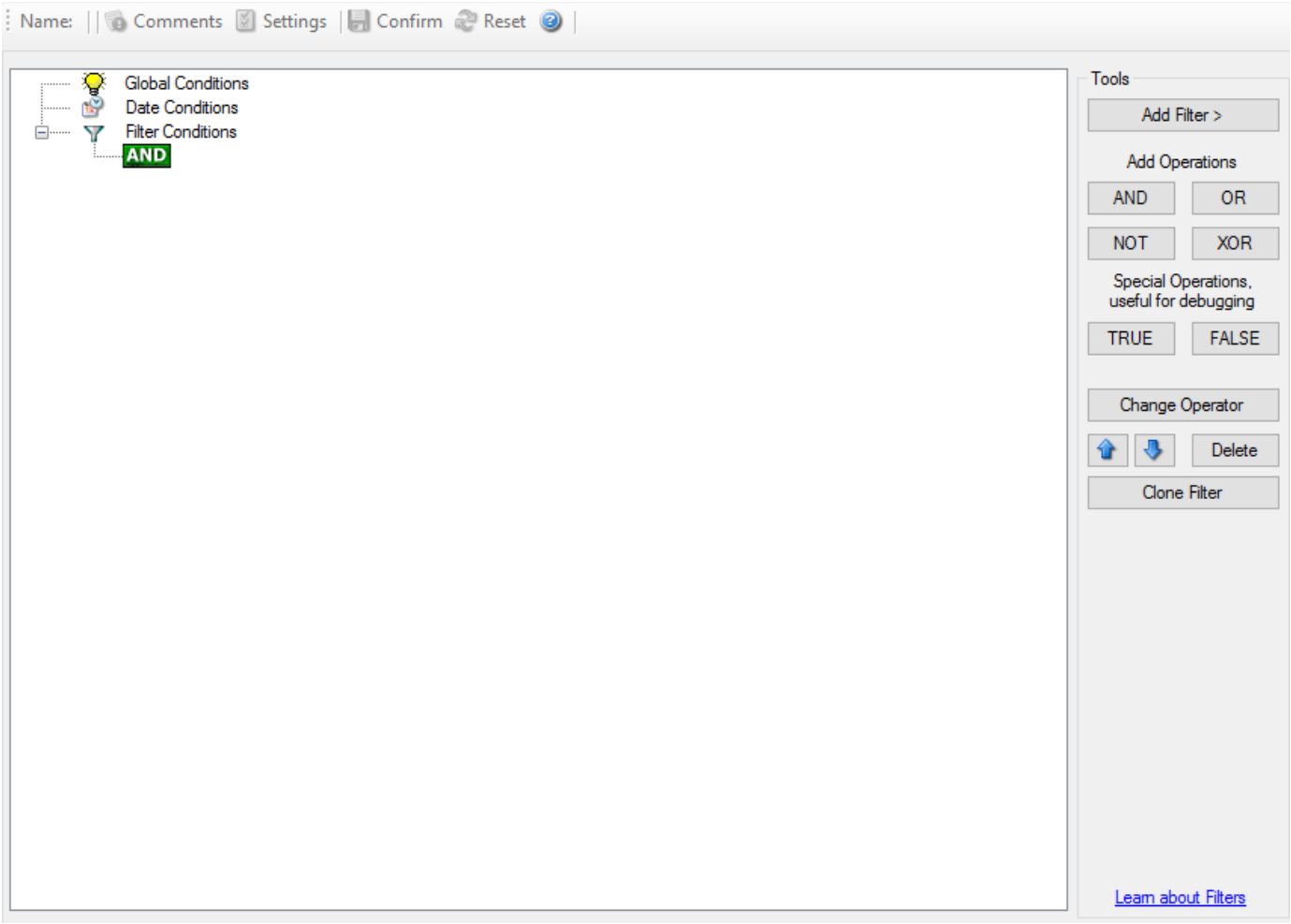
Name of the ruleset to be used for this service. The RuleSet name must be a valid RuleSet.

Filter Conditions

Filter conditions specify **when** to apply a rule. If the filter condition evaluates to true, the rule containing those conditions is treated as matching and the actions specified in that rule are carried out.

Filter conditions can be as complex as needed. Full support for Boolean operations and nesting of conditions is supported.

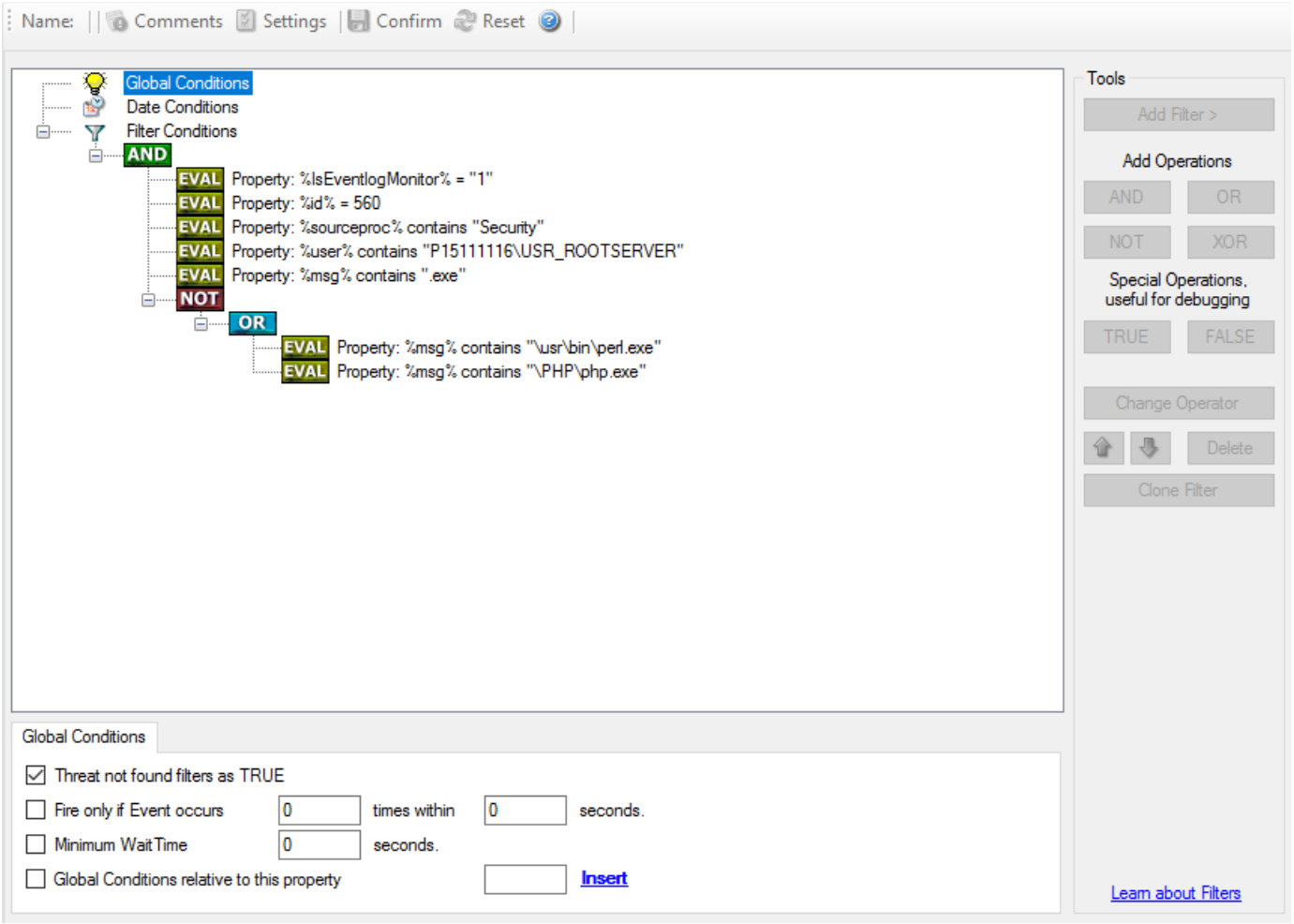
By default, the filter condition is empty, respective tree contains only a single “AND” at the top level. This is to facilitate adding filters (the top level-node is typically “AND” and thus provided by default). A filter condition containing only the “AND” always evaluates as true. A sample screenshot can be found below



Filter Conditions - Figure 1

The default filter condition means that the actions associated with the rule are to be carried out for every information unit received. It is often used for actions that should be broadly taken, for example to write all incoming information units to a database or text file.

On the other hand, there are actions that should only be executed under very special conditions. They may even require a complex filter condition including multiple levels of Boolean operations. Below is a sample of such a condition:



Filter Conditions - Figure 2

This filter condition is part of an intrusion detection ruleset. Here, Windows file system auditing is used to detect a potentially successful intrusion via Internet Information Server (IIS). This is done by enabling auditing on all executable files. Internet Information Server accesses them under the `IUSR_<machinename>` account, which in our sample is `"P15111116\IUSR_ROOTSERVER"`. If that user runs any unexpected executables, chances are good that someone was able to intrude the machine via IIS. Please note that Perl and PHP scripts need to run the Perl and PHP engine. This is reflected by specifically checking, if `perl.exe` and `php.exe` is executed – and if so, no alarm is triggered.

Here is how the above sample works: first, the message contents are checked if it contains either the full path name to `perl.exe` or `php.exe`. This is done in the “OR” branch at the bottom. We now need to keep in mind that when a filter condition evaluates to “true”, the actions are executed. In case of `perl.exe` and `php.exe`, this is just the opposite of what we want. We need it to be executed, when other files are executed. Consequently, we negate (Boolean “NOT”) the result of the OR. The end result of the “NOT” operation is then combined via a “AND” with some other properties describing the event we need.

First, we check if the specific event really occurred. For this, we need to make sure we deal with an Event Log Monitor information unit. Then, these information units are identified by the event source as well as the Event ID. We also check for the Event User to identify only IIS generated requests. Lastly, we check if the message contains the string `".exe"`.

In order to avoid too frequent alerts, we also have specified a minimum wait time of 60 seconds. Therefore, the filter condition evaluates as “true” at most every 60 seconds, even if all other conditions are true. **Note:** If you want to know more about complex filter conditions you can click on the “Learn about Filters” link.

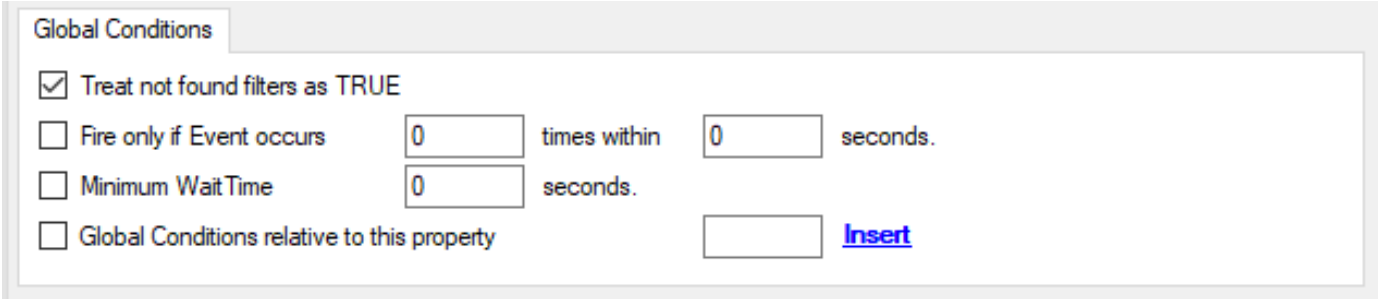
String comparison in Filter Conditions are “Case Sensitive”! For example, if the Source System name is “ws01” and you had written “WS01” while applying the

filter, then this filter condition would**”NEVER”** evaluate to True! Please double check before proceeding further!

If you are not still sure about what to do, you can drop a word about your requirements to https://ticket.adiscon.com, and we look into it!

Global Conditions

Global Conditions apply to the rule as whole. They are automatically combined with a logical “AND” with the conditions in the filter tree.



- Global Conditions*

Treat not found Filters as TRUE

If a property queried in a filter condition is not present in the event, the respective condition normally returns “FALSE”. However, there might be situations where you would prefer if the rule engine would evaluate this to “TRUE” instead. With this option, you can select the intended behavior. If you check it, conditions with properties not found in the event evaluates to “TRUE.

Fire only if Event occurs

This is kind of the opposite of the “Minimum WaitTime”. Here, multiple events must come in before a rule fires. For example, this time we use a ping probe. Ping is not a very reliable protocol, so a single ping might be lost. Thus, it may not be the best idea to restart some processes just because a single ping failed. It would be much better to wait for repetitive pings to fail before doing so.

Exactly this is why the “Fire only if Event Occurs” filter condition is made for. It waits until a configured amount of the same events occurs within a period. Only if the count is reached, the filter condition matches and the rule can fire.

Note: If you used previous versions of the product, you might remember a filter called “Occurrences”. This has just been renamed.

Minimum Wait Time

This filter condition can be used to prevent rules from firing too often. For example, a rule might be created to check the status of a port probe event. The port probe probes an smtp server. If the event is fired and the rule detects it, it spawns a process that tries to restart the service. This process takes some time. Maybe the SMTP gateway need some more time to fully start up so that the port probe might fail again while the problem is already taken care of. The port probe as such generates an additional event.

Setting a minimum wait time prevents this second port probe event to fire again if it is – let’s say – within 5 minutes from the original one. In this case, the minimum wait time is not yet reached and as such, the rule does not match. If, however, the same event is generated 5 hours later (with the mail gateway failing again), the rule once again fires and corrective actions are taken.

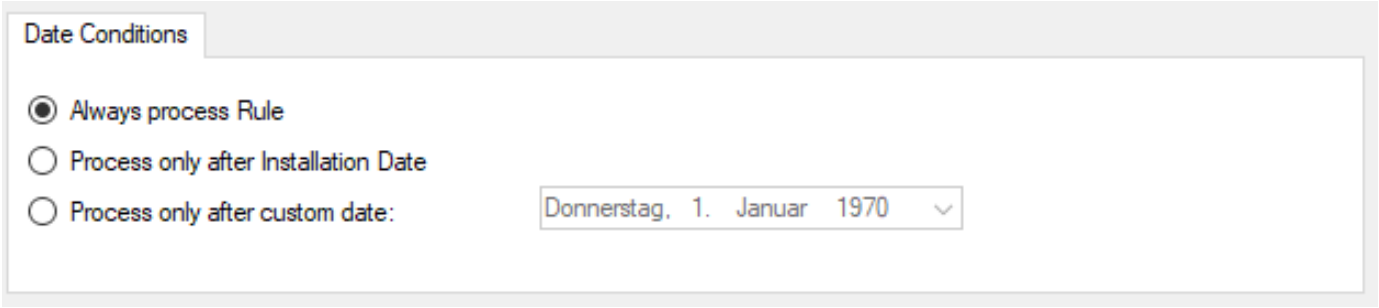
Global Conditions relative to this property

This feature enables you to control the Global Conditions based on a property.

For example take the source of a message as property. In this case, the Minimum WaitTime for example would be applied individual on each message source.

Date Conditions

Rule processing can be bound to a specific or the installation date. By default a Rule will always be processed.



- Date Conditions*

Always process Rule

No date filter will be applied

Process only after Installation Date

Rule will only be processed if message was generated after the application installation date.

Process only after custom date

Rule will only be processed if message was generated after the custom specified date.

Operators

In general, operators describes how filter conditions are linked together. The following operators can be used. Boolean operators like “AND” or “OR” can be used to create complex filter conditions.

AND:

All filters placed below must be true. Only then AND returns TRUE.

OR:

If one or both of the filters placed below is true, OR returns TRUE.

NOT:

Only one Filter can be placed below NOT operator, and if the filter evaluation is true, NOT returns FALSE.

XOR:

If one of the two filters are possible in the XOR Operator.

TRUE:

Useful for debugging, just returns TRUE.

FALSE:

Useful for debugging as well, returns FALSE.

Filters

Filters can be added under each Operation node. There are a few common filters which can be used for all services and there are special filters which only apply if a special kind of Information Unit is evaluated.

What happens with Filters that are not available in an “Information Unit”?

Every filter that is not found in an Information Unit is ignored in the filtering process. If you want to create filters specialized for types of Information Units, always make sure to add an “Information Unit Type” filter.

An example, you have one ruleset, rule and action. In the filters you have one EventID filter. Then you have two services, one Eventlog Monitor and the other is Heartbeat monitor both pointing to this ruleset. The Information Units from the Eventlog Monitor would be filtered correctly, but those from the Heartbeat monitor would not be filtered as they do not have an EventID property. The EventID filter would be ignored and the actions would be executed every time.

Note, if a filter is used that does not apply to the evaluated Info Unit, it will be just ignored. This gives you the possibility to build one filter set for several types of Information Units.

There are different types of filters, and so there a different ways in which you can compare them to a value. The following Types exist:

String:

Can be compared to another String with “=”, “Not =”, “Range Match” or through

REGEX Compare Operation

The property will be evaluated against a regular expression. Everything known in the regular expression syntax can be used to define a matching pattern.

Here are some regular expressions samples:

Regular Expression:

`[0-9]{4,4}-[0-9]{1,2}-[0-9]{1,2} [0-9]{1,2}:[0-9]{1,2}:[0-9]{1,2}`

Matches typical Date like `2018-11-20 12:11:01`

Regular Expression: `\n[0-9]{4,4}`

Matches Linefeed and 4-digit number.

Regular Expression: `(;|:)` Matches semicolon or a colon.

More samples and details about the Regular Expression Syntax can be found here:

https://msdn.microsoft.com/en-us/library/bb982727(v=vs.90).aspx

number:

can be compared with another number with “=”, “not =”, “<” and “>”

boolean:

can be compared to either true or false with “=” and “not =”

time:

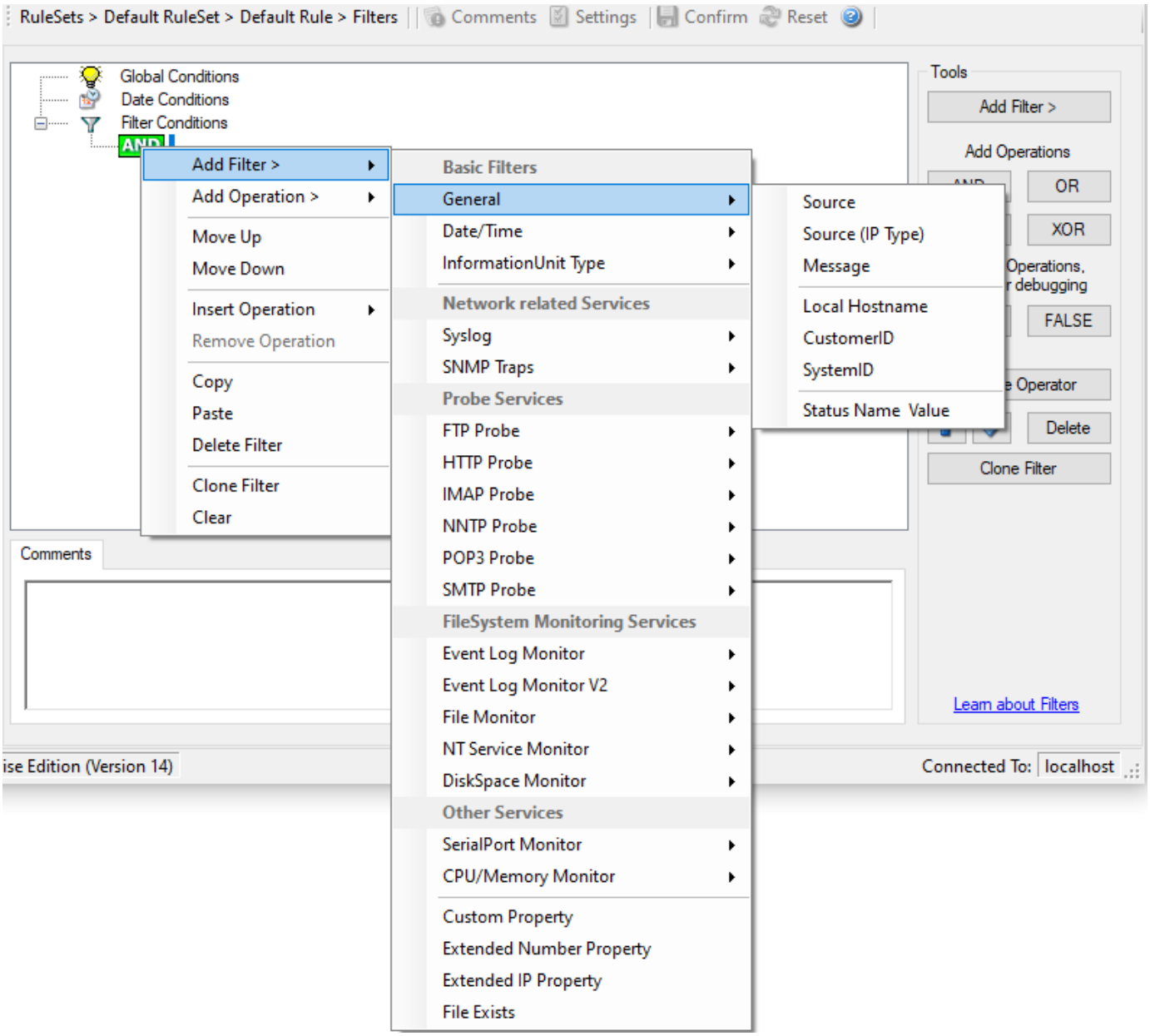
can be compared with another time but only with “=”

the list of possible filters, which can be evaluated is described in the following sections.

basic filters

General

These are non-event log specific settings.



- Filter Conditions - General*

Source

This filter condition checks the system that generated the information unit. For example, in case of the Syslog server, this is the Syslog device sending a Syslog message.

This filter is of type string and should contain the source system name or IP address.

Source System (IP)

The IP Filter can basically work on any property, but we recommend to only use it on the %source% property, as we usually can be sure that this is contains a valid IP Address or hostname. The IP Filter can filter against hostnames and IP Addresses, hostnames are automatically resolved using the internal DNSCache (for obvious performance reasons).

This filter is of type string and should contain the source system name or IP address.

Please see the description for extended ip property for more information on how to use this property.

Message Content

The message content filter condition is very powerful. It evaluates to true if the specified content is found anywhere within the message. As there is implicit wildcarding, there is no need for extra wildcards to be specified.

The content search can be limited to a region within the message. To do so, select a starting and ending position within the string by choosing the **“contains within range”** compare operation. This can be done by specifying the start range and end range into the respective boxes.**Please note that you can enter the character position you desire in these fields. The default “Start Range” and “End Range” are set to 0.**

If you would like to search for a string just between positions 10 and 50, specify these values as start and end values, respectively. Similarly if you want to receive all logs from 192.168.0.1 then set this as:

- Property value = 192.168.0.0
- Range Start = 0
- Range End = 10

Which means 10 characters starting at zero (“192.168.0.”). Please note that the final DOT must be included. If you just used range “9”, then 192.168.010 would also match.

This filter is of type string.

CustomerID

CustomerID is of type integer provided for customer ease. For example if someone monitors his customer’s server, he can put in different CustomerIDs into each of the agents. Let us say someone monitors servers A and B. A has 5 servers all of them with CustomerID = 1 and B has 2 servers all of them with CustomerID = 2. Both A and B happen to have a server named “SERVER”. Together with the customerID, these machines are now uniquely identifiable. This is user configurable.

CustomerID (Type=Number).

SystemID

SystemID is of type integer to be used by our customer. In addition, it is user configurable.

SystemID (Type=Number).

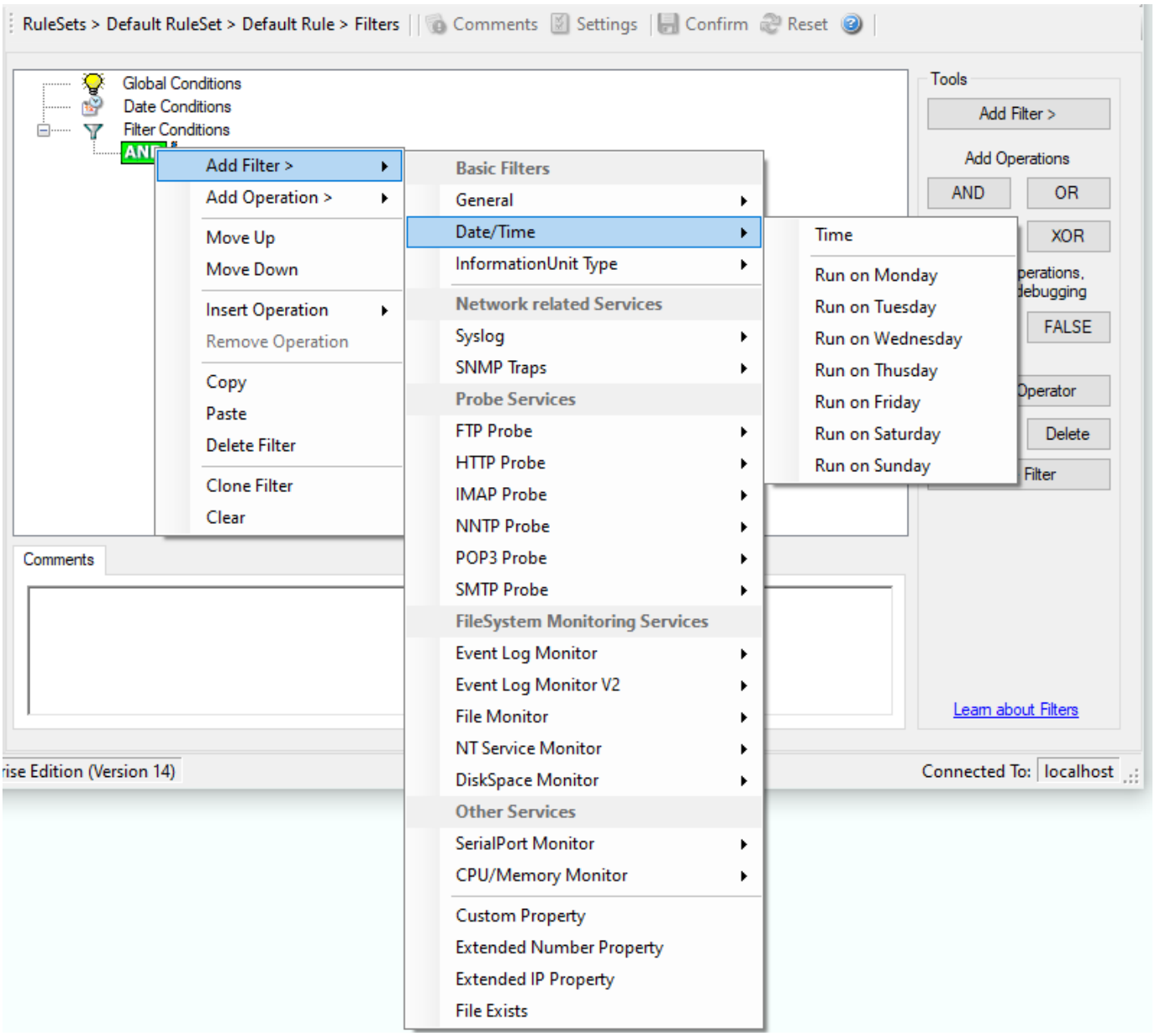
Status Name and Value

These filter type corresponds to set status action .

Status Name and Value (Type=String)

Date/Time

This filter condition is used to check the time frame and / or day of week in which an event occurred.



- Filter Conditions - Date/Time*

Time

This filter condition is used to check the period in which an event occurred. For example, a Syslog message from a Cisco router saying that it dialed up is normal if it occurs during office hours. If it occurs at night, so, it is an alerting signal and an administrator might receive notification of this event (while he might otherwise decide to discard it). This can be done with the time setting.

You can also set the timezone setting (DefaultTimemode, UTC or Localtime) for the TimeMode's (DeviceReportedTime/ReceivedTime).

Weekdays

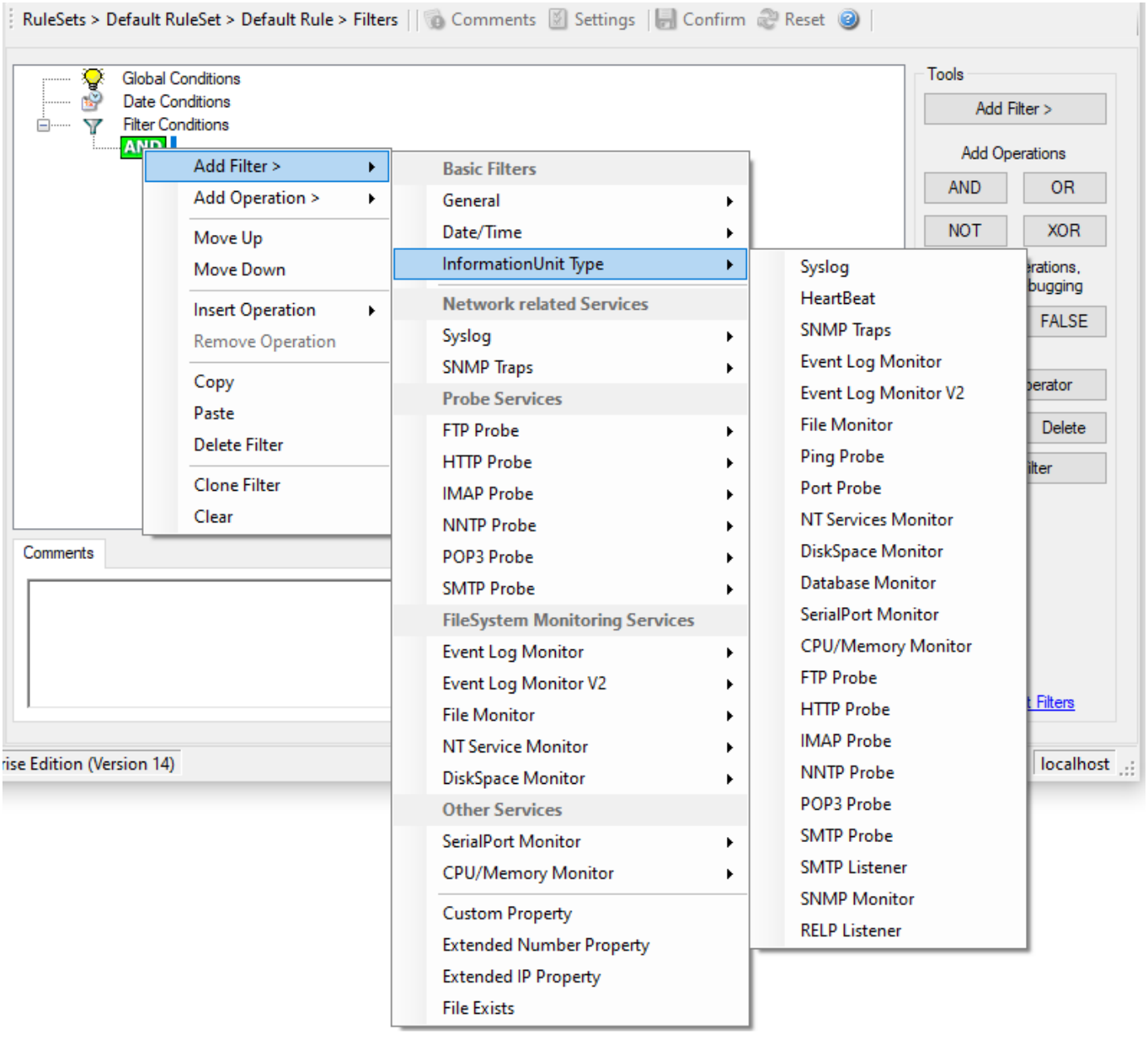
This is closely equivalent to the time filter condition, except that it is applied on a per-day basis. So it can be used to detect for example events occurring on weekends and act differently on them. The following filters are available:

1. Run on Monday (Type=Boolean)
2. Run on Tuesday (Type=Boolean)
3. Run on Wednesday (Type=Boolean)
4. Run on Thursday (Type=Boolean)
5. Run on Friday (Type=Boolean)

6. Run on Saturday (Type=Boolean)
7. Run on Sunday (Type=Boolean)

InformationUnit Type

Select the specific information if a rule should just be processed for some information unit types. This is especially useful if a specific type needs non-standard processing. There is one pre-defined filter for each possible InformationUnit Type available (shown below).



- Filter Conditions - InformationUnit Type*

The following filters are available:

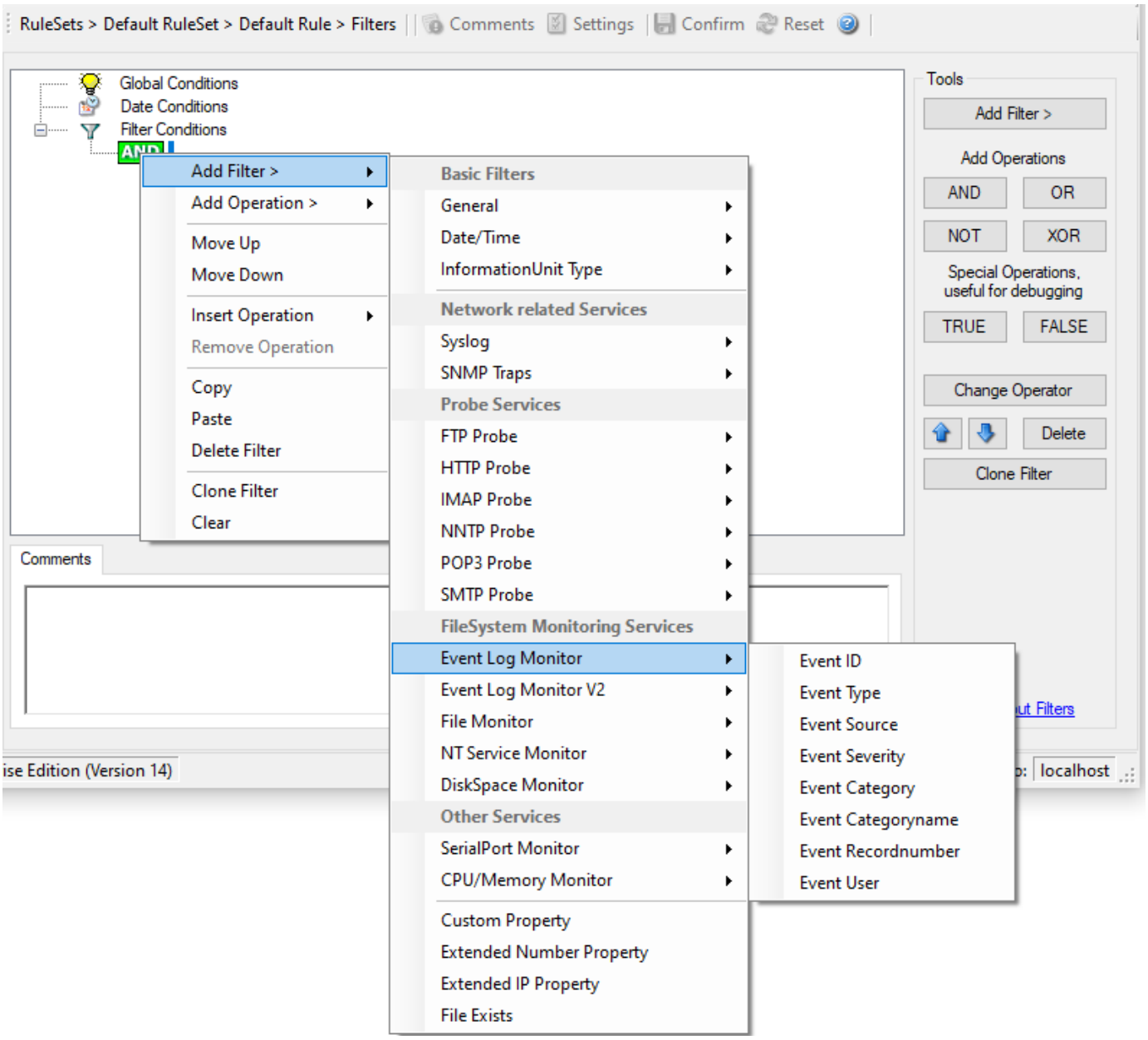
1. Syslog (Type=Boolean)
2. Heartbeat (Type=Boolean)
3. SNMP Traps (Type=Boolean)
4. Event Log Monitor (Type=Boolean)
5. File Monitor (Type=Boolean)
6. Ping Probe (Type=Boolean)

7. Port Probe (Type=Boolean)
8. NT Services Monitor (Type=Boolean)
9. Disk Space Monitor (Type=Boolean)
10. Database Monitor (Type=Boolean)
11. Serial Port Monitor (Type=Boolean)
12. CPU/Memory Monitor (Type=Boolean)
13. FTP Probe (Type=Boolean)
14. HTTP Probe (Type=Boolean)
15. IMAP Probe (Type=Boolean)
16. NNTP Probe (Type=Boolean)
17. POP3 Probe (Type=Boolean)
18. SMTP Probe (Type=Boolean)

filesystem monitoring filters

Event Log Monitor

Event Log Monitor specific filters are grouped here.



- Filter Conditions - Event Log Monitor V1*

Event ID

This is the event log ID as specified in the Windows Event Log. If enabled, the event must have the configured event ID or the rule will not match. This is an integer value.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type number.

Event Type

This is the event log type as specified in the Windows Event Log. If enabled, the event must have the configured event type or the rule will not match. The supported values can be selected from the list box.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

Event Source

This is the event log source as specified in the Windows Event Log. If enabled, the event must have the configured event source or the rule will not match. This is a string value. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

Event Severity

This is the event log severity as specified in the Windows Event Log. If enabled, the event must have the configured severity or the rule will not match. The supported values can be selected from the list box.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type number.

Event Category

This is the event log category as specified in the Windows Event Log. If enabled, the event must have the configured event category or the rule will not match.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type number.

Event Categoryname

This value contains the Category value as string if it can be resolved. Otherwise it contains the category number.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

Event Recordnumber

This value contains the internal event record number. Please note that if the event log has been truncated before, it may not start with 0 or 1 but a higher number.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type number.

Event User

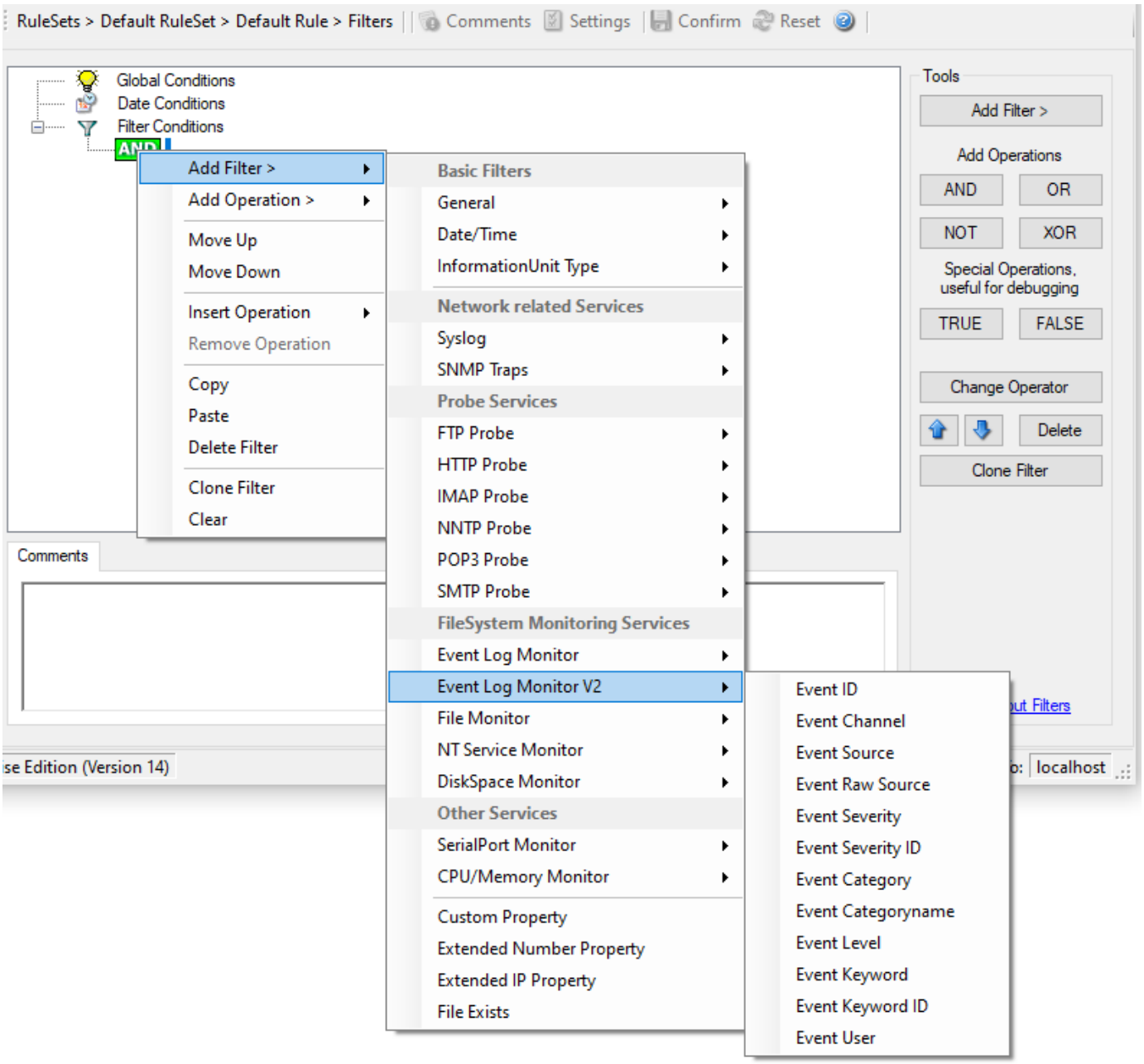
This is the event log user as specified in the Windows Event Log. If enabled, the event must have the configured event user or the rule will not match. Since it is a string value there must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

Event Log Monitor V2

Event Log Monitor V2 specific filters are grouped here.



- Filter Conditions - Event Log Monitor V2*

Event Channel

The channel property for event log entries, for classic Event logs they match the %nteventlogtype% property, for new event logs, they match the “Event Channel”. If enabled, the event must have the configured event type or the rule will not match. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

Event Raw Source

This contains the full internal name of the event source for new event logs, for classic event logs it contains the same value as in %sourceproc%. If enabled, the event must have the configured event source or the rule will not match. This is a string value. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

Event SeverityID

This is the internal ID of the event log level as number. This is a integer value. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type number.

Event Level

This is a textual representation of the event log level (which is stored as number in %severityid%). This property is automatically localized by the system. If enabled, the event must have the configured level or the rule will not match. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

Event Keyword

This is a textual representation of the event keyword. This property is automatically localized by the system. If enabled, the event must have the configured event keyword or the rule will not match. There must be an exact match. Please note that this value is case-sensitive.

This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

Event KeywordID

This is the internal keyword ID as string. If enabled, the event must have the configured event keyword ID or the rule will not match. There must be an exact match. Please note that this value is case-sensitive.

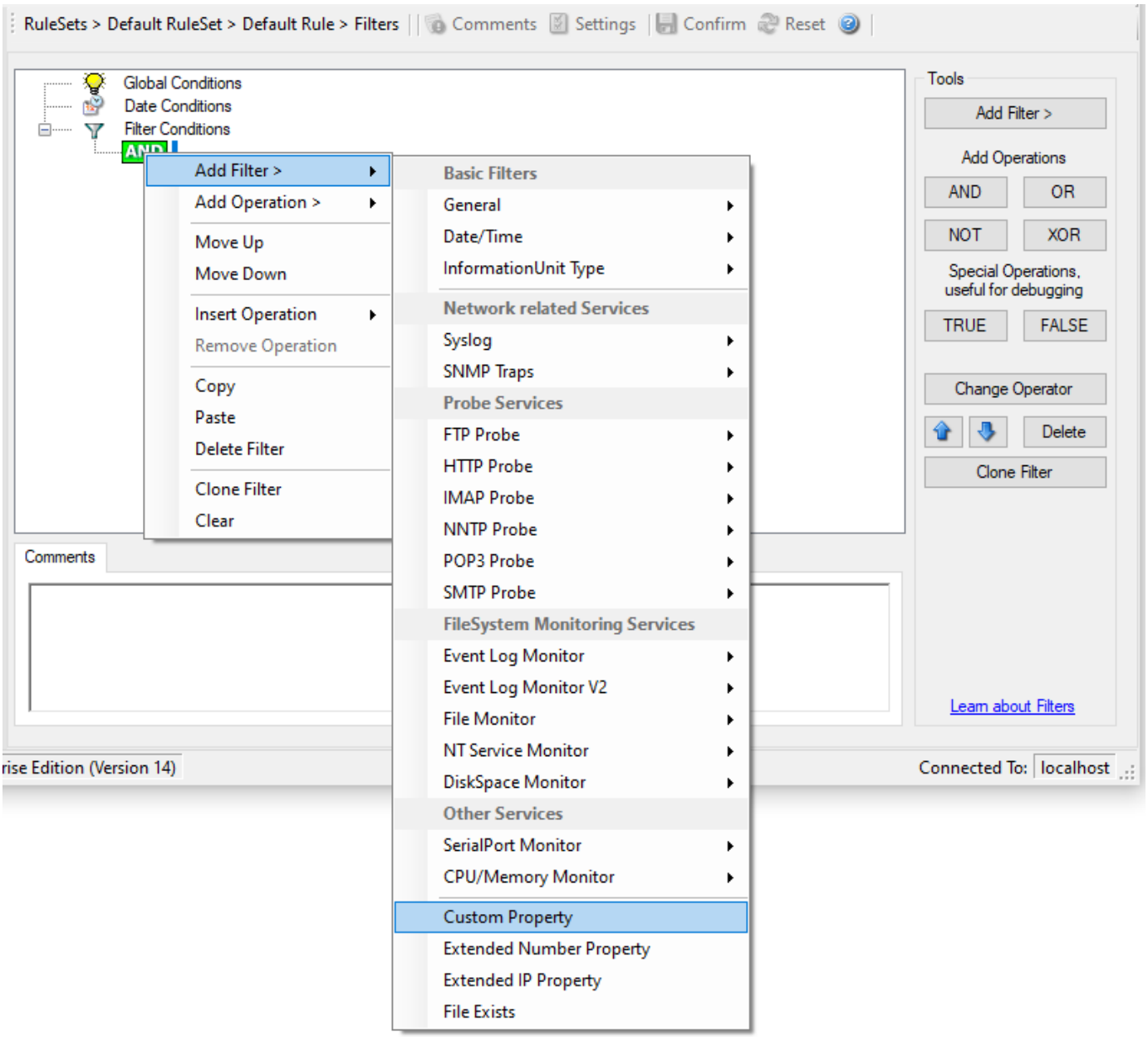
This filter condition should only be used with event log information units. If used with others, a mapped value is to be used which might not properly reflect the actual value.

This filter is of type string.

custom properties

Custom Property

Custom Property specific filter is described here.



- Filter Conditions - Custom Property*

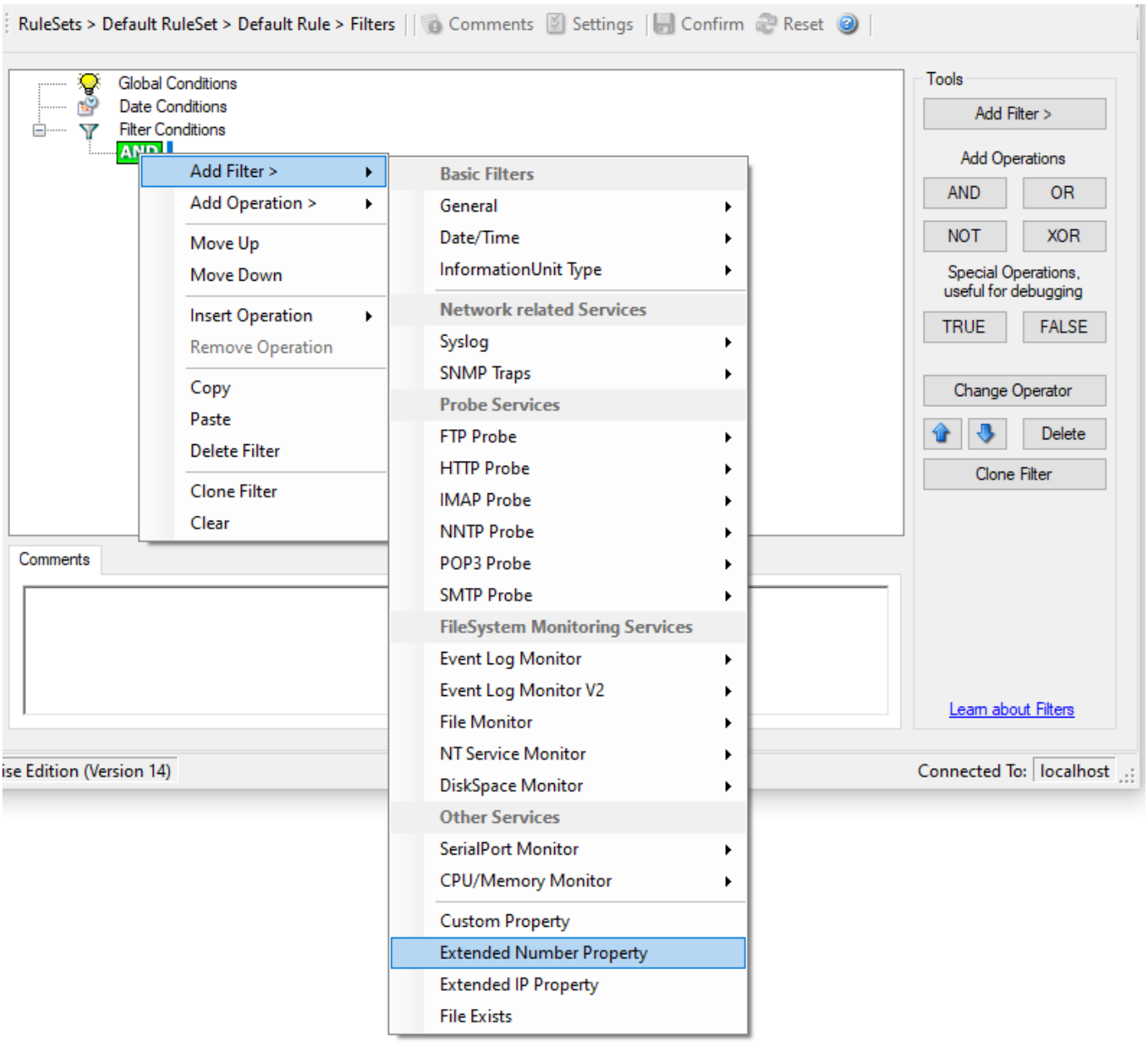
Custom Property

As the name suggests it is a “Custom Property”. Internally in MonitorWare Agent all values are stored in properties. For example the main message is stored in a property called “msg”. By using this dialog you can access properties which are dynamic (Like those from SNMP Trap Monitor when using V2 protocol).

This filter is of type string.

Extended Number Property

Extended Number Property specific filter is described here.



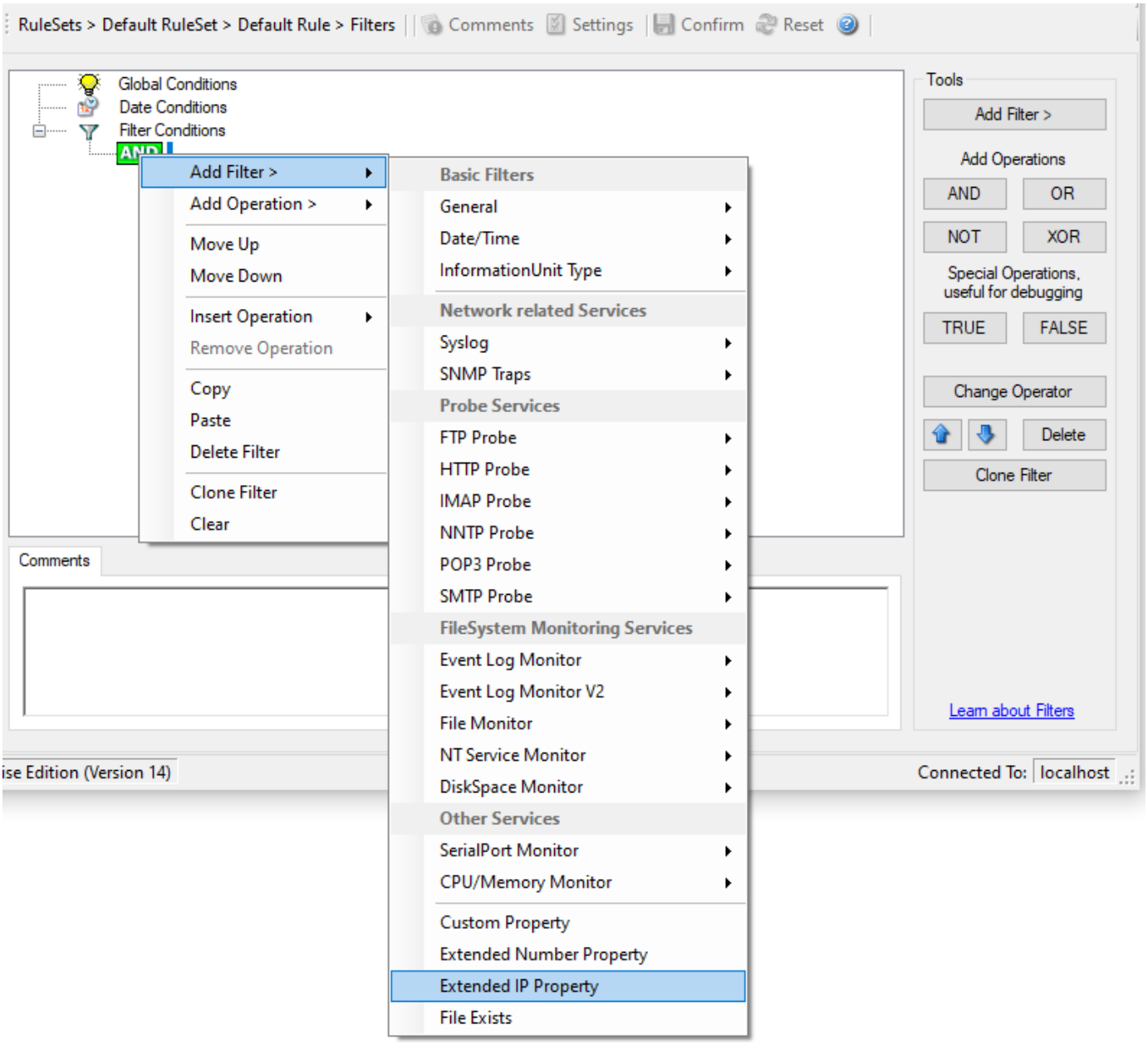
- Filter Conditions - Extended Number Property*

Extended Number Property

As the name suggests it is a “Extended Number Property”. Internally in MonitorWare Agent all values are stored in properties. For example the main message is stored in a property called “msg”. By using this dialog you can access properties which are dynamic (Like those from SNMP Trap Monitor when using V2 protocol).

This filter is of type numeric.

Extended IP Property



- Filter Conditions - Extended IP Property*

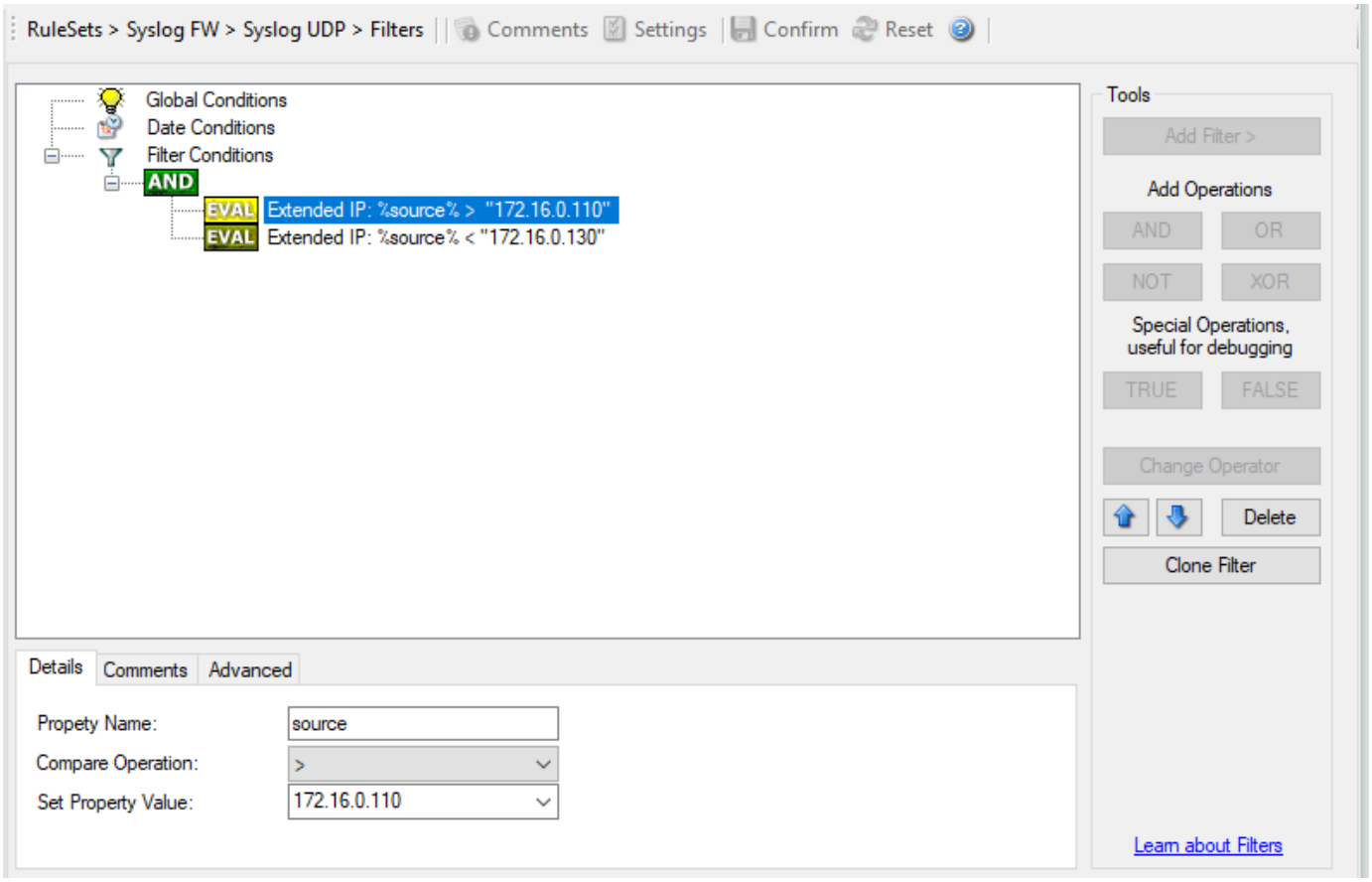
Extended IP Property filter settings

The IP Filter can basically work on any property, but we recommend to only use it on the %source% property, as we usually can be sure that this contains a valid IP Address or hostname. The IP Filter can filter against hostnames and IP Addresses, hostnames are automatically resolved using the internal DNSCache (for obvious performance reasons). If you are going to use a different or custom property, please make sure, that the data in the property is a valid IP Address.

Available compare operations for the IP Filter Type are:

Equal (=): The IP Address must match the one you configured in the Property Value field. Not Equal (!=): The IP Address must not match the one you configured in the Property Value field. Higher (>): The IP Address must be higher than the one you configured in the Property Value field. You can use IP Address Formats like: `192.168.0.10, 192.168.0, 192.168 or even 192`. It depends on what IP Ranges you are going to filter for. Lower (<): The IP Address must be lower than the one you configured in the Property Value field. You can use IP Address Formats like: `192.168.0.10, 192.168.0, 192.168 or even 192`. It depends on what IP Ranges you are going to filter for.

If you want to filter for IP Ranges, I recommend to use two filters to define the range, one filter with the “Higher (>)” compare operation and one with the “Lower (<)” compare operation. This could look like the following:

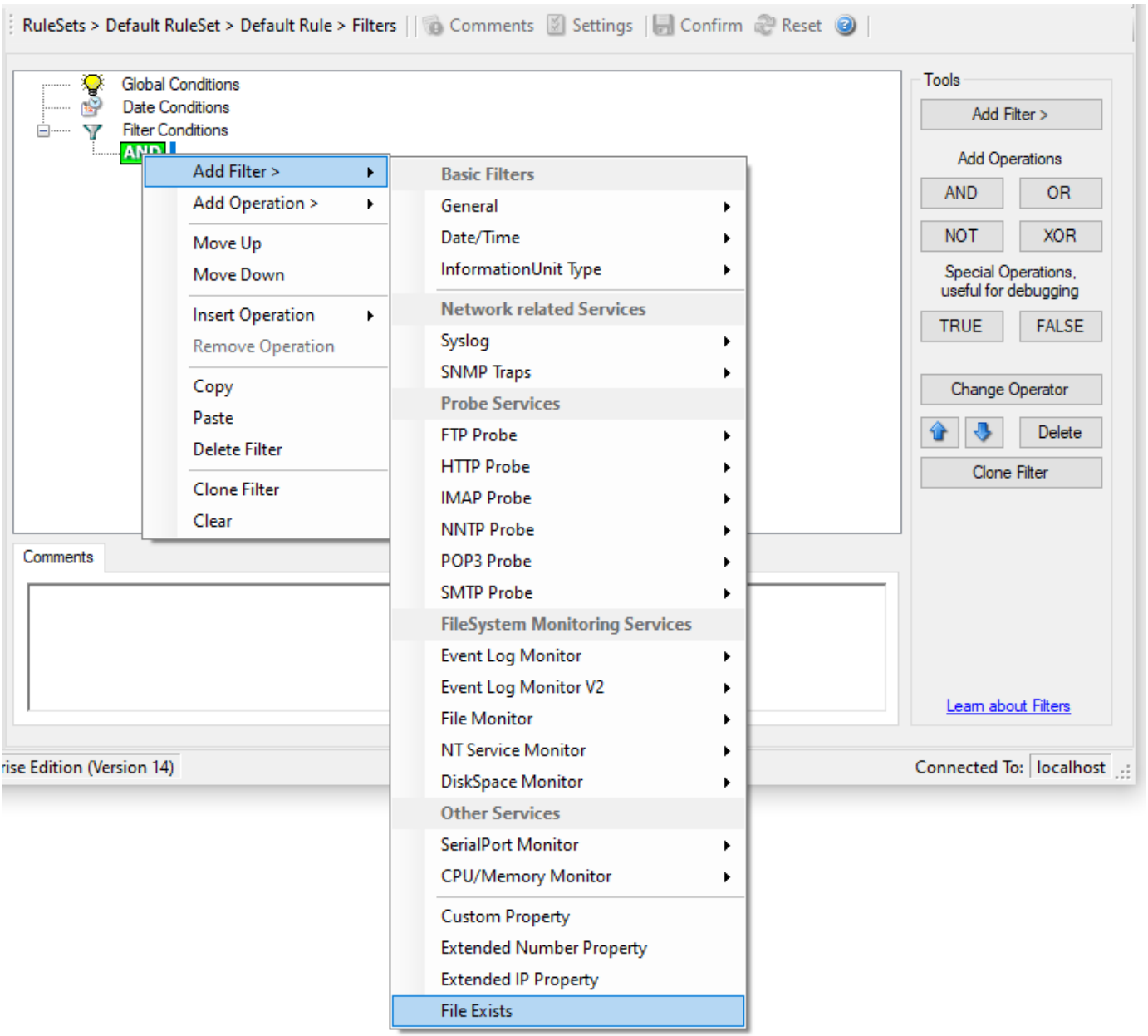


- Filter Conditions - Filtering for an IP Range*

The filter you can see here will accept all IPs which lie between `172.16.0.110` AND `172.16.0.130`. That means, that for every IP that matches these two conditions, the whole filter will evaluate to true and therefore the message will be processed. If the filter does not evaluate to true, the rule will be aborted and the message is sent to the next rule.

File Exists

Filter setting by string.



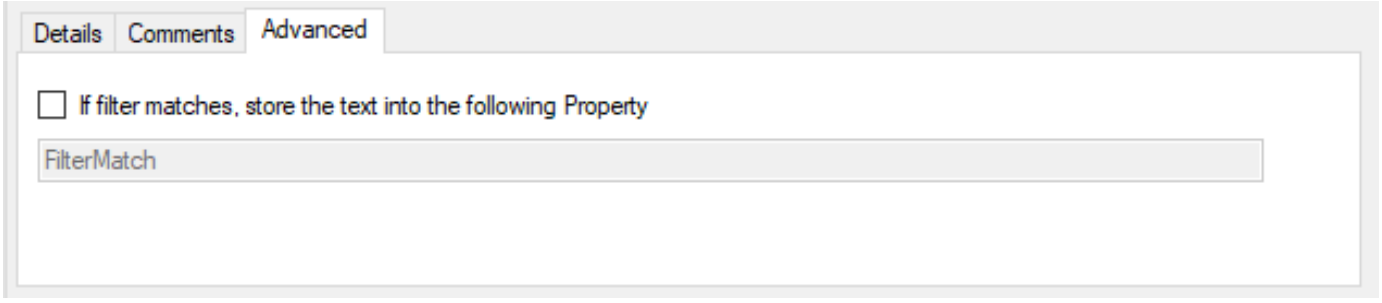
- Filter Conditions - File Exists*

File Exists

With this Filter you can simply check if a file exists or not. You can directly enter the file and its location or you can use the browse-button to find it.

Store Filter Results

How to store Filter Results is described here.



- Filter Conditions - Store Filter Results*

Store Filter Results

If a filter matches, you can now store the result of the match into a custom property.

This custom property can be used in Actions later.

Actions

Actions tell the application that what to do with a given event. With actions, you can forward events to a mail recipient or Syslog server, store it in a file or database, or do many other things with it.

There can be multiple actions for each rule. Actions are processed in the order they are configured. However you can change the order of the actions by moving them Up or Down.

Storing Actions

ODBC Database Options

Use database logging to store messages into a database.

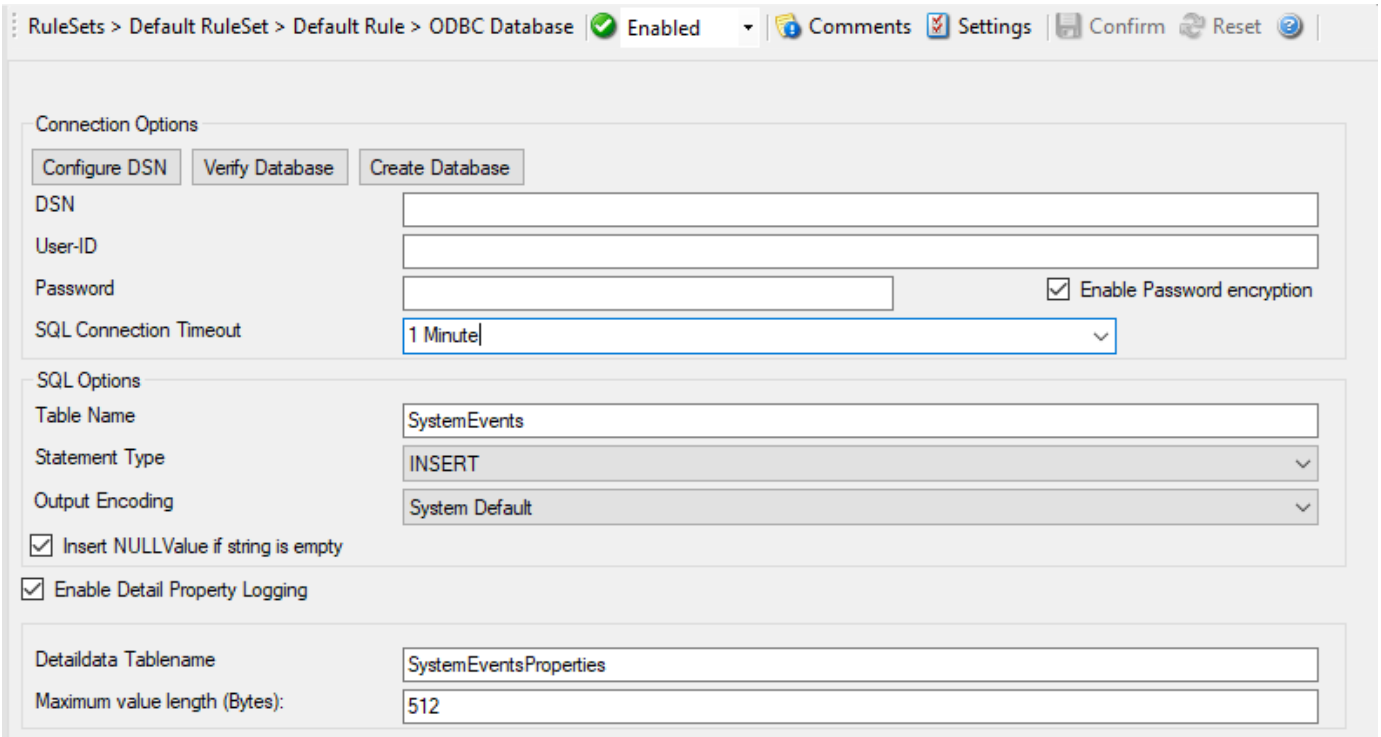
Database logging allows writing incoming events directly to any ODBC - compliant database (virtually any database system currently available for the Windows operating system supports ODBC). We support any database system that provides OLEDB or ODBC drivers. This includes Microsoft JET databases (as used by Microsoft Access), Microsoft SQL Server, MySQL, Oracle, PostgreSQL, Sybase, and many other database systems.

Once stored inside the database, different message viewers as well as custom applications can easily browse them. The defaults for the write database action are suitable Adiscon Loganalyzer (web interface).

The database format can be fine-tuned. This is most useful if you intend to run some additional analysis on the database. Also, in high volume environments, tuning the database action to exactly those fields need helps getting best performance out of the database.

The main feature of the “Write To Database” property sheet is the field list. The default reflects the typical assignment of event properties to database columns. However, you can modify this assignment in any way you like. You only need to keep in mind that Adiscon analysis products need the database contents as specified. As such, malfunctions may occur if you modify the database assignments and then use these tools.

Connection Options



- Action - ODBC Database Connection*

Configure DSN

If you click on this button, it starts the ODBC administrator of the operating system where you can add, edit, or remove a data source(s).

Note: The DSN must be a System DSN. **Verify Database** The configuration client will attempt to establish a database connection to your configured ODBC System DSN.

Create Database

If you click on this button, it will create the default tables for SystemEvents and SystemEventsProperties into the database specified in the DSN.

DNS

File Configuration field:

szODBCDsn

Description:

This is the name of the system data source (DSN - data source name) to be used when connecting to the database. Create this in ODBC manager (can be found in control panel under Windows). Press the “Data Sources (ODBC)” button to start the operating system ODBC administrator where data sources can be added, edited, and removed.

Note: The DSN must be a system DSN, not a user or file DSN. The DSN must be configured to have the correct connection parameters (for example database type and name, server name, authentication mode etc.).

User-ID

File Configuration field:

szODBCUid

Description:

The User-ID used to connect to the database. It is dependent on the database system used if it is to be specified (e.g. Microsoft Access does not need one, while Microsoft SQL Server can force you to use one). If in doubt, please see your database administrator

Password

File Configuration field:

szODBCPwd

Description:

The password used to connect to the database. It must match the “User-ID”. Like the User ID, it is dependent on the database system if a password is needed. Passwords can be stored either encrypted or unencrypted. We highly recommend storing them encrypted.

Enable Encryption

File Configuration field:

nODBCEnCryption

Description:

Check to store the ODBC password encrypted. If left unchecked, the password is stored unencrypted. We strongly recommend checking this box.

If you store the password unencrypted for some reason, please be aware of the security implications. In this case, we recommend using an account with limited access privileges. Even when stored encrypted, we recommend using limited privileges accounts. We are not applying strong cryptography here.

SQL Connection Timeout

File Configuration field:

nSQLConnectionTimeOut

Description:

Defines the Timeout for the connection.

SQL Options

Table Name

File Configuration field:

szTableName

Description:

The name of the table to log to. This name is used to create the SQL insert statement and must match the database definition. Default is “SystemEvents”.

SQL Statement Type

File Configuration field:

nSQLStatementType

Description:

You can select between a INSERT and Call Statement, which is Microsoft specific for Stored Procedures. This means also this type of SQL Statement will only work if MSSQL is used as database. If you select MSSQL Call Statement, the tablename field will automatically be used as stored procedure name.

Output Encoding

File Configuration field:

nOutputEncoding

Description:

This setting is most important for Asian languages. A good rule is to leave it at “System Default” unless you definitely know you need a separate encoding. “System Default” works perfect in the far majority of cases, even on Asian (e.g. Japanese) Windows versions.

Insert NULL Value if string is empty

File Configuration field:

nSQLConnectionTimeOut

Description:

This option inserts a NULL value, if a property is empty.

Enable Detail Property Logging

File Configuration field:

nPropertiesTable

Description:

This option logs event properties other than the standard properties to the SystemEventProperties table. A single event can potentially have multiple properties, so selecting this option can result in multiple writes. With Syslog data, however, there are seldom any additional properties. They most often occur when you use the “Post Process” action to define your own properties. Additional properties are typically found in SETP received data originating from an Event Log Monitor, file monitor, or database monitor (plus other monitors, but these are the most prominent ones).

For example, with Event Log data received via SETP, these properties contain the actually Windows event properties and the event data. Please note that this does not apply to event log messages received via Syslog, because they are no native events but rather Syslog data.

Please make sure you actually need this before activating it. As a side note: some of the MonitorWare Console reports may need detail logging.

Detaildata Tablename

File Configuration field:

szPropertiesTableName

Description:

Tablename for Detail Property Logging

Maximum value length (Bytes)

File Configuration field:

nMaxValueLength

Description:

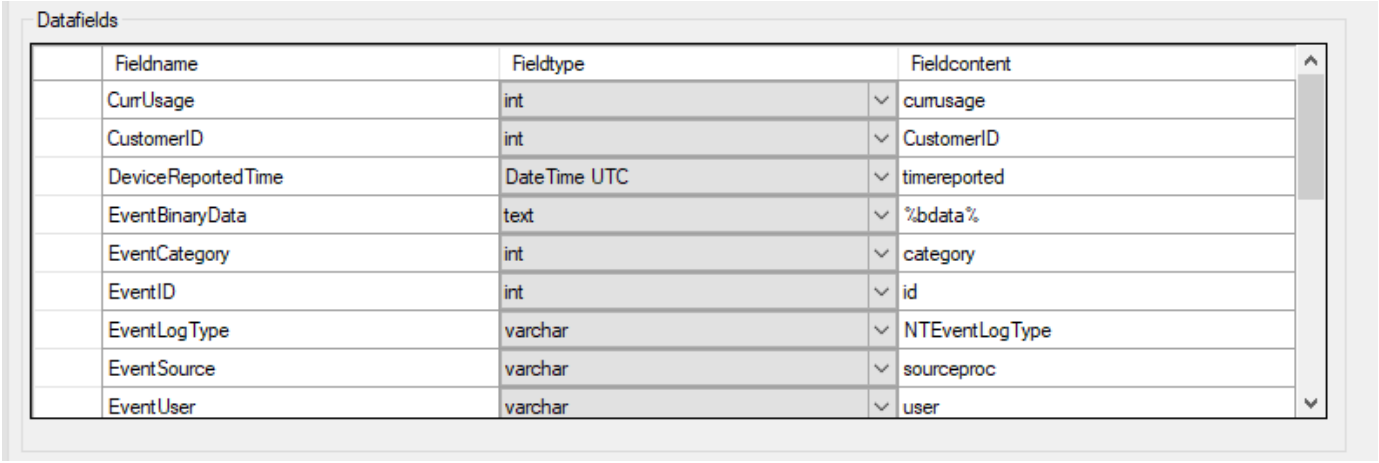
Maximum length in bytes for values stored in Detaildata table.

Datafields

The provided fieldnames are those that Adiscon's schema uses - you can add your own if you have a need for this.

You can edit the field list by selecting a row and then modifying the text fields above the table. You can insert and delete rows by selecting the respective button. If you press delete, the currently selected row is deleted.

For string data types, you can use the property replacer. This can be helpful if you would like to store a substring. For example, if you intend to store only the first 200 characters of each message, you can use `%msg:1:200%`.



- Action - ODBC Database Datafields*

Fieldname

File Configuration field:

szFieldName_[n]

Description:

The Fieldname is the database column name. It can be any field inside the table.

Fieldtype

File Configuration field:

nFieldType_[n]

- 1 = varchar
- 2 = int
- 3 = text
- 4 = DateTime

Description:

Fieldtype is the data type of the database column. It must reflect the column type selected in the database. It must also be consistent in type with the actual property that must be stored. For example, an integer type property like the syslogpriority can be stored in a varchar column. A string data type like the syslogtag can - for obvious reasons - not be stored in an integer column.

Fieldcontent

File Configuration field:

szFieldContent_[n]

Description:

Finally, the Fieldcontent is the event property. For a complete list of supported properties, see event properties

Action Queue Options

- Action - Send RELP Action Queue*

Use Diskqueue if connection to Syslog server fails

File Configuration field:

nUseDiscQueue

Description:

Enable diskqueuing syslog messages after unexpected connection loss.

Split files if this size is reached

File Configuration field:

nDiskQueueMaxFileSize

Description:

Files will be split until they reach the configured size in bytes. The maximum support file size is 10485760 bytes.

Diskqueue Directory

File Configuration field:

szDiskQueueDirectory

Description:

The directory where the queue files will be generated in. The queuefiles will be generated with a dynamic UUID bound to the action configuration.

Waittime between connection tries

File Configuration fields:

nDiskCacheWait

Description:

The minimum waittime until the Syslog Action retries to establish a connection to the Syslog server after failure.

Overrun Prevention Delay (ms)

File Configuration field:

nPreventOverrunDelay

Description:

When the Action is processing syslog cache files, an overrun prevention delay can be added to avoid flooding the target Syslog server.

Double wait time after each retry

File Configuration field:

bCacheWaittimeDoubling

Description:

If enabled, the configured waittime is doubled after each try.

Limit wait time doubling to

File Configuration field:

nCacheWaittimeDoublingTimes

Description:

How often the waittime is doubled after a failed connection try.

Enable random wait time delay

File Configuration field:

bCacheRandomDelay

Description:

If enabled, a some random time will be added into the waittime delay. When using many syslog senders, this can avoid that all senders start sending cached syslog data to the Syslog server at the same time.

Maximum random delay

File Configuration field:

nCacheRandomDelayTime

Description:

Maximum random delay time that will be added to the configured waittime if Enable random wait time delay is enabled.

OLEDB Database Action

Due the changes to x64, it became more important to also support the newer database layer from Microsoft called OLEDB. The OLEDB Action works similar to the ODBC Action from configuration point of few. The MS SQL OLEDB Provider and JET4.0 OLEDB Provider have been successfully tested in the Win32 environment. Unfortunately, the JET4.0 Provider has not been ported to the x64 platform yet. In our internal performance tests, there was an enhancement of up to 30% compared to ODBC. So this action may also be interesting for people with a huge amount of incoming data.

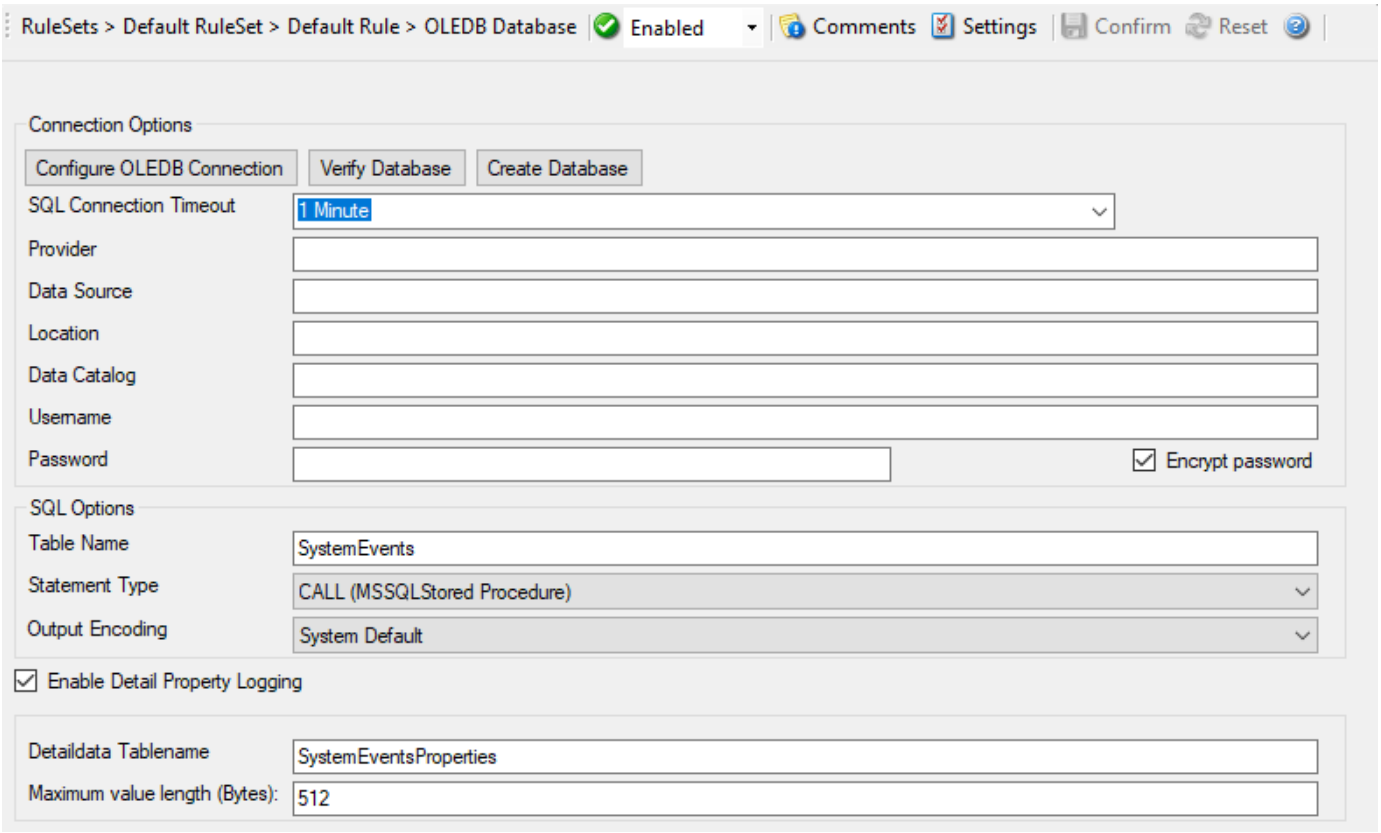
This Action allows writing incoming events directly to any OLEDB - compliant database.

Once stored inside the database, different message viewers as well as custom applications can easily browse them. The defaults for the write database action are suitable for Adiscon Loganalyzer (web interface).

The database format can be fine-tuned. This is most useful if you intend to run some additional analysis on the database. Also, in high volume environments, tuning the database action to exactly those fields need helps getting best performance out of the database.

The main feature of the “OLEDB Database Action” property sheet is the field list. The default reflects the typical assignment of event properties to database columns. However, you can modify this assignment in any way you like.

Connection Options



- Action - OLEDB Database Connection*

Configure OLEDB Connection

If you click on this button, it starts an OLEDB configuration wizard that will

help you configuring your OLEDB data source.

Verify Database

The configuration client will attempt to establish a database connection to your configured OLEDB Connection.

Create Database

If you click on this button, the configuration client will create the default tables for SystemEvents and SystemEventsProperties into your configured OLEDB database.

SQL Connection Timeout

File Configuration field:

nSQLConnectionTimeOut

Description:

Defines the Timeout for the connection

Provider

File Configuration field:

szProvider

Description:

OleDB Provider like SQL Server Client (SQLNCLI11.1). Should be filled automatically with Configure OLEDB Connection button.

Data Source

File Configuration field:

szDataSource

Description:

Data source is most often the server name or IP address like SERVERNAMESQLEXPRESS for example. Should be filled automatically with Configure OLEDB Connection button.

Location

File Configuration field:

szLocation

Description:

OLEDB Location. Should be filled automatically with Configure OLEDB Connection button.

Data Catalog

File Configuration field:

szDataCatalog

Description:

Is the database name in most cases. Should be filled automatically with Configure OLEDB Connection button.

Username

File Configuration field:

szUsername

Description:

Username used for authentication. Should be filled automatically with Configure OLEDB Connection button.

Password

File Configuration field:

szPassword

Description:

Password used for authentication. Should be filled automatically with Configure OLEDB Connection button.

Encrypt password

File Configuration field:

szPassword

Description:

Password used for authentication. Should be filled automatically with Configure OLEDB Connection button.

Table Name

File Configuration field:

szTableName

Description:

The name of the table to log to. This name is used to create the SQL insert statement and must match the database definition. Default is “SystemEvents”.

Please note that the default table name must be used when other members of the MonitorWare family (like the web interface or the MonitorWare Console) should work with the database. This customization option is meant for those customers that use third-party or custom software.

Statement Type

File Configuration field:

nSQLStatementType

Description:

You can select between a INSERT and Call Statement, which is Microsoft specific for Stored Procedures. This means also this type of SQL Statement will only work if MSSQL is used as database. If you select MSSQL Call Statement, the tablename field will automatically be used as stored procedure name.

Output Encoding

File Configuration field:

nOutputEncoding

Description:

This setting is most important for Asian languages. A good rule is to leave it at “System Default” unless you definitely know you need a separate encoding. “System Default” works perfect in the far majority of cases, even on Asian (e.g. Japanese) Windows versions.

Enable Detail Property Logging

File Configuration field:

nPropertiesTable

Description:

This option logs event properties other than the standard properties to the SystemEventProperties table. A single event can potentially have multiple properties, so selecting this option can result in multiple writes. With Syslog data, however, there are seldom any additional properties. They most often occur when you use the “Post Process” action to define your own properties. Additional properties are typically found in SETP received data originating from an Event Log Monitor, file monitor, or database monitor (plus other monitors, but these are the most prominent ones).

For example, with Event Log data received via SETP, these properties contain the actually Windows event properties and the event data. Please note that this does not apply to event log messages received via Syslog, because they are no native events but rather Syslog data.

Please make sure you actually need this before activating it. As a side note, some of the MonitorWare Console reports may need detail logging.

Detaildata Tablename

File Configuration field:

szPropertiesTableName

Description:

Tablename for Detail Property Logging

Maximum value length (Bytes)

File Configuration field:

nMaxValueLength

Description:

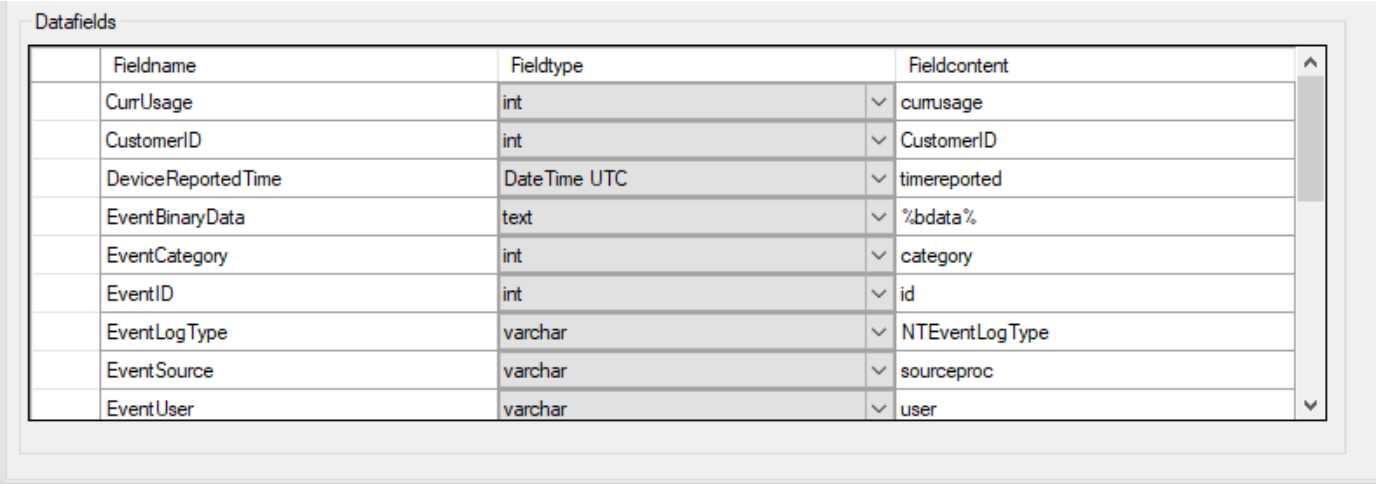
Maximum length in bytes for values stored in Detaildata table.

Datafields

The provided fieldnames are those that Adiscon’s schema uses - you can add your own if you have a need for this.

You can edit the field list by selecting a row and then modifying the text fields above the table. You can insert and delete rows by selecting the respective button. If you press delete, the currently selected row is deleted.

For string data types, you can use the property replacer. This can be helpful if you would like to store a substring. For example, if you intend to store only the first 200 characters of each message, you can use “%msg:1:200%”.



- Action - OLEDB Database Datafields*

Fieldname

File Configuration field:

szFieldName_[n]

Description:

The Fieldname is the database column name. It can be any field inside the table.

Fieldtype

File Configuration field:

nFieldType_[n]

- 1 = varchar
- 2 = int
- 3 = text
- 4 = DateTime

Description:

Fieldtype is the data type of the database column. It must reflect the column type selected in the database. It must also be consistent in type with the actual property that must be stored. For example, an integer type property like the syslogpriority can be stored in a varchar column. A string data type like the syslogtag can - for obvious reasons - not be stored in an integer column.

Fieldcontent

File Configuration field:

szFieldContent_[n]

Description:

Finally, the Fieldcontent is the event property. For a complete list of supported properties, see event properties

Action Queue Options

- Action - Send RELP Action Queue*

Use Diskqueue if connection to Syslog server fails

File Configuration field:

nUseDiscQueue

Description:

Enable diskqueuing syslog messages after unexpected connection loss.

Split files if this size is reached

File Configuration field:

nDiskQueueMaxFileSize

Description:

Files will be split until they reach the configured size in bytes. The maximum support file size is 10485760 bytes.

Diskqueue Directory

File Configuration field:

szDiskQueueDirectory

Description:

The directory where the queue files will be generated in. The queuefiles will be generated with a dynamic UUID bound to the action configuration.

Waittime between connection tries

File Configuration fields:

nDiskCacheWait

Description:

The minimum waittime until the Syslog Action retries to establish a connection to the Syslog server after failure.

Overrun Prevention Delay (ms)

File Configuration field:

nPreventOverrunDelay

Description:

When the Action is processing syslog cache files, an overrun prevention delay can be added to avoid flooding the target Syslog server.

Double wait time after each retry

File Configuration field:

bCacheWaittimeDoubling

Description:

If enabled, the configured waittime is doubled after each try.

Limit wait time doubling to

File Configuration field:

nCacheWaittimeDoublingTimes

Description:

How often the waittime is doubled after a failed connection try.

Enable random wait time delay

File Configuration field:

bCacheRandomDelay

Description:

If enabled, a some random time will be added into the waittime delay. When using many syslog senders, this can avoid that all senders start sending cached syslog data to the Syslog server at the same time.

Maximum random delay

File Configuration field:

nCacheRandomDelayTime

Description:

Maximum random delay time that will be added to the configured waittime if Enable random wait time delay is enabled.

File Logging Options

This configuration dialog is available both in the defaults section as well as with file logging actions.

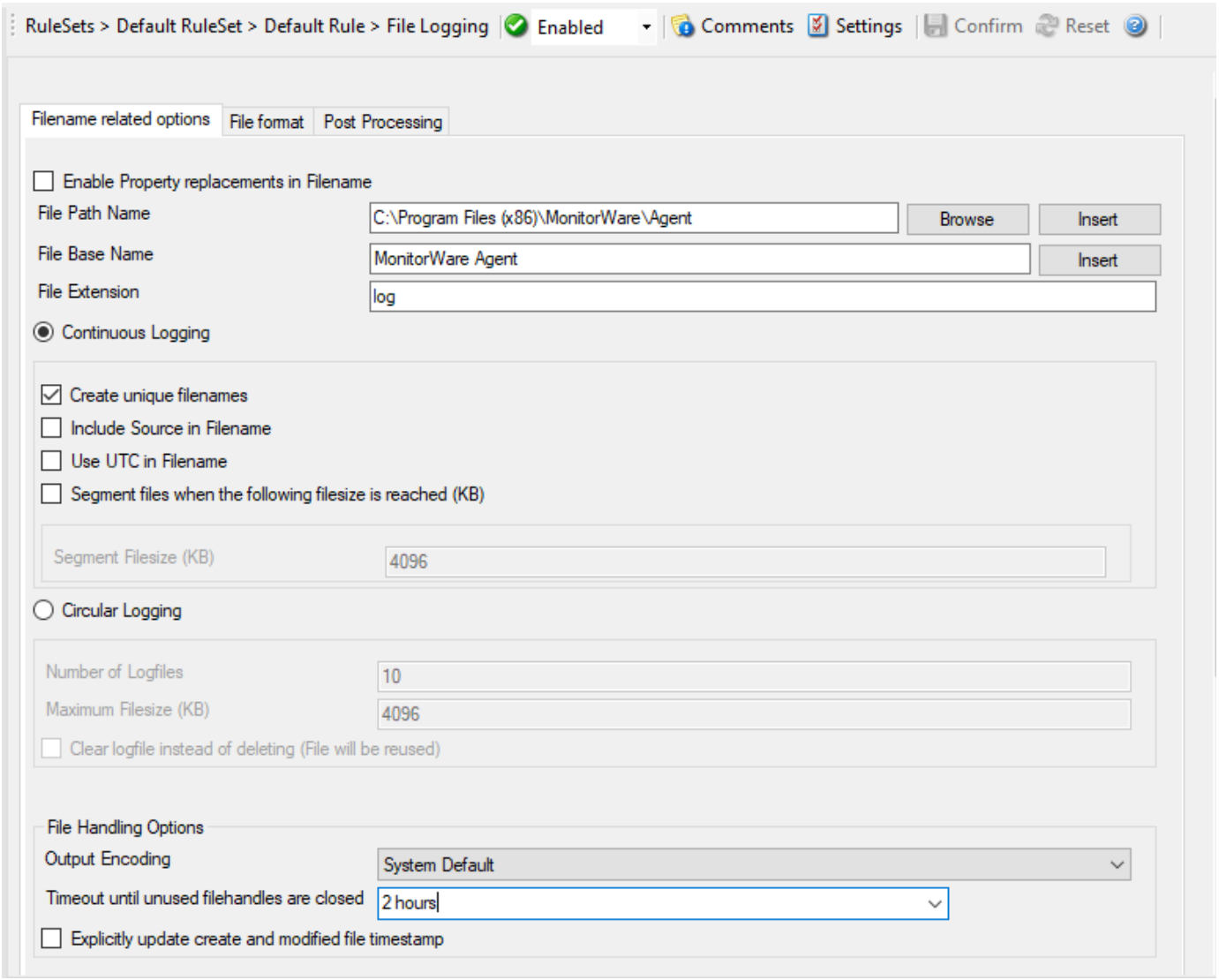
File logging is used to write text files of received messages. One file per day is written. New entries are appended to the end of the file.

File locks are released when currently no data is written. Therefore, other applications can access the files while the service is running. However, please be sure that the other applications do not place a file-lock onto it. Popular WordPad does so. In this case, the service will not be able to log any further messages (an error event is written to the Windows Event Log in this case). We recommend copying the file when accessing it at runtime - or use notepad.exe, which does not place file-locks on the files it opens.

The filename is build as follows:

`<FilePathName><FileBaseName>-year-month-day.<FileExtension>`

Parameters in the brackets can be configured via dialog shown below:



- Action - File Logging Filename related*

Enable Property replacements in Filename

File Configuration field:

nEnablePropertyFileName

Description:

By activating this option, you can use properties within the file or pathname like `%source%` and all the others. For example: File Path Name can be `F:\syslogs\%source%` File Base Name can be `IIS-%source%`

If your source is `10.0.0.1`, that writes the following file: `F:\syslogs\10.0.0.1\IIS-10.0.0.1.log`

The path `f:\syslogs\10.0.0.1` was generated because the source property was used inside the path.

Please Note that you can use ANY property inside the path and base name. event properties are described in the property replacer section.

File Path Name

File Configuration field:

szFilePath

Description:

The base path (directory) of the file. Please see above for exact placement. Default is `c:\temp`. The Insert Menu entry allows you to create “Dynamic Directories”. For example:

File Path Name can be ``**F:syslogs%source%**``

event properties are described in the property replacer section.

On network paths: The File Logging action can also work on network storages. There are two ways of storing log files in a network path.

1. Direct the action to a full UNC path. In this case, make sure the system account with which the service is running is able to access the network path or the service will fail to access with a permission error. Sample path: `\Hostname\folder1\folder2\`
2. Map the UNC path to a local drive letter in Windows. In this case, the path will look like a regular local path, but actually points to a network location. This requires a workaround, which is to run a scheduled task at system startup under Local System and perform a net use specifying the user and password of the share. Else, the service will not be able to access the mapped UNC path, because the mapping usually happens for interactive sessions only.

File Base Name

File Configuration field:

szFileBaseName

Description:

The base name of the file. Please see above for exact placement. Default is “MonitorWare”. The Insert Menu entry allows you to recreate “Dynamic Base Filenames”. For example:

File Base Name can be `IIS-%source%`

File Extension

File Configuration fields:

szFileExtension

Description:

The extension to be used when writing the file. Please see above for exact placement. Default is `.log`.

Continuous Logging

Description

When enabled log files will not be overwritten, there is a single file with consistent file name. See below checkboxes to choose in which cases a new file should be created.

Create unique Filenames

File Configuration field:

nUniqueFileName

Description:

If checked, a unique file name is created for each day. This is done by adding the current date to the base name.

If left unchecked, the date is not added and as such, there is a single file with consistent file name. Some customers that have custom scripts to look at the file name use this.

Include Source in Filename

File Configuration field:

nIncludeSourceInFilename

Description:

This works together with the “Create unique Filenames” setting. If checked, the file name generation explained above is modified. The source of the Syslog message is automatically added to the file name.

This feature has been introduced because many customers would like to have separate log files for each device. While this can be achieved with multiple rules, it is much more straight forward with this single checkbox. If it is checked, the messages are automatically written to separate files and the file name includes the originating device information.

Use UTC in Filename

File Configuration field:

nUseUTCInFileName

Description:

This works together with the “Create unique Filenames” setting. If unique names are to be created then select the “Use UTC in Filename” option, in this case the file name is generated on the basis of universal coordinated time (UTC) or on local time. UTC was formerly referred to as “GMT” and is the basis of the time zone system. For example, New York, USA is 5 hours behind UTC. Therefore, if it is 12 noon in New York, the UTC time is 5pm.

When it comes to log file creation, it means that the date is computed on UTC. Taking the same example, if the “Use UTC in Filename” is checked, the log file name would roll over to the next date at 7 pm New York time. If it were unchecked, the rollover would occur exactly at midnight New York time (5 am UTC).

Using UTC for file name creation can be helpful if log files are written among different time zones and later consolidated. Using UTC ensures a consistent time notation across all log files.

Please note that this setting does affect the file name creation only. A different setting controls the dates recorded inside the file.

Segment files when the following file size is reached (KB)

File Configuration field:

nSegmentFileEnable

Description:

Files are segmented if the defined file size: Segment Filesize (KB) is reached. A sequence number is appended to the file name: _1 to _n.

Circular Logging

File Configuration field:

nCircularLogging

Description:

If enabled, log files are created and overwritten in a cycle.

Number of Log Files

File Configuration field:

nNumberOfLogfiles

Description:

Once the last log file is reached, circular logging begins and overwrites the first log file again. If set to 0, log files will not be rotated but can still be processed by Rotate Post Processing (for example compression or backup) along with the Rotate Conditions.

Maximum Filesize (KB)

File Configuration field:

nMaxFileSize

Description:

Max filesize of a log file, once this size is reached a new logfile is created.

Clear logfile instead of deleting (File will be reused)

File Configuration field:

nReUseFile

Description:

This option causes the File Action to truncate the log file instead of deleting and recreating it.

File Handling Options

Output Encoding

File Configuration field:

nOutputEncoding

Description:

This setting is most important for Asian languages. A good rule is to leave it at “System Default” unless you definitely know you need a separate encoding. “System Default” works perfectly in the far majority of cases, even on Asian (e.g. Japanese) Windows versions.

Timeout until unused filehandles are closed

File Configuration field:

nCleanFileHandlesTimeout

Description:

When dynamic filenames are used, filehandles are cached internally to avoid massive amount of File open/close operations. This timeout specifies after which time handles should be finally closed if not used anymore. Each write to a file will reset the timeout counter for the current filehandle.

Explicitly update create and modified file Timestamp

File Configuration field:

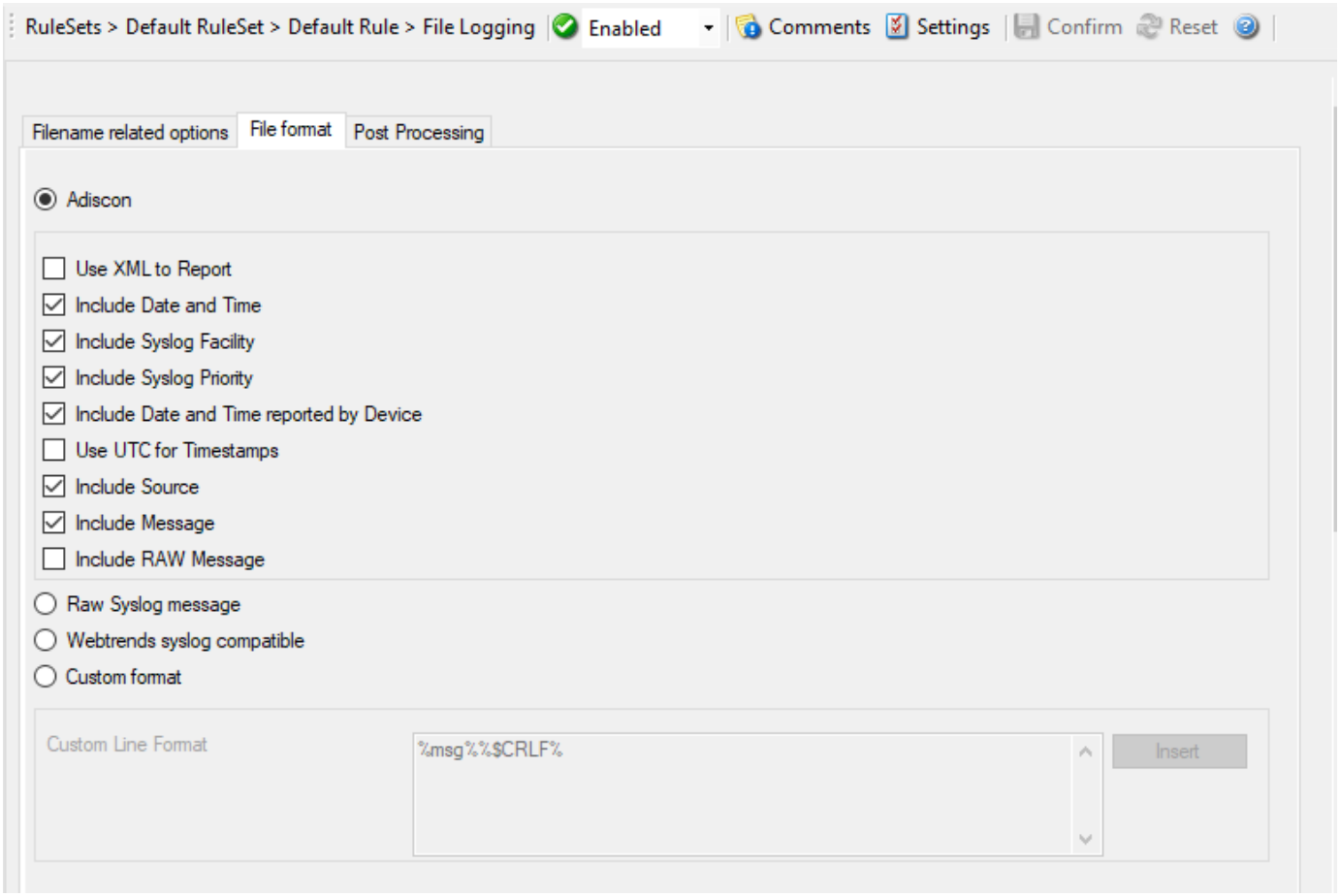
nEnableUpdateFileTime

Description:

If the checkbox is not selected the operating system updates the timestamps for creating and modifying files. In cases where the filesystem does not do this reliably, the checkbox can be selected. Now the service itself updates the timestamps for creating and modifying files.

File Format

The format in which the log file is written can be selected here. The default is “Adiscon”, which offers most options. Other formats are available to increase log file compatibility to third party applications.



- Action - File Logging File Format*

Adiscon

> **Note**
>
> Any other format besides “Adiscon Default” are fixed formats. As such, if it is selected, all other formatting options do not apply and consequently are turned off.

The following options are possible:

Use XML to Report

File Configuration field:

nUseXMLtoReport

Description:

If checked, the message part includes a complete XML-formatted information record. It includes additional information like timestamps, syslog facility and priority, and others in an easy to parse format. If XML output format is selected, you might consider turning all other information fields off, as they are already included in the XML stream. However, this is not a requirement.

Use UTC for Timestamps

File Configuration field:

nUseUTCForTimestamps

Description:

Please see the definition of utc above at “Use UTC in Filename”. This setting is very similar. If checked, all time stamps are written in UTC. If unchecked, local time is used instead. Again, UTC is useful if logs written in multiple time zones are to be consolidated.

Include <Fieldname>

File Configuration field:

- nFileDateTime
- nFileFacility
- nFilePriority
- nFileDateTimeReported
- nFileSource
- nIncludeMessage
- nIncludeRAWMessage

Description:

The various “include” settings controls are used to specify the fields which are to be written to the log file. All fields except the message part itself are optional. If a field is checked, it is written to the log file. If unchecked, it will not be written. All fields are comma-delimited.

Please note the difference between the “Date and Time” and “Date and Time reported by Device”. Both are timestamps. Either both are written in local time or utc based on the “Use UTC for Timestamps” check box. However, “Date and Time” is the time when MonitorWare Agent received the message. Therefore, it is always a consistent value.

In contrast, the “Date and Time Reported by Device” is a timestamp taken from the actual message. As such, it is dependent on the reporting device clock, which might be off. In addition, in the case of Syslog messages, there is no time zone information within the device reported timestamp. As such, if devices from multiple time zones are reporting, the timestamp information is not consistent. This is due to Syslog design as of rfc 3164. The Syslog server can be configured to ignore the RFC in this case and provide a consistent time stamp. However, from the view of the log file writer, the “Date and Time Reported by Device” might not be as trustworthy as the “Date and Time” field. Nevertheless, it might also be more useful than the former one. This is the reason both timestamps are present and can individually be selected.

The “Include Message” and “Include RAW Message” fields allow customizing the message part that is being written. The raw message is the message as – totally unmodified, was received. This might be useful if a third party application is expecting raw Syslog entries. The message itself is just that part of the Syslog message that is being parsed as message. That is without e.g. host information or a tag value. Please note that we recommend selecting only one of these options, as otherwise two message fields are written. Similarly, if none is selected no message is written at all. Please note that we support these configurations, too – there might be a legitimate need for them.

Raw Syslog message

The “Raw Syslog message” format writes raw Syslog format to the log file. That is, each line contains the Syslog message as of RFC 3164. No specific field processing or information adding is done. Some third party applications require that format.

Webtrends syslog compatible

The “WebTrends Syslog compatible” mimics the format that WebTrends applications expect. Please note that we only mimic the log file format. It is still the job of the reporting device (most notable firewall) to generate the correct WebTrends WELF format. The “WebTrends” format is supported because many customers would like to use MonitorWare Agent 3.0 enhanced features while still having the ability to work with WebTrends.

Custom format

The “Custom format” allows you to customize formats to increase log file compatibility for third party applications. When you choose this option then Custom line format is enabled.

Custom Line Format

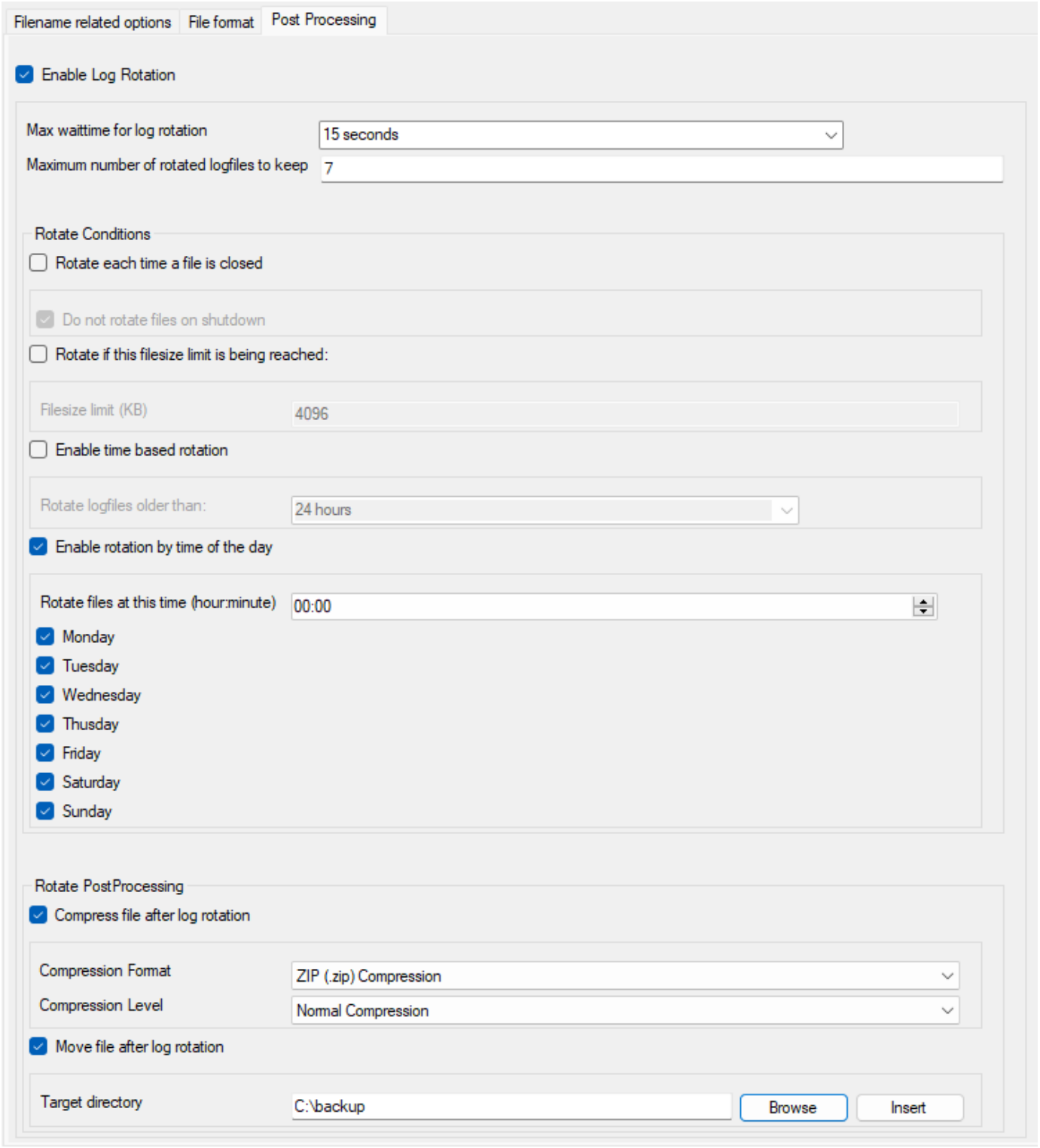
File Configuration field:

szLineFormat

Description:

Custom Line Format enables you to fully customize the output for the log file. The Insert Menu entry provides further options and they only work in custom line format. Default value is `%msg%%$CRLF%`.

Post Processing



- Action - File Logging Post Processing*

Enable Log Rotation

File Configuration field:

nCircularLogging

Description:

When enabled log files are created and over written in a cycle.

Maximum wait time for log rotation

File Configuration field:

nLogRotateMaxWait

Description:

Maximum Wait time when log rotation is processed within the Queue Engine.

Maximum number of rotated log files to keep

File Configuration field:

nNumberOfLogfiles

Description:

Once the last log file is reached, circular logging begins and overwrites the first log file again. If set to 0, log files will not be rotated but can still be processed by Rotate Post Processing (for example compression or backup) along with the Rotate Conditions.

Rotate Conditions

Rotate each time a file is closed

File Configuration field:

nLogRotateOnClose

Description:

When a file is closed (Timeout for example), log rotation will be done.

Do not rotate files on Shutdown

File Configuration field:

nLogDoNotRotateOnShutdown

Description:

Do not rotate log files if service is stopped even with “Rotate each time a file is closed” enabled.

Rotate if this filesize limit is being reached

File Configuration field:

nLogRotateOnSizeLimit

Description:

Enable log rotation if a configured file size is reached.

Filesize limit (KB)

File Configuration field:

nLogRotateSizeLimit

Description:

The actual file size in KB for “Rotate if this filesize limit is being reached”.

Enable time based rotation

File Configuration field:

nLogEnableRotateTimeout

Description:

Enable time based log rotation.

Rotate log files older than

File Configuration field:

nLogRotateTimeout

Description:

Sets the maximum file age before a logfile is being rotated when “Enable time based rotation” is enabled.

Enable rotation by time of the day

File Configuration field:

nLogEnableRotateTimeOfTheDay

Description:

Rotate this file at this time (hour:minute) and the checked day/days.

Rotate PostProcessing

Compress File After log rotation

File Configuration field:

nLogZipAfterRotate

Description:

Enable file compression after log rotation.

Compression Format

File Configuration field:

nLogZipAfterRotateFormat

Description:

It is possible to compress to ZIP or GZIP format.

Compression Level

File Configuration field:

nLogZipCompressionLevel

Description:

There are different levels that can be selected:

- Best Speed
- Low Compression
- Normal Compression
- Best Compression

Move file after log rotation

File Configuration field:

nLogMoveAfterRotate

Description:

Move logfile after rotation & compression.

Target directory

File Configuration field:

szLogMoveAfterRotatePath

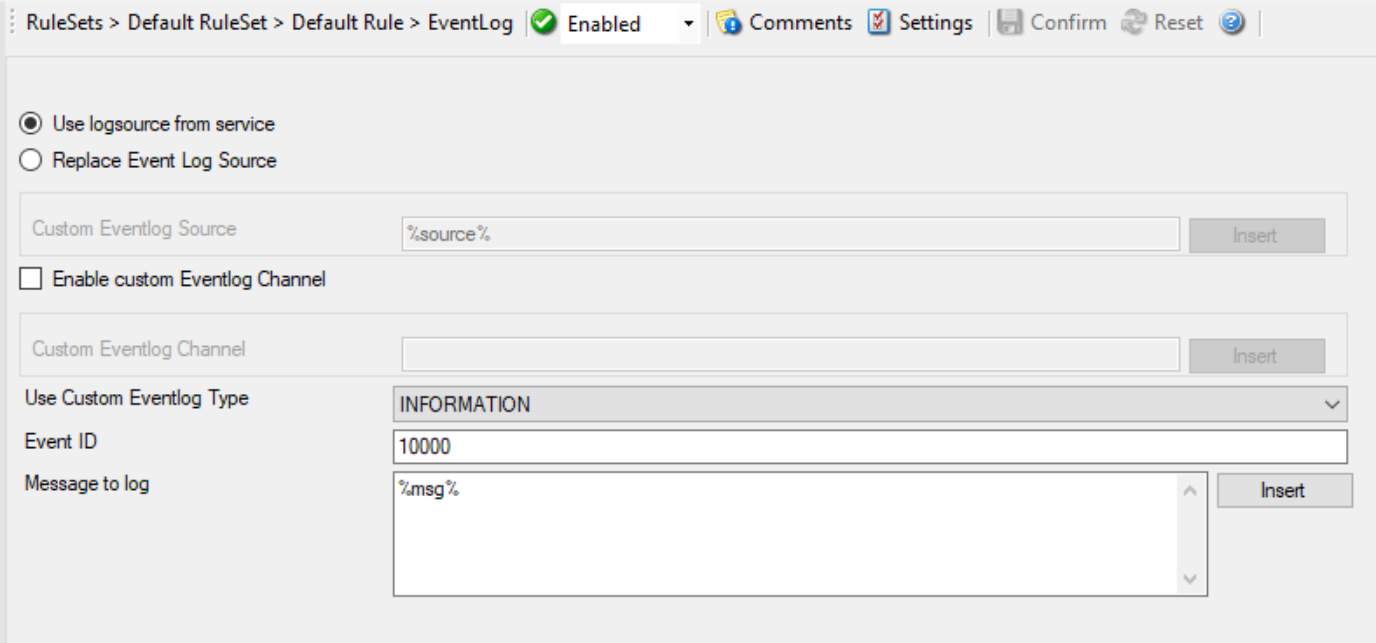
Description:

Location where to move the logfile after rotation & compression.

forwarding actions

Event Log Options

This tab is used to configure the logging to the Windows Event Log. It is primarily included for legacy purposes.



- Action - EventLog*

Use logsource from service

File Configuration field:

bUseCustomEventLog = 0

Description:

Takes the service name as logsource for the log entry. This option is enabled by default.

Replace Event Log Source

File Configuration field:

bUseCustomEventLog = 1

Description:

If checked, a special mapping mechanism is activated. In this mode, the Windows event source is set to the IP address of the system sending the Syslog message. In addition, the ID is set to syslog facility. This mode helps to quickly gather information about the system state in Windows event viewer.

However, this mode has its drawbacks. Effectively, we are writing invalid event source information to the event log. This does not harm any application, but Windows event viewer will try to locate the matching message libraries. Of course, this is impossible. As such, event viewer will warn the user that the message library could not be found. Nevertheless, it will display the complete logged message. This happens only in detail view.

Users should fully understand the implications of this mapping mechanism for their environment before turning this option on.

Custom Event Log Source

File Configuration field:

szCustomSource

Description:

EventSource is now fully configurable with all possibilities the property engine gives you. Please note that content of this field can be configured. event properties are described in the property replacer section.

Enable custom Eventlog Channel

File Configuration field:

bUseCustomEventLog

Description:

If enabled, a custom event log channel will be used instead of application.

Custom Eventlog Channel

File Configuration field:

szCustomEventLog

Description:

The custom Eventlog channel to be used instead of application. Will be automatically created if the channel does not exist.

Use Custom Eventlog Type

File Configuration field:

nEventType

- 0 = EVENTLOG_SUCCESS (Information event)
- 1 = EVENTLOG_ERROR_TYPE (Error event)
- 2 = EVENTLOG_WARNING_TYPE (Warning event)
- 4 = EVENTLOG_INFORMATION_TYPE (Information event)
- 8 = EVENTLOG_AUDIT_SUCCESS (Success Audit event)
- 16 = EVENTLOG_AUDIT_FAILURE (Failure Audit event)

Description:

The type – or severity – this log entry is written with. Select from the available Windows system values.

EventID

File Configuration field:

nEventID

Description:

The ID to be used when writing to the event log. Different IDs can be used to provide other processes with a consistent interface to specific messages. WinSyslog does not restrict the IDs that can be used. However, if an ID is written that is not registered with the operating system, Windows event viewer places an error message pointing this out before the actual message text. To avoid this text, event IDs 10,000 to 10,100 have been registered with the OS. We highly recommend that these IDs should be used for all custom messages. IDs below 10,000 should not be used as they might potentially interfere with events generated by MonitorWare Agent 3.0 itself.

Message to Log

File Configuration field:

szMessagecontent

Description:

It is the message which will be logged into the Windows Event Log. It is fully configurable what is logged into the Eventlog.

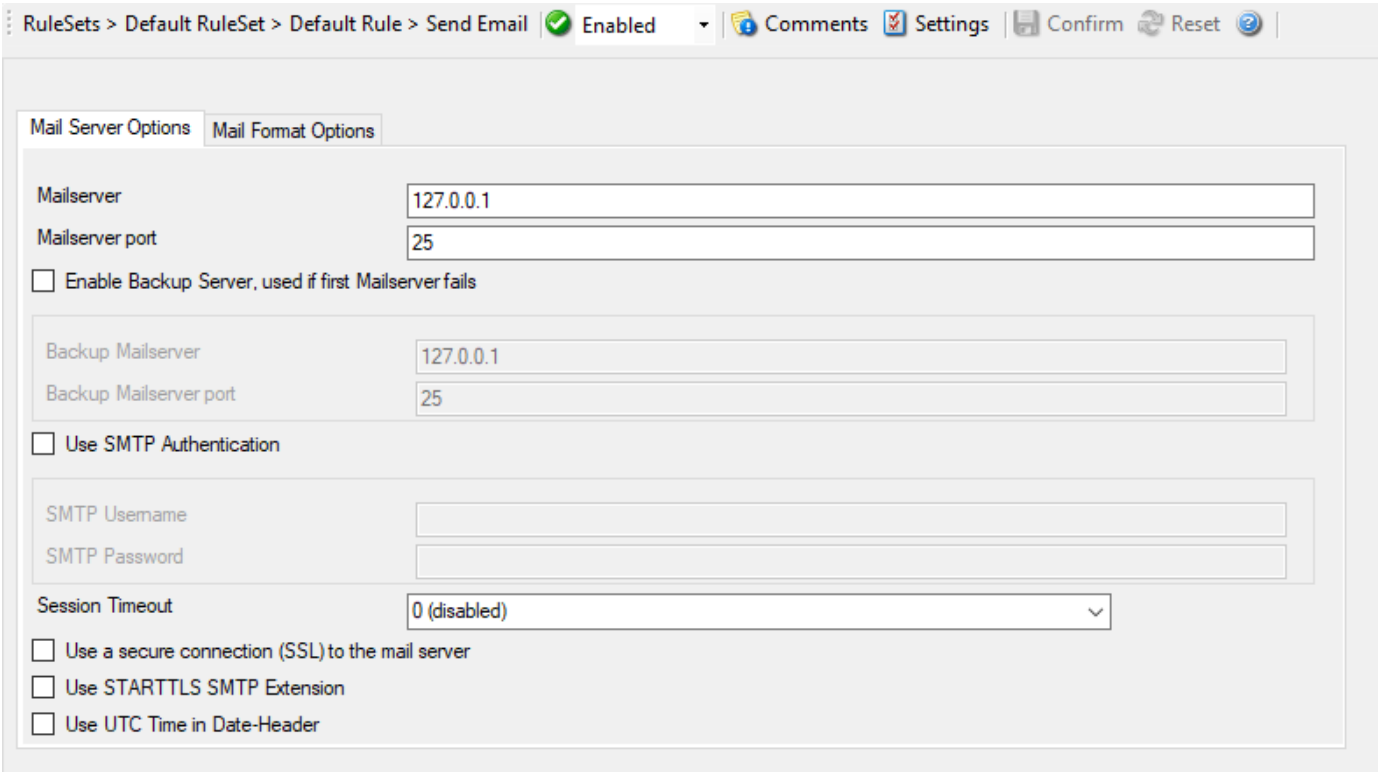
Insert Menu entry allows you to add replacement characters e.g. ``%msg%`` - you can write the actual message of an event into the Windows Event Log.

Please note that the message content of the message field can be configured. event properties are described in the property replacer section.

Send Email

This tab is used to configure mail (SMTP) parameters. These are the basic parameters for email forwarding. They need to be configured correctly, if mail message should be sent by the service.

Mail Server Options



- Action - Send Email - Mail Server Options*

Mailserver

File Configuration field:

szSMTPServer

Description:

This is the Name or IP address of the mail server to be used for forwarding messages. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address. Please note that this server must be able to relay messages if the recipient is not hosted at this server. Be sure to contact your mail server's administrator if in doubt on this issue.

The service expects to talk to a standard SMTP mail server. Message relaying to the final destination must be permitted.

Mailserver port

File Configuration field:

nSMTPPort

Description:

Port the mail server is to be contacted at. Usually, this is 25. It might, however, be changed in your system. Then, specify the port your mail server uses. If in doubt, try the default of 25 - or contact your mail server administrator.

Enable Backup Server, used if first Mailserver fails

File Configuration field:

nEnableBackupServer

Description:

When enabled, you can configure a second Mailserver that will be used if the regular Mailserver is not available/accessible.

Backup Mailserver

File Configuration field:

szSMTPServerBackup

Description:

In case that the connection to the main configured mail server cannot be established, the backup mail server is tried. Note that an error is only generated, if the connection to the backup server fails as well.

Backup Mailserver port

File Configuration field:

nSMTPPortBackup

Description:

Port the mail server is to be contacted at. Usually, this is 25. It might, however, be changed in your system. Then, specify the port your mail server uses. If in doubt, try the default of 25 - or contact your mail server administrator.

Use SMTP Authentication

File Configuration field:

nUseSMTPAuth

Description:

Check this box if your server requires SMTP authentication. To fight SPAM, more and more server operators allow relaying only for authenticated users. It might even happen that an existing account does no longer work because the server has been reconfigured to disallow anonymous posting.

If your server requires (or supports) SMTP authentication, check this box and enter your User ID and password in the boxes below. The exact values will be provided by your server operator – if in doubt, please ask the mail server support.

If the mail server does not support authentication, leave this box unchecked.

We recommend using authentication if it is available. Even when the current server configuration allows unauthenticated relay, this potentially will change in the future (as the SPAM problem grows). If you already use authentication, such a server configuration change will not affect you. Otherwise, it will disrupt mail service.

Session Timeout

File Configuration field:

nTimeoutValue

Description:

This option controls if multiple rapidly incoming messages should be combined to a single email message. The SMTP session with the server is held open for the specified timeout period. Please note that the period is specified in milliseconds, not seconds.

If a new event arrives within the specified timeout period, that event will be included in the same email message as the previous one. Then, the timeout is re-started. As such, any events coming in within successive timeout periods will be combined in a single mail.

This is most appropriate when large burst of messages are expected and these should be combined in few mail messages. Otherwise, multiple mail messages can easily overflow the administrator's mailbox.

The session timeout is user configurable between 1 and 2147483647 milliseconds (32bit integer) or different pre-set values. Larger values are not supported as they probably affect the SMTP server performance and can lead to unpredictable results.

The session timeout of zero milliseconds has a special meaning: if it is selected, every event will be sent in a separate message, no matter how fast two messages occur after each other.

Use a secure connection (SSL) to the mail server

File Configuration field:

nUseSSL

Description:

This option enables SSL-secured traffic to the mail server. Please note, that this only works, if the receiving mail server supports SSL-secured transmission of emails.

Use STARTTLS SMTP Extension

File Configuration field:

nUseUTCTimeStamp

Description:

Some Email Readers do not support UTC time in date-headers. Therefore here is a switch to turn the UTC time on or off.

Use UTC Time in Date-Header

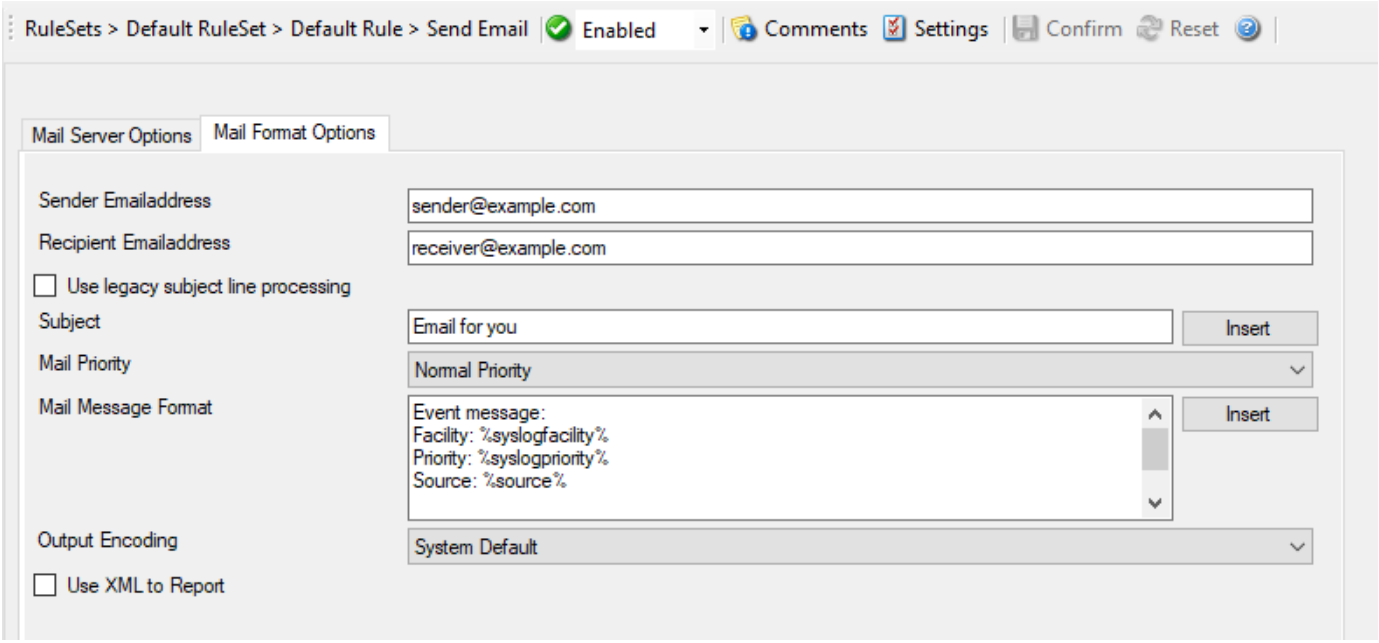
File Configuration field:

nUseUTCTimeStamp

Description:

Some Email Readers do not support UTC time in date-headers. Therefore here is a switch to turn the UTC time on or off.

Mail Format Options



- Action - Send Email - Mail Format Options*

Sender email address

File Configuration field:

szSMTPSender

Description:

Email address used as the sender address for outgoing messages. In order for your SMTP server to accept it, it probably must be a valid address.

Recipient email address

File Configuration field:

szSMTPRecipient

Description:

The recipient emails are addressed to. To send a message to multiple recipients, enter all recipient's email addresses in this field. Separate addressed by spaces, semicolons, or commas (`e.g. "receiver1@example.com, receiver2@example.com"`). Alternatively, you can use a single email address and define a distribution list in your mail software. The distribution list approach is best if the recipients frequently change or there is a large number of them. Multiple recipients are also supported. They can be delimited by space, comma, or semicolon.

Use legacy subject line processing

File Configuration field:

nUseLegacySubjectProcessing

Description:

This checkbox specifies which type of subject line processing will be done. If it is checked, the old-style processing using single character replacement sequences is applied. If it is left unchecked, the far more powerful event property based method is used.

In legacy mode, the following replacement characters are recognized inside the subject line:

`%s` IP address or name (depending on the "resolve hostnames" setting) of the source system that sent the message.

`%f` Numeric facility code of the received message

`%p` Numeric priority code of the received message

`%m` the message itself. Please note: this is the complete message text and can be rather lengthy. As such, it is most probably subject to truncation. If that occurs, all other information after the `%m` replacement character is also truncated. As such, we strongly recommend using the `%m` replacement at the end of the subject line only.

`%%` It represents a single `%` sign. As an example, you may have the subject line set to `Event from %s: "m"` and enabled legacy processing. If a message `This is a test` were received from `172.16.0.1`, the resulting email subject would read: `Event from 172.16.0.1: This is a test`

In non-legacy mode, the Property Replacer can be used. With it, you can include any property from the event message and also modify it. Please visit the Property Replacer documentation for details.

As an example, in non-legacy mode, you can set the subject line to `Mesg: '%msg:1:15%' From: %fromhost%`. If the message `This is a lengthy test message` were received from `172.16.0.1`, the resulting email subject would read: `Mesg: 'This is a lengt' From: 172.16.0.1`. Please note that the message is truncated because you only extracted the first 15 characters from the message text (position 1 to 15).

Subject

File Configuration field:

szSMTPSubject

Description:

Subject line to be used for outgoing emails and it is used for each message sent. It can contain replacement characters or “Event Properties” to customize it with event details. This is especially useful when sending email to cellular phones or pagers, which often display only the subject line and not the actual message body. The subject line – after expansion of the any replacement sequences – can hold a maximum of 255 characters. Characters beyond this will be truncated. Please note that many email systems impose a more strict limit and truncation may occur before the 255-character limit. It is advisable to limit the subject line length to 80 characters or less.

The mail body will also include full event information, including the source system, facility, priority, and actual message text as well as any other information that came with this event. As there is no size limitation for message bodies, the body always contains the full message received (except otherwise configured – see below).

Please note that Insert Menu entry allows you to add replacement characters e.g. `%msg%` - you can send out the actual message of an event in the subject line.

There will be one email for each received message. Email delivery is meant for urgent notifications and actions (e. g. calling pagers and such). It is not meant to provide an email report.

Please note that The message content of the Message field can be configured. Event properties are described in the property replacer section.

Mail Priority

File Configuration field:

nMailPriority

- 0 = low
- 1 = Default
- 2 = High

Description:

Here you can adjust the priority with which the mail will be sent. You can choose between “low”, “normal”, and “high” priority. With this you can give your setup some complexity, being able to send some events as “important” and others with less importance.

Mail Message Format

File Configuration field:

szSMTPBody

Description:

This is the format of the message body. Properties from the event can be included by using the Property Replacer. Please note that the message body is only sent if “Include Message/Event in Email Body” is checked.

Output Encoding

File Configuration field:

nOutputEncoding

Description:

Determines the character encoding mode.

Use XML to Report

File Configuration field:

nUseXMLtoReport

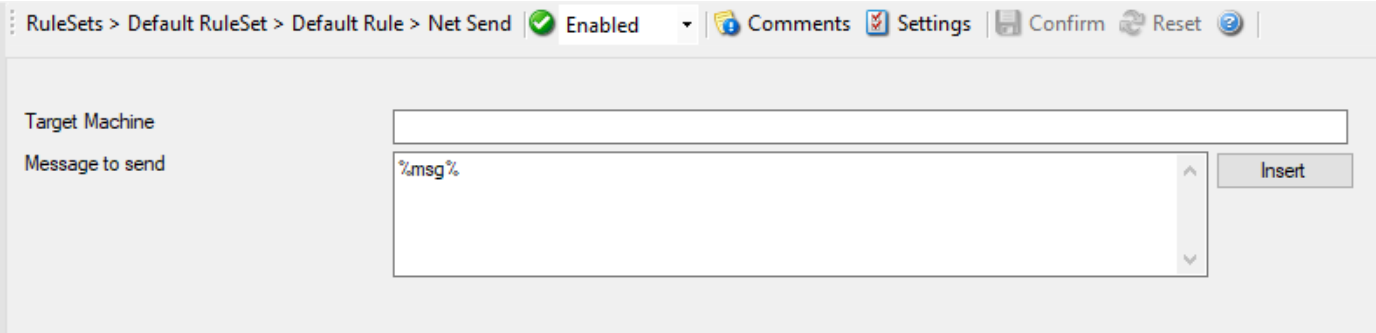
Description:

If checked, the received event will be included in XML format in the mail. If so, the event will include all information, like the original timestamp, the facility, priority etc. XML format is especially useful if the mail is sent to an automated system, which will then parse the message.

If unchecked, just the plain text message will be included in the mail. This format is more readable for a human reader.

Net Send

With the “Net Send” action, short alert messages can be sent via the Windows “net send” facility. These messages are delivered on a best-effort basis. If the recipient can be reached, they will pop up in a message box on the recipient’s machine. If the recipient cannot be reached, they will simply be discarded. No buffering takes place. Consequently, the rule engine does not check if the message can be delivered. It will never flag an action to be in error due to a reported delivery problem with “net send”.



- Action - Net Send*

Target Machine

File Configuration field:

szTarget

Description:

This is the Windows user name of the intended recipient, a NETBIOS machine name, or even an IP address (in the form of `10.1.1.1`). You can either use an IPv4, an IPv6 Address or a Hostname that resolves to an IPv4 or IPv6 Address.

Message to send

File Configuration field:

szMessage

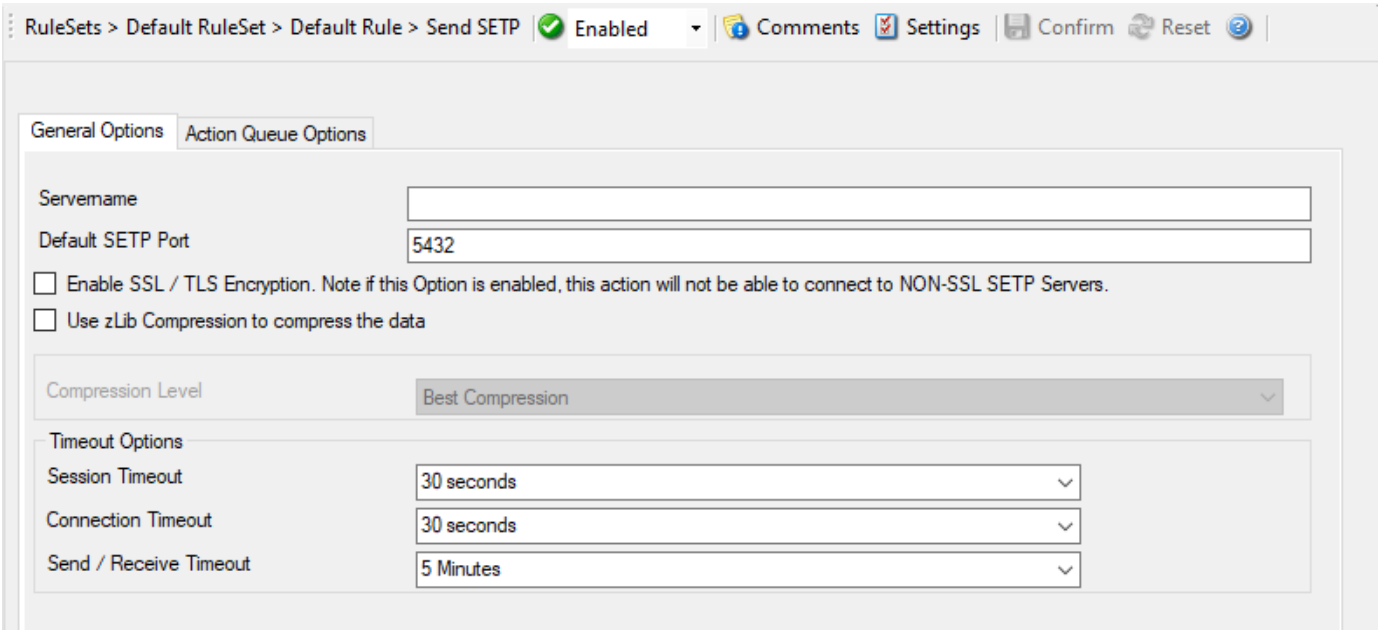
Description:

This is the message that is sent to the intended target.

Please note that the message content of the Message to send field can now be configured. event properties are described in the property replacer section.

Send SETP

With the “Send SETP” action, messages can be sent to a SETP server.



- Action - Send SETP General*

Servername

File Configuration field:

szServer

Description:

The MonitorWare Agent sends setp to the server/listener under this name. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address.

Default SETP Port

File Configuration field:

nMIAPSendPort

Description:

The Send setp sends outgoing requests on this port. The default value is 5432. Set the port to 0 to use the system-supplied default value (which defaults to 5432 if not modified by a system administrator).

Instead of the port number, a service name can be used. If so, that name is looked up via the socket service database functions. The lookup is for protocol TCP.

Please Note: The SETP port configured here must match the port configured at the listener side (i.e. MonitorWare Agent or WinSyslog Enterprise edition). If they do not match, a Send SETP session cannot be initiated. The rule engine will log this to the Windows Event Log.

Enable SSL / TLS Encryption

File Configuration field:

nUseSSL

Description:

If this option is enabled then this action will be able to connect to SSL/TLS setp servers. Please make sure that you want this option to be enabled.

Use zLib Compression to compress the data

File Configuration field:

nZlibComp

Description:

It enables zLib compression support. Note that the SETP receiver must have zLib Compression support and enabled, otherwise it does not work.

Compression Level

File Configuration field:

nCompLevel

- 1 = Best Speed
- 3 = Low Compression
- 6 = Normal Compression
- 9 = Best Compression

Description:

Higher level results in better compression but slower performance.

Session Timeout

File Configuration field:

nTimeOutSession

Description:

The maximum time a session to a SETP server is to be kept open.

Connection Timeout

File Configuration field:

nConnectTimeOut

Description:

Maximum time a connection can take to connect or disconnect.

Send / Receive Timeout

File Configuration field:

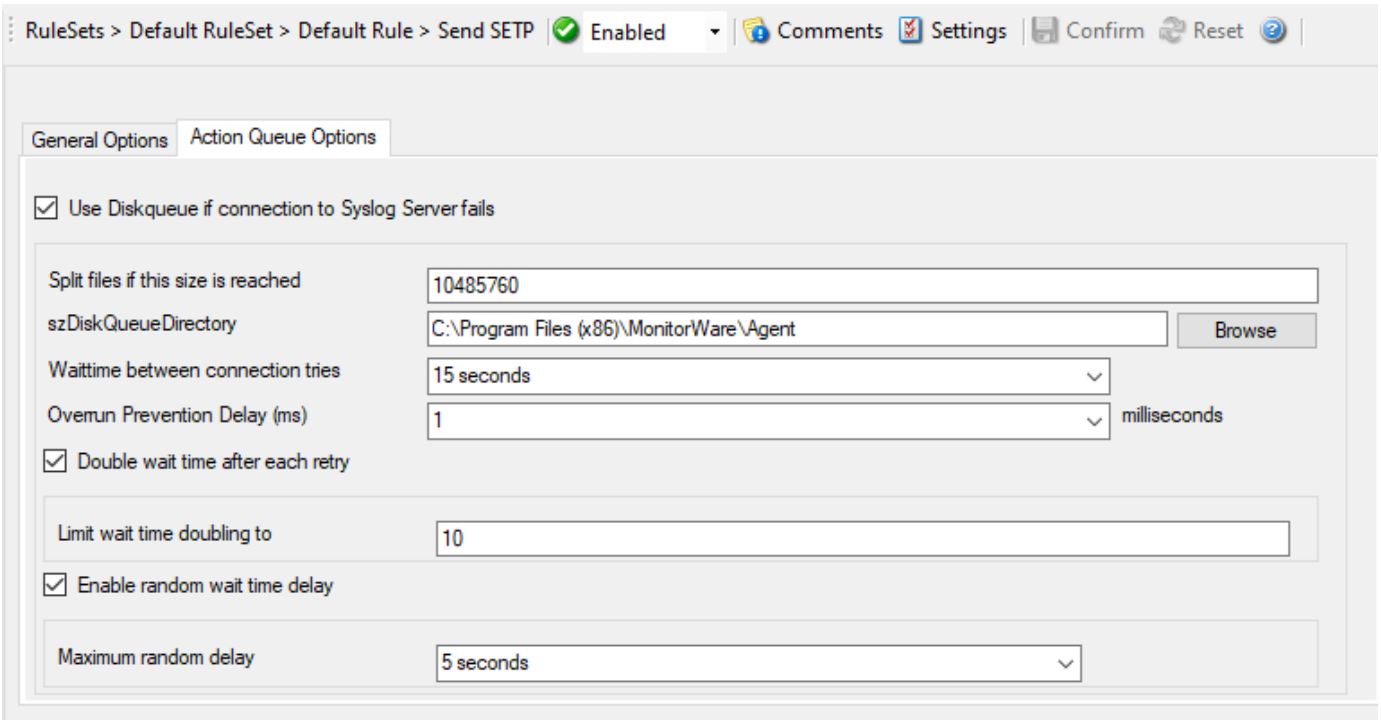
nSendRecvTimeOut

Description:

When sending or receiving data, this timeout applies.

Please note: If this option is enabled, this action is not be able to connect to NON-SSL SETP servers.

Action Queue Options



- Action - Send SETP Action Queue*

Use Diskqueue if connection to Syslog server fails

File Configuration field:

nUseDiscQueue

Description:

Enable diskqueuing syslog messages after unexpected connection loss.

Split files if this size is reached

File Configuration field:

nDiskQueueMaxFileSize

Description:

Files will be split until they reach the configured size in bytes. The maximum support file size is 10485760 bytes.

Diskqueue Directory

File Configuration field:

szDiskQueueDirectory

Description:

The directory where the queue files will be generated in. The queuefiles will be generated with a dynamic UUID bound to the action configuration.

Waittime between connection tries

File Configuration fields:

nDiskCacheWait

Description:

The minimum waittime until the Syslog Action retries to establish a connection to the Syslog server after failure.

Overrun Prevention Delay (ms)

File Configuration field:

nPreventOverrunDelay

Description:

When the Action is processing syslog cache files, an overrun prevention delay can be added to avoid flooding the target Syslog server.

Double wait time after each retry

File Configuration field:

bCacheWaittimeDoubling

Description:

If enabled, the configured waittime is doubled after each try.

Limit wait time doubling to

File Configuration field:

nCacheWaittimeDoublingTimes

Description:

How often the waittime is doubled after a failed connection try.

Enable random wait time delay

File Configuration field:

bCacheRandomDelay

Description:

If enabled, a some random time will be added into the waittime delay. When using many syslog senders, this can avoid that all senders start sending cached syslog data to the Syslog server at the same time.

Maximum random delay

File Configuration field:

nCacheRandomDelayTime

Description:

Maximum random delay time that will be added to the configured waittime if Enable random wait time delay is enabled.

Syslog Forwarding

Protocol Type

There are various ways to transmit syslog messages. In general,they can be sent via UDP, TCP, or RFC 3195 RAW. Typically, syslog messages are received via UDP protocol, which is the default. UDP is understood by almost all servers, but does not guarantee transport. In plain words, this means that syslog messages sent via UDP can get lost if there is a network error, the network is congested or a device (like a router or switch) is out of buffer space. Typically, UDP works quite well. However, it should not be used if the loss of a limited number of messages is not acceptable.

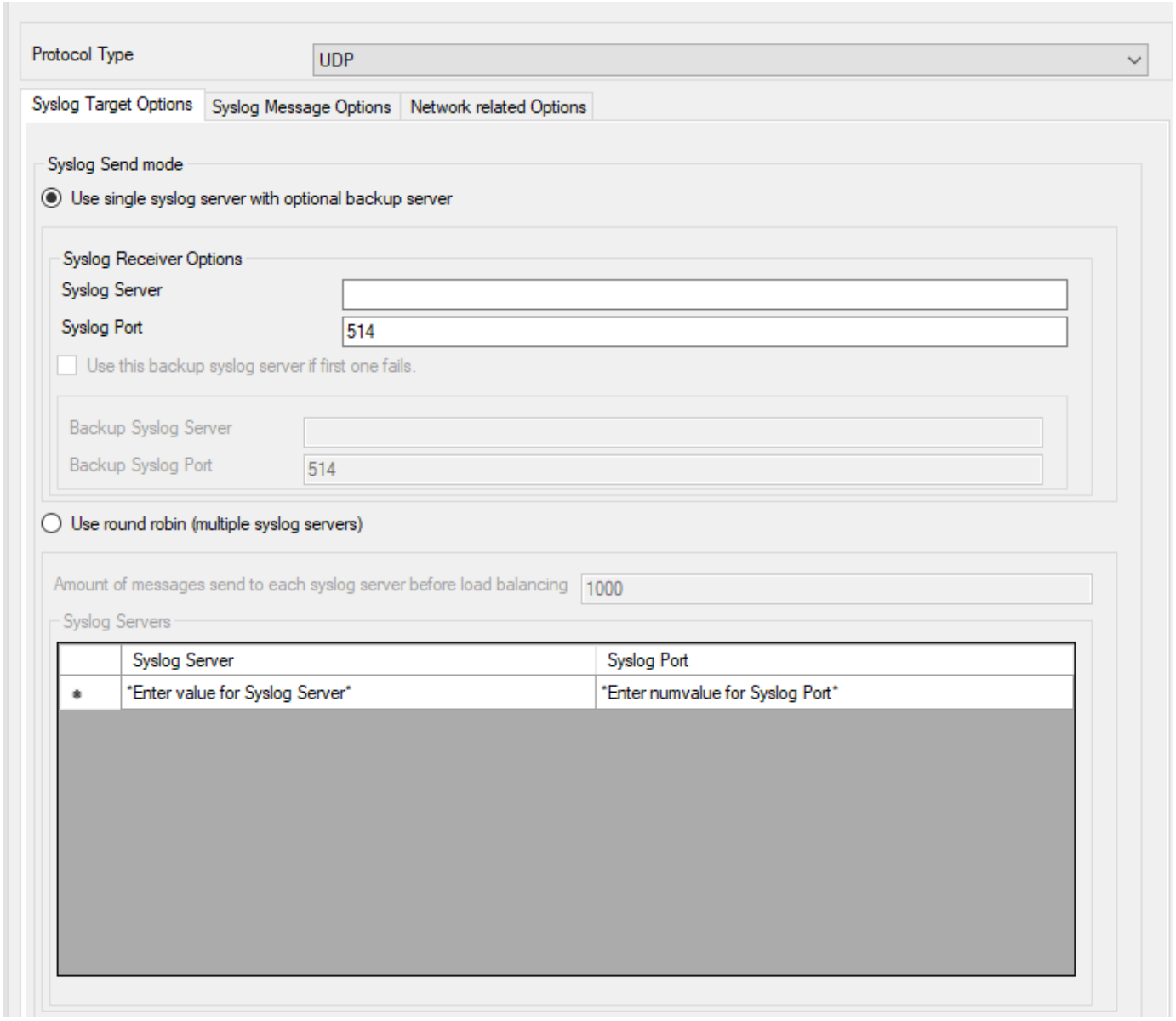
TCP and RFC 3195 based syslog messages offer much greater reliability. RFC 3195 is a special standardized transfer mode. However, it has not receive any importance in practice. Servers are hard to find. As one of the very few, Adiscon products support RFC 3195 also in the server implementations. Due to limited deployment, however, RFC 3195 is very little proven in practice. Thus we advise against using RFC 3195 mode if not strictly necessary (e.g. part of your requirement sheet).

TCP mode comes in three flavors. This stems back to the fact that transmission of syslog messages via plain TCP is not yet officially standardized (and it is doubtful if it ever will be). However, it is the most relevant and most widely implemented reliable transmission mode for syslog. It is a kind of unwritten industry standard. We support three different transmission modes offering the greatest compatibility with all existing implementations. The mode “TCP (one message per connection)” is a compatibility mode for Adiscon servers that are older than roughly June 2006. It may also be required for some other vendors. We recommend not to use this setting, except when needed. “TCP

(persistent connection)” sends multiple messages over a single connection, which is held open for an extended period of time. This mode is compatible with almost all implementations and offers good performance. Some issues may occur if control characters are present in the syslog message, which typically should not happen. The mode “TCP (octet-count based framing)” implements algorithms of an IETF standard RFC 6587. It also uses a persistent connection. This mode is reliable and also deals with embedded control characters very well. This standard is now widely supported by modern syslog receivers and implementations.

As a rule of thumb, we recommend to use “TCP (octet-count based framing)” if you are dealing only with (newer) Adiscon products. Otherwise, “TCP (persistent connection)” is probably the best choice. If you select one of these options, you can also select a timeout. The connection is torn down if that timeout expires without a message being sent. We recommend to use the default of 30 minutes, which should be more than efficient. If an installation only occasionally sends messages, it could be useful to use a lower timeout value. This will free up connection slots on the server machine.

Syslog Target Options



- Action - Forward Syslog Target Options*

Syslog Send mode

File Configuration field:

nSendMode

Description

The Sendmode has been added since 2018 into all products supporting the forward syslog action. There are two options available.

Use single Syslog server with optional backup server This is the classic syslog send mode which uses a primary Syslog server and a secondary backup Syslog server if configured.

Use round robin (multiple syslog servers) This new method allows you to configure multiple targets that will be used one by one after a configured amount of messages has been sent to each target.

Syslog server (Syslog Send mode)

File Configuration field:

szSyslogSendServer

Description:

This is the name or IP address of the system to which Syslog messages should be sent to. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address.

Syslog Port (Syslog Send mode)

File Configuration field:

nSyslogSendPort

Description:

The remote port on the Syslog server to report to. If in doubt, please leave it at the default value of 514, which is typically the Syslog port. Different values are only required for special setups, for example in security sensitive areas. Set the port to 0 to use the system-supplied default value (which defaults to 514 on almost all systems).

Instead of the port number, a service name can be used. If so, that name is looked up via the socket service database functions.

Use this backup Syslog server if first one fails

File Configuration field:

nEnableBackupServer

Description:

The backup server is automatically used if the connection to the primary server fails. The primary server is automatically retried when the next Syslog session is opened. This option is only available when using TCP syslog.

Use round robin (multiple Syslog server)

Amount of messages send to each Syslog server before load balancing

File Configuration field:

nRoundRobinMsgCount

Description:

When using round robin mode, this is the amount of messages to be sent to each configured Syslog server.

Syslog server (Round robin mode)

File Configuration field:

szSyslogServer_[n]

Description:

This is the name or IP address of the system to which Syslog messages should be sent to. You can either use an IPv4, an IPv6 Address, or a Hostname that resolves to an IPv4 or IPv6 Address.

Syslog Port (Round robin mode)

File Configuration field:

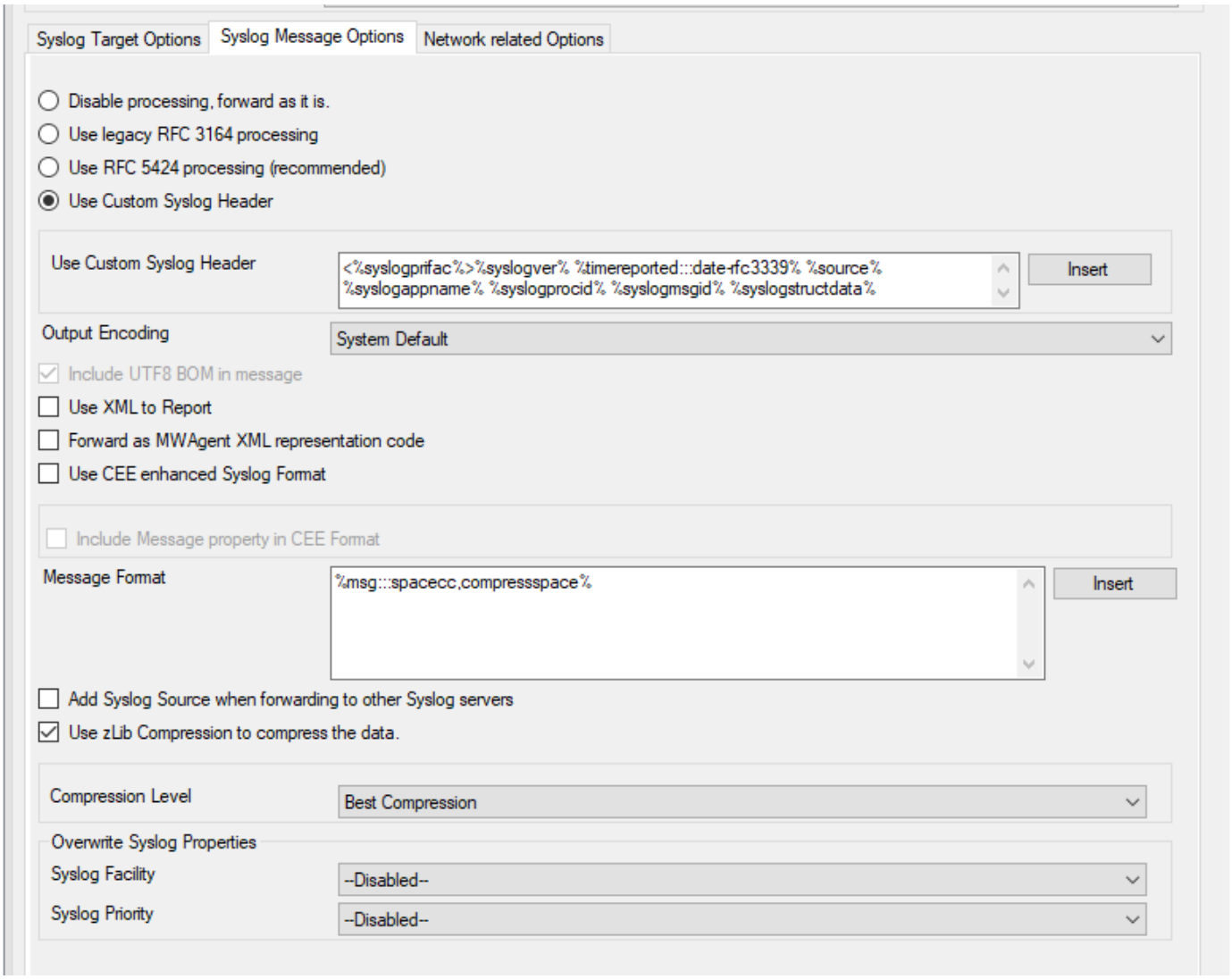
nSyslogPort_[n]

Description:

The remote port on the Syslog server to report to. If in doubt, please leave it at the default value of 514, which is typically the Syslog port. Different values are only required for special setups, for example in security sensitive areas. Set the port to 0 to use the system-supplied default value (which defaults to 514 on almost all systems).

Instead of the port number, a service name can be used. If so, that name is looked up via the socket service database functions.

Syslog Message Options



- Action - Forward Syslog - Message Options*

Syslog processing

File Configuration field:

bProcessDuringRelay

- 0 = Disable processing, forward as it is
- 1 = RFC3164 Header - Use legacy RFC 3164 processing
- 2 = RFC5424 Header - Use RFC 5424 processing (recommended)
- 3 = Custom Syslog Header

Description:

With this settings you can assign how your syslog messages will be processed.

For processing syslog you can choose out of four different options. You can use rfc3164 or RFC5424 (recommended) which is the current syslog standard, you are able to customize the syslog header or you do not process your syslog and forwards it as it is.

Use Custom Syslog Header

File Configuration field:

szCustomSyslogHeader

Description:

In this field you can specify the contents of your syslog header. This option is only available when you choose “Use Custom Syslog Header” in the Syslog Processing menu. The contents can be either a fixed message part which you can write into the field yourself or you use properties as dynamic content. By default the Header field is filled with the content of the RFC 5424 header.

Please note that the header content of the Header field can be configured. event properties are described in the property replacer section.

Output Encoding

File Configuration field:

nOutputEncoding

Description:

This setting is most important for Asian languages. A good rule is to leave it at “System Default” unless you definitely know you need a separate encoding. “System Default” works perfect in the far majority of cases, even on Asian (e.g. Japanese) Windows versions.

Include UTF8 BOM in message

File Configuration field:

nProtocolType

Description:

If enabled (default), the UTF8 BOM code will be prepended to the output message if you are using UTF8 Output encoding. If the syslog receiver cannot handle and remove the UTF8 BOM you can disabled this option.

Use XML to Report

File Configuration field:

bReportInXML

Description:

If this option is checked, the forwarded Syslog message is a complete XML-formatted information record. It includes additional information like timestamps or originating system in an easy to parse format.

The XML formatted message is especially useful if the receiving system is capable of parsing XML data. However, it might also be useful to a human reader as it includes additional information that cannot be transferred otherwise.

Forward as MonitorWare Agent XML Representation Code

File Configuration field:

nForwardIUT

Description:

MonitorWare supports a specific XML-Representation of the event. If it is checked, that XML representation is used. It provides additional information (like informationunit type, original source system, reception time & many more) but is harder to read by a human. At the same time, it is obviously easier to parse.

Use CEE enhanced Syslog Format

File Configuration field:

nReportInJSON

Description:

If enabled, the CEE enhanced Syslog format will be used. All useful properties will be included in a JSON Stream. The message itself can be included as well, see the “Include message property in CEE Format” option. Here is a sample how the format looks like for a security Eventlog message:

```
@cee: {"source": "machine.local", "nteventlogtype": "Security", "sourceproc": "Mic
rosoft-Windows-Security-Auditing", "id": "4648", "categoryid": "12544", "category"
: "12544", "keywordid": "0x8020000000000000", "user": "N\A", "SubjectUserSid": "S-
1-5-11-222222222-333333333-4444444444-5555", "SubjectUserName": "User", "SubjectDo
mainName": "DOMAIN", "SubjectLogonId": "0x5efdd", "LogonGuid": "{00000000-0000-000
0-0000-000000000000}", "TargetUserName": "Administrator", "TargetDomainName": " DO
MAIN ", "TargetLogonGuid": "{00000000-0000-0000-0000-000000000000}", "TargetServer
Name": "servername", "TargetInfo": " servername ", "ProcessId": "0x76c", "ProcessN
ame": "C:\Windows\System32\spoolsv.exe", "IpAddress": "-", "IpPort": "-", "catname
": "Logon", "keyword": "Audit Success", "level": "Information", }
```

Additionally to this format you can set: Include message property in CEE Format.

If enabled, the message itself will be included in the JSON Stream as property. Disable this option if you do not want the message itself in the CEE Format.

Please note you can also make Event ID part of the actual Syslog message while forwarding to a Syslog server then you have to make some changes in the Forward Syslog Action. Click here to know the settings.

Include message property in CEE Format

Description

If enabled, the message itself will be included in the JSON Stream as property. Disable this option if you do not want the message itself in the CEE Format.

Please note you can also make Event ID part of the actual Syslog message while forwarding to a Syslog server then you have to make some changes in the Forward Syslog Action. click here to know the settings.

Message Format

File Configuration field:

szMessageFormat

Description:

The custom format lets you decide how the content of a syslog message looks like. You can use properties to insert content dynamically or have fixed messages that appear in every message. Event properties are described in the property replacer section.

Add Syslog Source when forwarding to other Syslog servers

File Configuration field:

nSyslogInsertSource

Description:

If this box is checked, information on the original originating system is prepended to the actual message text. This allows the recipient to track where the message originally came from.

Please note: This option is not compatible with RFC 3164. We recommend selecting it primarily when message forwarding to a WinSyslog Interactive Server is intended.

Use zLib Compression to compress the data

File Configuration field:

nUseCompression

Description:

With this option you can set the grade of compression for your syslog messages. For more information please read the note at the bottom of this page.

Compression Level

File Configuration field:

nCompressionLevel

- 1 = Best Speed
- 3 = Low Compression
- 6 = Normal Compression
- 9 = Best Compression (default)

Description:

With this option you can set the grade of compression for your syslog messages. For more information please read the note at the bottom of this page.

Note on Using Syslog Compression

Compressing syslog messages is a stable but rarely used feature. There is only a very limited set of receivers who are able to understand that format. Turning on compression can save valuable bandwidth in low-bandwidth environments. Depending on the message, the saving can be anything from no saving at all to about a reduction in half. The best savings ratios have been seen with Windows Event Log records in XML format. In this case, 50% or even a bit more can be saved. Very small messages do not compress at all. Typical syslog traffic in non-xml format is expected to compress around 10 to 25%.

Please note that compression over TCP connections requires a special transfer mode. This mode uses OpenSSL TLS Implementation 3.x for secure transmission. TLS compression is not implemented; instead, the system uses standard OpenSSL compression mechanisms.

Besides the fact that the mechanisms behind compression are experimental, the feature itself is solid.

Overwrite Syslog Properties

Syslog Facility

File Configuration field:

nSyslogFacility

Description:

When configured, will overwrite the Syslog Facility with the configured value.

Syslog Priority

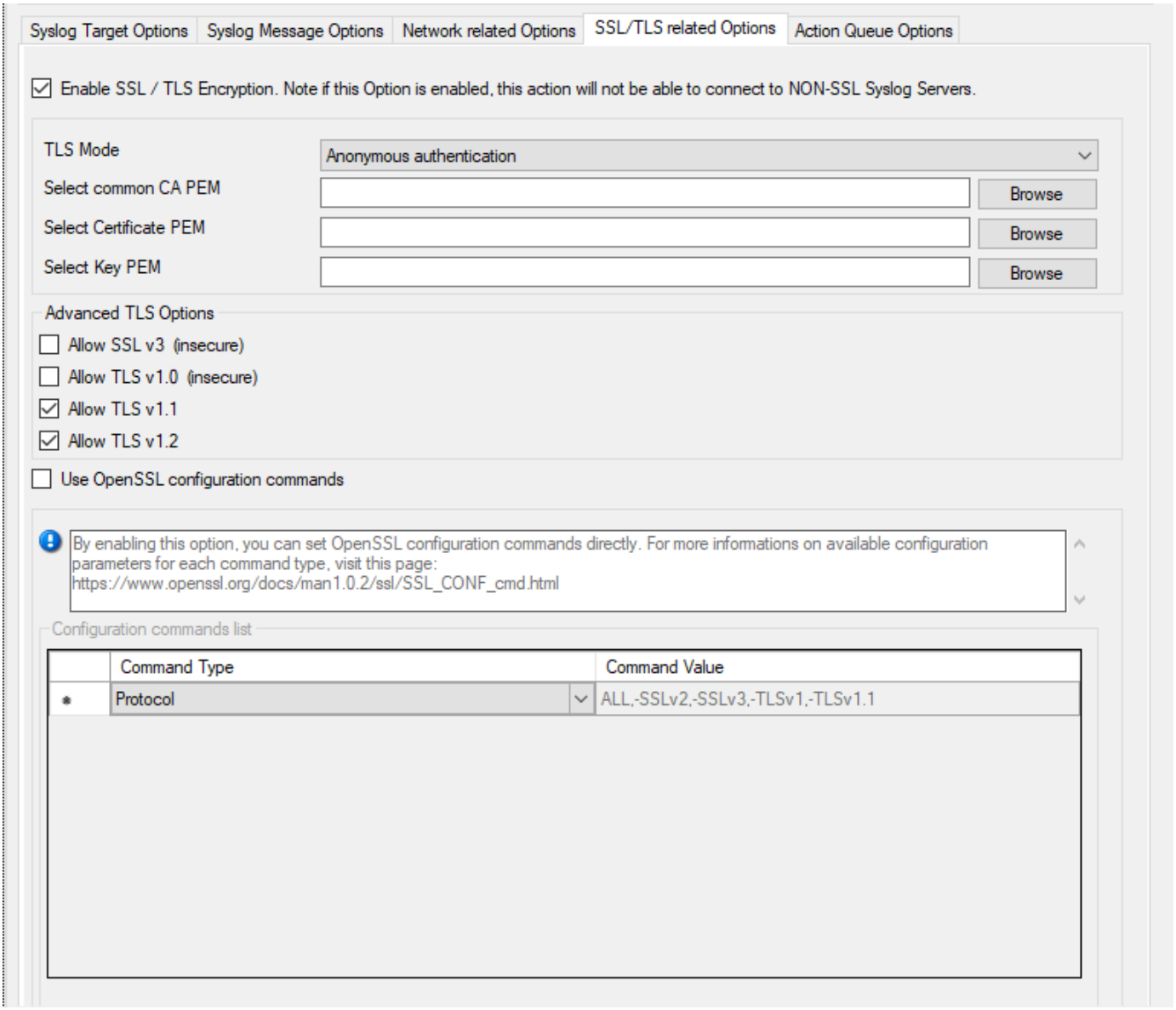
File Configuration field:

nSyslogPriority

Description:

When configured, will overwrite the Syslog Priority with the configured value.

SSL/TLS related Options



- Action - Forward Syslog SSL/TLS related Options*

Enable SSL / TLS Encryption

File Configuration field:

nUseSSL

Description:

If this option is enabled, the action will not be able to talk to a NON-SSL secured server. The method used for encryption is compatible to RFC5425 (Transport Layer Security (TLS) Transport Mapping for Syslog).

TLS Mode

File Configuration field:

nTLSMode

Description:

Anonymous Authentication

Default option. This means that a default certificate will be used.

Use Certificate

If this option is enable, you can specify your own certificate. For further authentication solutions, you will need to create your own certificates using OpenSSL Tools for example.

Select common CA PEM

File Configuration field:

szTLSCAFile

Description:

Select the certificate from the common Certificate Authority (CA). The syslog receiver should use the same CA.

Select Certificate PEM

File Configuration field:

szTLSCertFile

Description:

Select the client certificate (PEM Format).

Select Key PEM

File Configuration field:

szTLSKeyFile

Description:

Select the keyfile for the client certificate (PEM Format).

Allow SSL v3

File Configuration field:

nTLSAllowSSLv3

Description:

This option enables insecure protocol method SSLv3. We recommend NOT enabling this option as SSLv3 is considered broken.

Allow SSL v1.0

File Configuration field:

nTLSAllowTLS10

Description:

This option enables insecure protocol method TLSv1. We recommend NOT enabling this option as TLSv1 is considered broken.

Allow SSL v1.1

File Configuration field:

nTLSAllowTLS11

Description:

This option enables protocol method TLS1.1 which is enabled by default.

Allow SSL v1.2

File Configuration field:

nTLSAllowTLS12

Description:

This option enables protocol method TLS1.2 which is enabled by default.

Allow TLS v1.3

File Configuration field:

nTLSAllowTLS13

Description:

This option enables protocol method TLS1.3 which provides enhanced security and performance.

Use OpenSSL configuration commands

File Configuration field:

nTLSUseConfigurationCommands

Description:

By enabling this option, you can set OpenSSL configuration commands directly. For more information's on available configuration parameters for each command type, visit this page:

https://www.openssl.org/docs/man1.0.2/ssl/SSL_CONF_cmd.html

We allow to the set the following OpenSSL configuration commands in the configuration commands list:

- CipherString: Sets the allowed/disallowed used Ciphers. Setting this value will OVERWRITE the internal default ciphers.
- SignatureAlgorithms: This sets the supported signature algorithms for TLS v1.2.
- Curves: This sets the supported elliptic curves.
- Protocol: Sets the supported versions of the SSL or TLS protocol. This will OVERWRITE the Allow SSL options from above!
- Options: The value argument is a comma separated list of various flags to set.

When setting advanced configuration commands, we highly recommend to enable debug logging and review it after changes have been made. An error will be logged in the debug logfile if a configuration command cannot be processed successfully.

TCP related Options

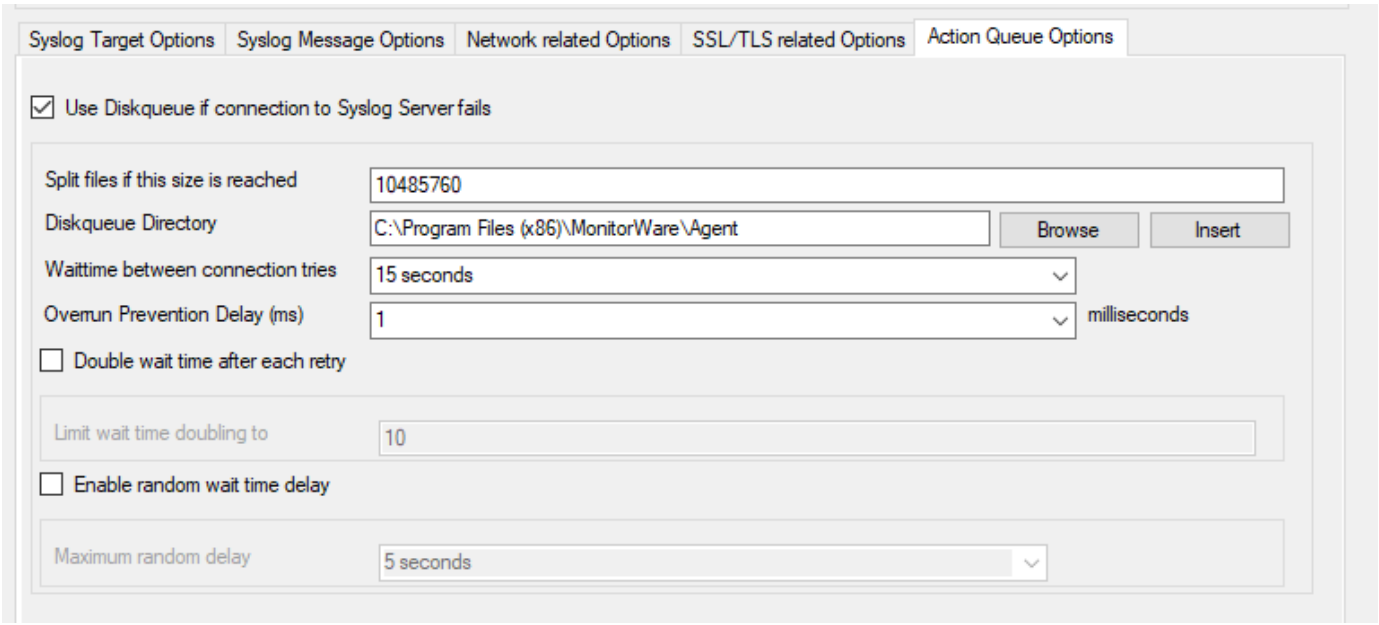
When using TCP-based syslog forwarding, you have the additional option to use the diskqueue. Whenever a connection to a remote Syslog server fails, the action starts caching the syslog messages into temporary files. The folder for these files can be configured. The filenames are generated using a unique GUID which is automatically generated for each Action, thus enabling you to use this feature in multiple Actions. Once the Syslog server becomes available again, the cached messages are being sent automatically. If you restart the Service while the Syslog Cache was active, it cannot be checked during service startup if the Syslog server is available now. Once the action is called again, the check is done and if the Syslog server is available, the messages are being sent. The size of this cache is only limited by the disk size. Files are split by 10MB by default, but this can also be configured. The maximum supported file size is 2GB.

Please Note: This option is not available for UDP or RFC 3195.

Session Timeout

File Configuration field: nTimeoutValue

Action Queue Options



- Action - Forward Syslog Action Queue*

Use Diskqueue if connection to Syslog server fails

File Configuration field:

nUseDiscQueue

Description:

Enable diskqueuing syslog messages after unexpected connection loss.

Split files if this size is reached

File Configuration field:

nDiskQueueMaxFileSize

Description:

Files will be split until they reach the configured size in bytes. The maximum support file size is 10485760 bytes.

Diskqueue Directory

File Configuration field:

szDiskQueueDirectory

Description:

The directory where the queue files will be generated in. The queuefiles will be generated with a dynamic UUID bound to the action configuration.

Waittime between connection tries

File Configuration fields:

nDiskCacheWait

Description:

The minimum waittime until the Syslog Action retries to establish a connection to the Syslog server after failure.

Overrun Prevention Delay (ms)

File Configuration field:

nPreventOverrunDelay

Description:

When the Action is processing syslog cache files, an overrun prevention delay can be added to avoid flooding the target Syslog server.

Double wait time after each retry

File Configuration field:

bCacheWaittimeDoubling

Description:

If enabled, the configured waittime is doubled after each try.

Limit wait time doubling to

File Configuration field:

nCacheWaittimeDoublingTimes

Description:

How often the waittime is doubled after a failed connection try.

Enable random wait time delay

File Configuration field:

bCacheRandomDelay

Description:

If enabled, a some random time will be added into the waittime delay. When using many syslog senders, this can avoid that all senders start sending cached syslog data to the Syslog server at the same time.

Maximum random delay

File Configuration field:

nCacheRandomDelayTime

Description:

Maximum random delay time that will be added to the configured waittime if Enable random wait time delay is enabled.

UDP related Options

Enable IP Spoofing for the UDP Protocol

File Configuration field:

nSpoofIPAddress

Description:

This option enables you to spoof the IP Address when sending Syslog messages over UDP. Some notes regarding the support of IP Spoofing. It is only supported the UDP Protocol and IPv4. IPv6 is not possible yet. Due system limitations introduced by Microsoft, IP Spoofing is only possible on Windows Server 2003, 2008, or higher. It is NOT possible in Windows XP, VISTA, 7, or higher. For more information see the Microsoft explanation. Also please note that most routers and gateways may drop network packages with spoofed IP Addresses, so it may only work in local networks.

Fixed IP or single property

File Configuration field:

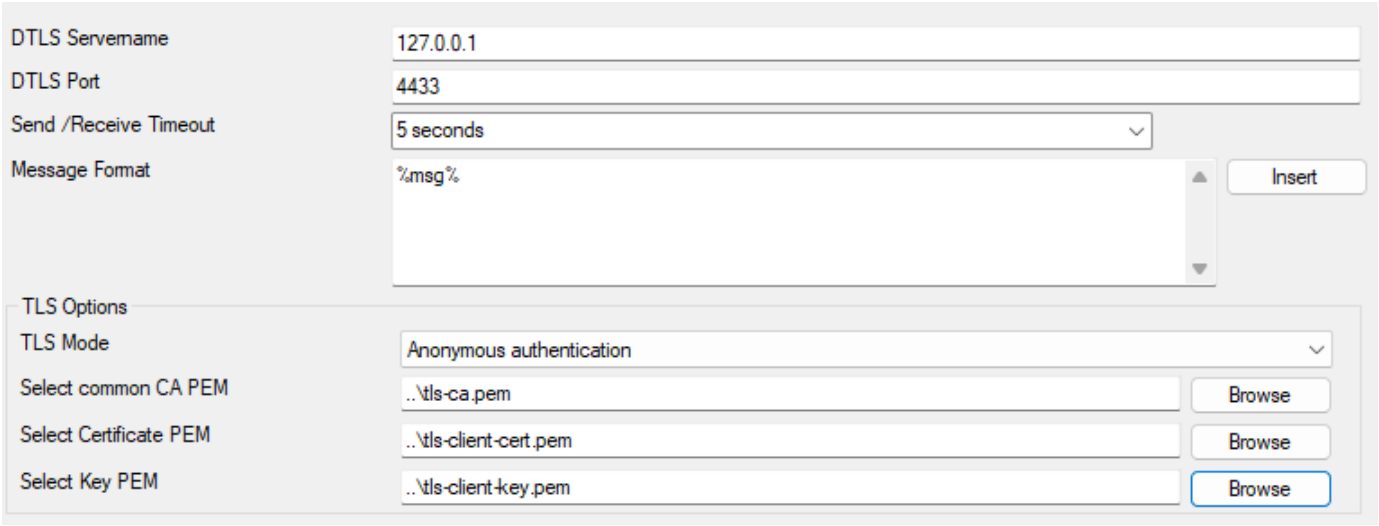
szSpoofedIPAddress

Description:

You can either use a static IP Address or a property. When using a property, the IP Address is tried to be resolved from the content of the property. For example by default the `%source%` property is used. If the name in this property cannot be resolved to an IP Address, the default local IP Address will be used.

Send DTLS

This action sends messages securely using the Datagram Transport Layer Security (DTLS) protocol. It ensures message confidentiality and integrity over an encrypted channel. The implementation uses OpenSSL to handle encryption and decryption, ensuring robust security and compatibility with industry standards. DTLS is suitable for applications requiring low latency and secure communication.



- Action - Send DTLS Configuration*

DTLS Servername

File Configuration field:

szDTLSServer

Description:

The hostname or IP address of the DTLS server to which messages should be sent. You can use an IPv4, IPv6 address, or a hostname resolving to one of these.

DTLS Port

File Configuration field:

nDTLSSendPort

Description:

The port number on the DTLS server where messages are sent. Typically, this port is 4433.

Send/Receive Timeout

File Configuration field:

nSendTimeOut

Description:

Specifies the time in seconds to wait for a response from the DTLS server before timing out. For instance, set this value to “5 seconds” for a 5-second timeout.

Message Format

File Configuration field:

szMessage

Description:

Defines the format of the message being sent. Use placeholders like “%msg%” to define the dynamic content of the message. Multi-line messages are supported.

TLS Options

TLS Mode

File Configuration field:

nTLSMode

Description:

Specifies the authentication method used. Options include “Anonymous authentication” or other modes requiring certificates.

Select common CA PEM

File Configuration field:

szTLSCAFile

Description:

Path to the CA certificate file (e.g., *tls-ca.pem*).

Select Certificate PEM

File Configuration field:

szTLSCertFile

Description:

Path to the client certificate file (e.g., *tls-client-cert.pem*).

Select Key PEM

File Configuration field:

szTLSKeyFile

Description:

Path to the private key file for the client (e.g., *tls-client-key.pem*).

internal actions

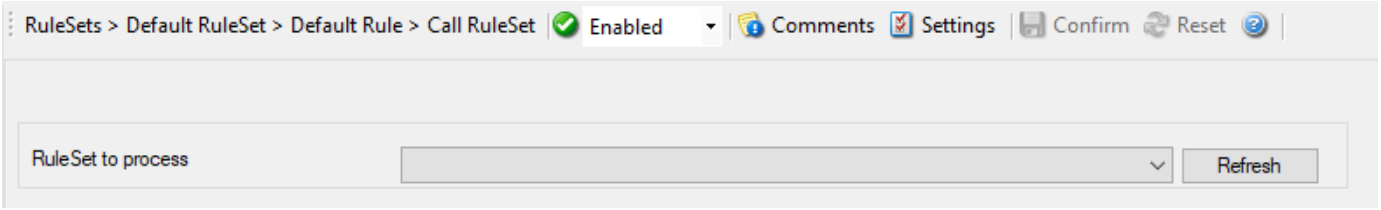
Call RuleSet

A Call RuleSet action simply calls another ruleset in some existing ruleset. When this action is encountered, the rule engine leaves the normal flow and goes to the called ruleset (which may contain many rules as well). It executes all the rules that have been defined in the called RuleSet. After the execution of all of them, it will return to its point from where it left the original flow. Let’s take an example to clarify it a little further.

Let’s say that Rule 1 has two actions - Action 1 and Action 2. The Action 1 of Rule 1 is an include (Call Ruleset) action. If the filter condition result of Rule 1 evaluates to true, it will execute the Action 1. Since Action 1 is the include action in this example, it will go to the included ruleset and will execute its filter condition. If that filter condition evaluates to true, it will execute all of its actions and will return to Action 2 of Rule 1 (of normal flow). If on the other hand, the filter condition of the included rule set evaluates to false, it will skip all of its actions and will come back to the Action 2 of Rule 1 (of normal flow).

Note: there is no limit on including the rules which means that a rule

that has been included in another rule may contain another rule in it which might contain another rule in it and so on.



- Action - Call RuleSet*

Ruleset to Call

File Configuration field:

szRuleSet

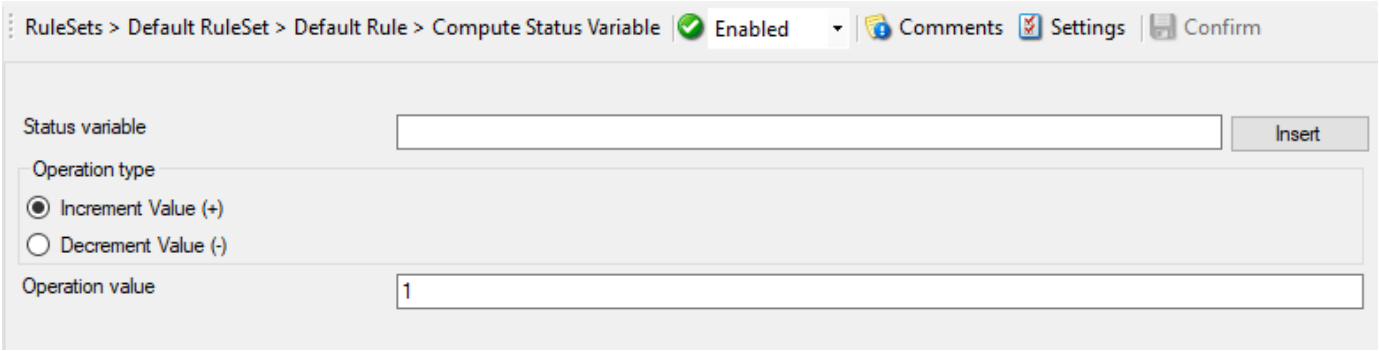
Description:

Select the Ruleset to be called.

Note: Call RuleSet stays disabled until you have more then “One” RuleSet!

Compute Status Variable

An internal action used to compute a status variable. This is needed for RuleSets which operate on a counter basis. This dialog controls the compute status variable options.



- Action - Compute Status Variable*

Status variable

File Configuration field:

szStatusVar

Description:

Name of the unique status variable.

Operation Type

Increment Value

File Configuration field:

nCalcType = 1

Description:

It increments the value by the operation value.

Decrement Value

File Configuration field:

nCalcType = 2

Description:

It decrements the value by the operation value.

Operation value

File Configuration field:

nChangeVal

Description:

The operation value that is to be used.

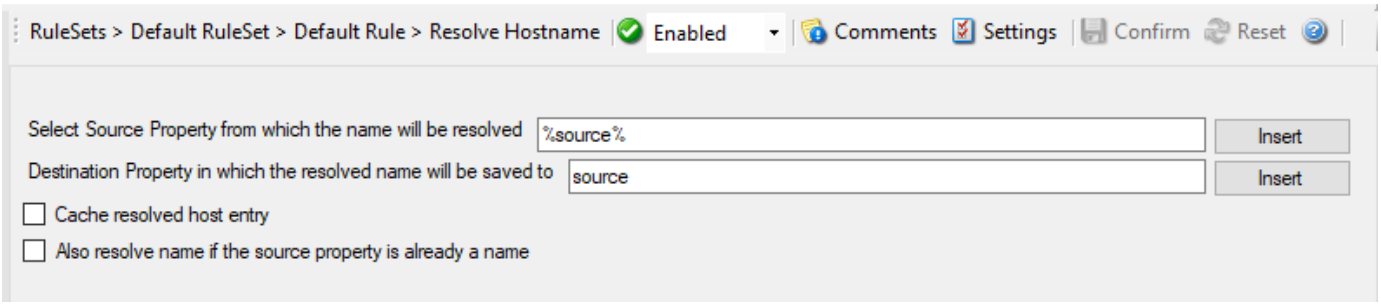
Discard

A Discard Action immediately destroys the current Information Unit and any action of any rule that has been defined after the Discard action execution. When this action has been selected then no dialog appears as nothing needs to be configured for this.

A sample how to ignore Events with the Discard Action can be found here: ignoring events.

Resolve Hostname Action

Many Customers asked for resolve hostname options in different services. This feature has now been implemented as an action. An action can be used with every service, and it does not delay the work of a service.



- Action - Resolve Hostname*

Select Source Property from which the name will be resolved

File Configuration field:

szSourcePropertyName

Description:

Click on the Insert menu link on the right side of the textfield to customize the source property from which the name will be resolved.

Destination Property in which the resolved name will be saved to

File Configuration field:

szDestinationPropertyName

Description:

Same as above, please click on the Insert menu link on the right side of the textfield to customize the destination property in which the resolved name will be saved to.

Also resolve name if the source property is already a name

File Configuration field:

nResolveIfName

Description:

Activates the feature that the name will also be resolved if there is already a source property with that name.

Cache resolved host entry

File Configuration field:

nCacheNameEntry

Description:

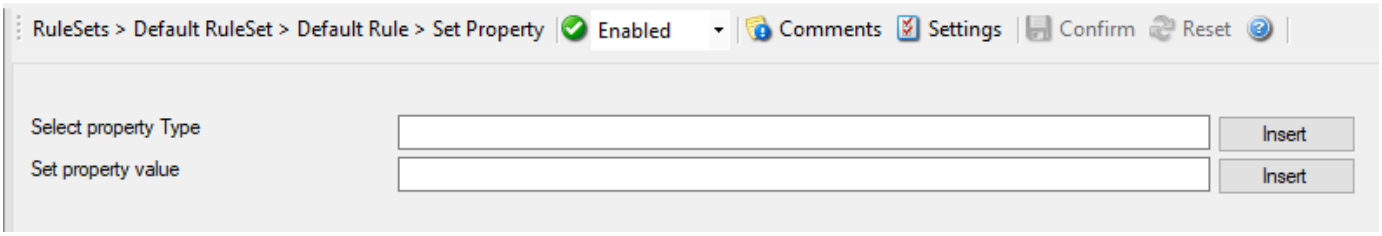
If activated this will, as it says, cache the resolved host entry.

Set Property

You can set every property and custom properties using this action.

This dialog controls the set property options. With the “Set Property” action, some properties of the incoming message can be modified. This is especially useful if an administrator would like to e.g. rename two equally named devices.

Please note: when you change or create a property, the value will be changed as soon as the set property action is carried out. It will not change before that happens and the old value is no longer available thereafter. That means all actions and filter conditions will use the new value after it is set. So, if you would like e.g. rename a system, make sure the set property actions are at the top of the rule base!



- Action - Set Property*

Select Property Type

File Configuration field:

szPropertyType

Description:

Select the property type to be changed. The list box contains all properties that can be changed. By default it is set to nothing.

Please note that the field content can be configured with event properties are described in the property replacer section.

Set Property Value

File Configuration field:

szPropertyValue

Description:

The value to be assigned to the property. Any valid property type value can be entered.

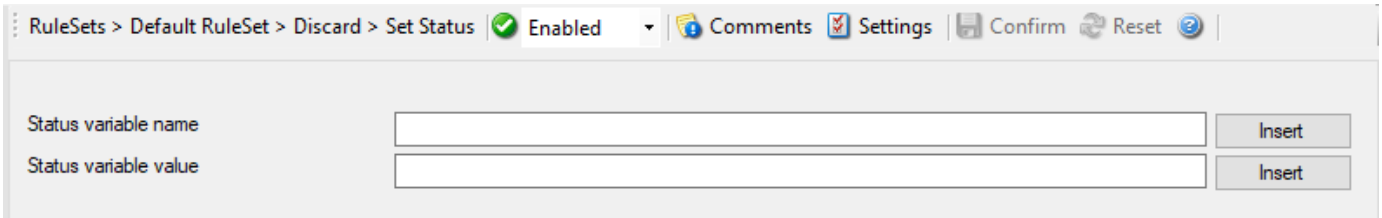
Please note that the field content can be configured with event properties are described in the property replacer section.

Set Status

Each information unit have specific properties e.g. EventID, Priority, Facility etc. These properties have some values. Lets suppose that EventID has property value 01. Now, If you want to add “a new property of your own choice” in the existing set of properties then Set Status action allows you to accomplish this!

You can create a new property and assign any valid desired value to it e.g. we create a new property as CustomerID and set its value to 01. After you have created the property through this action, then you can define filters for them. There is an internal status list within the product which you can use for more complex filtering.

Please note: when you change a property, the value will be changed as soon as the set status action is carried out. It will not change before that happens and the old value is no longer available thereafter. That means all actions and filter conditions will use the new value after it is set. So if you would like e.g. rename a system, make sure the set status actions are at the top of the rule base!



- Action - Set Status*

Status Variable Name

File Configuration field:

szPropertyName

Description:

Enter the Property name. That name will from now on be used inside the rule base. More precisely, it will be used in the filter conditions and actions.

Please note that the field content can be configured with event properties are described in the property replacer section.

Status Variable Value

File Configuration field:

szPropertyValue

Description:

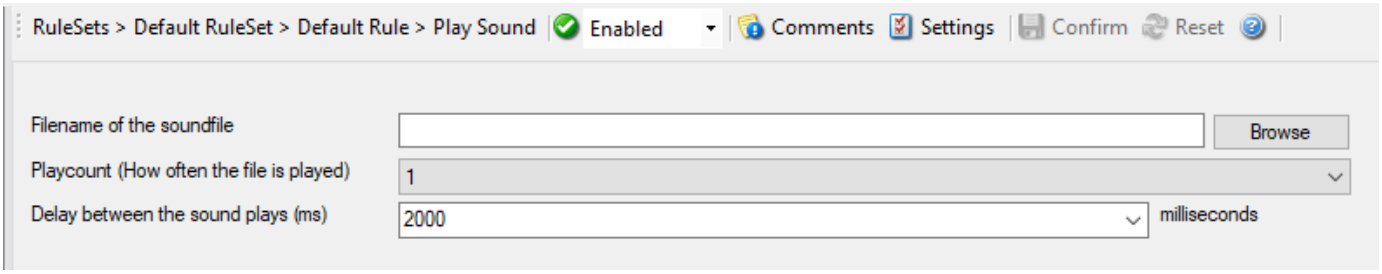
The value to be assigned to the property. Any valid property type value can be entered.

Please note that the field content can be configured with event properties are described in the property replacer section.

other actions

Play Sound

This action allows you to play a sound file. Since Windows VISTA/2008/7, Microsoft has disabled any interaction between a system service and the user desktop. This includes playing sounds as well. So if you want to use the Play Sound Action on any of this Windows Version, you will need to run the service in console mode (From command prompt with the -r option).



- Action - Play Sound*

Please note: if your machine has multiple sound cards installed, the “Play Sound” action will always use the card, that was installed first into the system.

There is a work around if you want to use play sound action for a second sound card!

Filename of the Soundfile

File Configuration field:

szFilename

Description:

Please enter the name of the sound file to play. **This must be a .WAV** file, other formats (like MP3) are not supported. While in theory it is possible that the sound file resides on a different machine, we highly recommend using files on the local machine only. Using remote files is officially not supported (but currently doable if you are prepared for some extra effort in getting this going).

If the file can either not be found or is not in a valid format, a system beep is emitted instead (this should - by API definition - be possible on any system).

Playcount

File Configuration field:

nCount

Description:

This specifies how many times the file is played. It can be re-played up to a hundred times.

Delay between Plays

File Configuration field:

nDelay

Description:

If multiple repeats are specified, this is the amount of time that is to be waited for between each individual play.

Start Program

With the “Start Program” action, an external program can be run. Any valid Windows executable can be run. This includes actual programs (EXE files), as well as scripts like batch files (.BAT), or VB scripts (.vbs).

Start process can, for example, be combined with the service monitor to restart failed services. Another example application is a script that deletes temporary files if the disk space monitor detects a low space condition.

- Action - Start Program*

Command to execute

File Configuration field:

szCommand

Description:

This is the path of actual program file to be executed. This can be the path of any valid executable file. A relative file name can be specified if it can be found via the operating system default search path.

Use legacy parameter processing

File Configuration field:

nUseLegacyProcessing

Description:

When enabled, old style parameter processing is used. Otherwise all properties can be used.

Parameters

File Configuration field:

szParameters

Description:

These parameters are passed to the program executed. They are passed as command line parameters. There is no specific format – it is up to the script to interpret them.

Parameters can contain replacement characters to customize it with event details. This allows passing event data to the script. The following replacement characters can be used:

- %d Date and time in local time
- %s IP address or name (depending on the “resolve hostnames” setting) of the source system that sent the message.
- %f Numeric facility code of the received message
- %p Numeric priority code of the received message
- %m The message itself
- %% Represents a single % sign.

In the example above, replacement characters are being used. If a message “This is a test” were received from “172.16.0.1”, the script would be started with 3 parameters:

Parameter 1 would be the string “e1” – it is assumed that this has some meaning to the script. Parameter 2 would be the IP address, 172.16.0.1. Parameter 3 would be “This is a test”. Please note that due to the two quotes (“), the message is interpreted as a single parameters. If they were missing, it would typically be split into several ones, with parameter 3 being “This”, 4 being “is”, and so on. So these quotes are very important!

Sync Timeout

File Configuration field:

nSyncTimeOut

Description:

Time Out option is under Sync. Processing. When a program is executed, the service waits for it to finish before it carries on further actions. This is needed in order to ensure that all actions are carried out in the correct sequence.

The external program should only run for a limited amount of time. If it would block for some reason, the agent would be prevented from carrying out any further processing. As such, a timeout value must be specified. If the program still runs after the configured timeout, the rule engine cancels it, flags the action as unsuccessful, and then carries on with processing.

Important: Even though the timeout value can be as high as 30 seconds, we strongly recommend limiting the run time of external program to below 5 seconds. Otherwise, they could affect the overall performance too much. If the average run time is 5 seconds, the default timeout of 10 seconds ensures that the program can finish even when there is high system activity.

For performance reasons, we also strongly recommend to use the “Start Program” action only for rules that apply relatively seldom.

Getting Help

EventReporter is very reliable. Here’s how to get help if you encounter problems.

Please note that all options (except priority support) are also open to evaluating customers. So do not hesitate to try them. Help is available in English and German language. Our local resellers may provide support in the local language. Please check with them.**

Frequently Asked Questions

For a current list of Frequently Asked Questions (FAQ), please visit http://www.EventReporter.com/faq/. The FAQ area is continuously being updated.

Customer Service System

Our Customer Service System is available at https://ticket.adiscon.com.

With it, you can quickly open a support ticket via a web-based interface. This system can be used to place both technical support calls as well as general and sales questions. We would appreciate if you select the appropriate category when opening your ticket.

Please note: the Customer Service System asks you for a User name or email and Password when you open it. If you do not have registered yet, you can simply click the “register now” button to do so. If you choose the continue “as guest” button you can also open a ticket without registering first.

Why using the Customer Service System? As you see further below, we also offer support by email. In fact, email is just another way to create a ticket in the Customer Service System. Whenever we reply to your ticket, the system automatically generates an email notification, which includes a link to your ticket as well as the answer we have provided. However, there are some situations where the support system should be used:

- **Email notifications do NOT include attachments!** If we provide an attachment, you must login into the ticket in order to obtain this. For your convenience, each email notification contains an active link that allows you to login immediately.
- **If you seem to not receive responses from us, it is a very good idea to check the web interface. Unfortunately, anti-SPAM measures are being setup more and more aggressive**. We are noticing an increasing number of replies that simply do not make it to your mailbox, because some SPAM filter considered it to be SPAM and removed it. Also, it may happen that your support question actually did not get past our own SPAM filter. We try very hard to avoid this. If we discard mail, we send a notification of this, so you should at least have an indication that your mail did not reach us. Using the customer support system via its own web interface removes all SPAM troubles. So we highly recommend doing this if communication otherwise seems to be disturbed. In this case, please remember that notification emails may also get lost, so it is a good idea to check your ticket for status updates from time to time.

Email

Please address all support requests to service@adiscon.com. An appropriate subject line is highly appreciated.

Please note: we have increasingly seen problems with too-aggressive SPAM filtering, resulting in loss of our replies. If you do not receive a response from us within two working days, we highly recommend re-submitting your support call via the** Customer Service System.

Phone

Phone support is limited to those who purchased support incidents. If you are

interested in doing so, please contact us via the Customer Service System for further details.

EventReporter Web Site

Visit the support area at http://www.EventReporter.com/help/support/ for further information. If for any reason that URL ever becomes invalid, please visit https://www.adiscon.com for general information.

Software Maintenance

Adiscon’s software maintenance plan is called upgradeinsurance. It offers unlimited free upgrades and priority support during its duration. It can be purchased for a period between 1 and 5 years.

Click here to learn more about UpgradeInsurance.

Non-Technical Questions

For non-technical questions please contact us via the Customer Service System.

Product Updates

The MonitorWare line of products is being developed since 1996. New versions and enhancements are made available continuously.

Please visit www.EventReporter.com for information about new and updated products.

Purchasing

All EventReporter features can be used for 30 days after installation without a license. However, after this period a valid license must be purchased. The process is easy and straightforward.

The License

The end user license agreement (EULA) is displayed during setup. If you need to receive a copy of the license agreement, please contact us via the Customer Service System. The license agreement of the current version can be also found on the web site of the product: https://www.EventReporter.com/EventReporter-eula/

Which Edition is for Me?

Information on all available EventReporter editions can be found on the web at the following URL. This includes a feature comparison.

https://www.EventReporter.com/edition-comparision/

Pricing & Ordering

Please visit https://www.EventReporter.com/products-prices/order-now/ to obtain pricing information. This form can also be used for placing an order online. If you would like to place a purchase order, please visit https://www.adiscon.com/purchase-orders/ to obtain details.

If you would like to receive assistance with your order or need a quote, please contact us via the Customer Service System.

Articles

Here you find articles about EventReporter:

Difference between Set Status - Set Property Action

The difference is, that a property is a part of the message object, while the status is a global object. That means that a property will change with every message, e.g. the timestamp or msg property. So properties are the actual different values of a message. Values are gone once the message is fully processed.

A status on the other hand is global and stays the same across all messages if not altered. A status variable can be filtered, but its value cannot be emitted in a message.

A setup for using a custom status object could be like this: Message “x” gets processed every now and then. Every time the message comes through the status variable “y” can be increased by 1. Once the status reaches a value of “n” a special message with properties from message “x” can be sent via email and the status variable “y” will be reset.

So, while setup looks similar, these are actually very different in their concept.

1. Initializing ‘status’: This is usually done when needed in the processing flow with “Set Status”.
2. How to set the status value again to initial value if I want to count up from the start: If you have a condition that waits for a specific status value to be reached, you can simply use another “Set Status” action after the output action.
3. Simple question: Status value should be numeral value? You can use numerical as well as alphabetical values for a status. But, I suppose it is best to only use something like boolean-like words when using alphabetical values, like on/off or yes/no. this is mostly because of logical understanding. Numerical values should be used for counters of course.

Include Event ID in Syslog message while forwarding to a Syslog server

We are using MonitorWare Agent / EventReporter to forward Windows Event logs to a Syslog server . The resulting syslog message does not have the Event IDs in them. How can we make Event ID part of the actual Syslog message?

One of the proposed solution would be to forward the Event Log messages using SETP Server . The resulting message would have the Event IDs in them. Click here to know the difference between SETP and Syslog!

But there are other ways to include the Event ID even without using setp (which is obviously not an option if you would like to send to a non-Adiscon backend).

So you can do one of the following:

Use XML Format - This is the best recommended option. With XML format, you get

everything about this event and you get it in a well-structured way. It includes all of the properties described in our event properties reference. To enable XML format, simply check “Use XML to Report” in the forward syslog options Action. Use Custom Format - In the “Forward Syslog” action, you can specify your own custom format in the “Message Format” text box. By default it is set to %msg%, but you can include whatever you like. Use the “Insert” link to do this (or simply type it)! Be sure to read the property replacer documentation to see the full power. This option is a good one, especially if you intend to parse the data because *you* can exactly specify what you would like to see. Use MoniLog Format - This is our former legacy format. It includes a bunch of useful information, but it has a number of anomalies, which might hit you in few cases when parsing. We do not recommend it, but if you would like to use it, you can select the “Insert” link in the “Forward Syslog” action properties. Then, select “Replace with MoniLog Format”. It will generate a custom format of the type given below. Again, we do not recommend this, but it is a way.
`## %severity% %timereported:::uxTimeStamp%: %source%/%sourceproc% (%id%) - "%msg%" ##`
`.`

Change Event Log Monitor Settings - You could also change the Event Log Monitor

itself to generate the legacy format. Then, you do not need to change the “Forward Syslog” action’s settings. The big drawback is that now the Event Log Monitor does emit an old format, which is not meant to be processed by any other MonitorWare product. If you just use the product as a back-end for your own front-end, this is not an issue. Anyhow, we still recommend to go for approach #3 instead of this. If you absolutely want to do it this way, this is how it is done: Go to the event log monitor properties. Click on the “Advanced Options” button. Check the “Use Legacy Format” checkbox. This will enable some other checkboxes. Review the options to see which of these you want.

We have provided the options at hand. We *strongly* recommend to go for either option 1 or 2. If you choose option 3 or 4, you can receive a parsing error from time to time. However this has been solved after introducing the newer formats.

As a general hint, you may want to take into account that Windows Event Log messages can become rather lengthy. They often go over the syslog RFC size of 1024 bytes. If you run a non-Adiscon Syslog server, you need to ensure it can receive such large messages, because otherwise some information might be missing (with option 2, you can customize what you would like to be missing in such cases - by limiting the size of %msg% via the property replacer).

How can I use a second sound card with the Play Sound Action?

I have got a second sound card on my machine, how can I use it with the Play Sound Action?

PlaySounds action plays a sound on the local machine. It is possible to play wave files and some other “system” supported sound files. This does “NOT” include mp3 files. As MonitorWare Agent is usually running “as a” System service, there are some things which needed to be noted!

On machines with more then ONE sound card, the MonitorWare Agent Service will take the “first active installed soundcard as output device regardless what is configured”.

If there is a need to play the sound on another sound card instead of the first active installed one, then there are two workarounds:

1. Specify a “User Account” for the Service which has a local profile where the sound card you want to use configured as primary playback device.
2. Run the MonitorWare Agent Service in console mode using the “-r” switch under a user account which has the sound card you want to use configured as primary playback device.

By following the above mentioned work around, you would be able to use second sound card (even x sound cards where x is user configurable) with the Play Sound Action.

The following things are user configurable in the Play Sound Action

Filename of the Soundfile - A full path and filename to the wave file which will be played. If the sound file specified here cannot be found or is not a valid wave file, a simple system beep will be played.

Playcount: Default is 1 - can be configured up to 100 times.

Delay between the sound plays - Only useful if the sound is played more then once. Between each play, MonitorWare Agent will wait for this time until it plays the sound again.

Note: A prior running sound will be aborted when this action is executed.

Default Timevalues Setting in EventReporter/MonitorWare Agent/WinSyslog explained

- Created: 2008-01-24 Andre Lorbach
- Updated: 2020-10-05 adisconteam

The general options of each product (EventReporter, MonitorWare Agent and WinSyslog) contain a setting for the “Default Timevalues”. This setting can be set to Localtime and UTC (Universal Coordinated Time) which is default.

If you switch this setting to Localtime, you may wonder why output timevalues still are in UTC.

Internally we need to calculate with UTC time. This is needed in order to maintain the time values if they are send via Syslog or SETP. If we wouldn’t do this, this could result to unexpected time differences.

So where does this setting have an effect?

- Send Email Action: The date in the email header is affected
- Start Program Action: Time parameters in the command line are affected
- Write File Action: Time properties in the file name are affected
- Filter Engine: If you filter by weekday or time fields, localtime does affect the filter result

But how can I get localtime output?

We added two additional options into the property engine which can be applied on time based values for this purpose.

Property Option: **localtime** = converts the output of the timestamp into localtime:

`Sample: %variable:::localtime%`

Property Option: **uxLocalTimeStamp** = same output as uxTimeStamp, but localtime is used

`Sample: %variable:::uxLocalTimeStamp%`

further articles you find on adiscon.com :

- articles

and on EventReporter :

- EventReporter Articles

FAQ

Here you find FAQ about EventReporter:

Why are Logfiles sometimes not rotated in EventReporter 18.5 to 19.1?

This article explains why log files may sometimes not be rotated as expected in EventReporter versions 18.5 to 19.1, and provides solutions for this issue.

Background

In EventReporter versions 18.5 to 19.1, there are several factors that can interfere with log file rotation under certain circumstances. These issues were addressed in later versions with improved rotation handling.

The Problem

Users may experience inconsistent log file rotation behavior where:

- Some log files rotate successfully every day as scheduled
- Some log files rotate only partially (not every day)
- Some log files never rotate at all

This typically occurs when log rotation is scheduled at specific times (e.g., at 0:00 every day) or when using dynamic filenames with property replacer.

Root Cause

The issue can be related to several factors:

- **File Handle Management:** When dynamic filenames are used, file handles are cached internally to avoid excessive file open/close operations. If files become inactive for extended periods, the cached handles may not be properly managed during rotation.
- **Timing Issues:** The rotation process may conflict with file access patterns, especially when multiple processes are accessing log files simultaneously.
- **Resource Constraints:** In high-volume logging environments, system resources may become constrained, affecting the rotation process.

Affected Versions

This issue affects EventReporter versions 18.5 to 19.1. Later versions include improvements to the log rotation logic that handle these scenarios more reliably.

Solutions

Recommended Solution: Upgrade EventReporter

The most effective solution is to upgrade to EventReporter version 19.1 or later, where the log rotation logic has been improved to handle these scenarios more reliably.

Alternative Solutions

If upgrading is not immediately possible, try these workarounds:

1. **Adjust File Handle Management:** * Review your file logging configuration * Ensure property replacer settings in filenames are optimized * Consider simplifying dynamic filename patterns if possible
2. **Modify Rotation Timing:** * Schedule rotations during periods of lower activity * Avoid scheduling rotations at exactly 00:00 when possible * Consider using size-based rotation instead of time-based rotation
3. **Resource Optimization:** * Monitor system resources during peak logging periods * Ensure adequate disk space is available * Consider load balancing if multiple instances are logging simultaneously

Important: Monitor your logging environment closely when implementing these workarounds, as they may affect overall system performance.

Is EventReporter v19+ supported on Windows Server IoT 2025?

Overview

This FAQ answers whether EventReporter v19+ is supported on Windows Server IoT 2025 and outlines current guidance, functional status, and considerations for deployments, including Server Core.

Notes

Support Status

Official support:

- Windows Server IoT 2025 is not yet explicitly listed in the EventReporter v19+ platform matrix.

Functional status:

- EventReporter v19+ is known to function properly on Windows Server IoT 2025 (including Server Core) based on internal testing and field feedback.

Guidance for Server Core Deployments

Windows Server IoT 2025 Server Core does not provide a graphical user interface. For headless deployments, we recommend configuring EventReporter using Adiscon Config Files (`*.cfg`), a portable, file-based configuration format.

Recommended workflow:

1. Create the configuration on a GUI-enabled machine - Install EventReporter and open the Configuration Client - Configure rules, services, and actions as required - Export the configuration as Adiscon Config Files (`*.cfg`)
2. Transfer the configuration to Server Core - Copy the exported `.cfg` to the Server Core system (e.g., via PowerShell Remoting or SMB)
3. Enable File Config Mode and set paths via registry

   Registry path: `HKEY_LOCAL_MACHINE\SOFTWARE\Adiscon\EventReporter\Settings`

   Required values:

   - `szFileConfig` (REG_SZ): Example `c:\configs\eventreporter\central-server.cfg`
   - `szDataDirectory` (REG_SZ): Example `c:\configs\eventreporter\`
   - `iAccessMode` (REG_DWORD): `1` (enables file config mode)

> **Important**
>
> When running in file config mode, ensure the service account has read access to the configuration file and write access to the data directory.

Troubleshooting the Start Program action in EventReporter

This article explains common issues with the Start Program action in EventReporter and provides solutions to resolve them.

Background

The Start Program action allows EventReporter to execute external programs, batch files, or scripts when specific Windows event log conditions are met. However, there are several common issues that can prevent this action from working correctly.

Common Issues and Solutions

Issue 1: Program not found or path problems

Symptoms: - The Start Program action appears to run but nothing happens - No error messages in the Windows Event Log - The external program works when run manually from command line

Root Cause: EventReporter may not be able to locate the executable file due to path issues or missing dependencies.

Solutions:

1. **Use absolute paths for all executables** - Instead of: `curl google.com > temp.txt` - Use: `C:\curl\curl-win\bin\curl.exe google.com > C:\temp\temp.txt`
2. **Verify executable location** - Check if the program exists in the specified path - Ensure all required DLL files are present - Test the command manually from Windows Command Prompt
3. **Check Windows PATH environment variable** - EventReporter may not have access to the same PATH as your user session - Use full paths instead of relying on PATH resolution

Issue 2: Permission problems

Symptoms: - No error messages in Event Log - Program works when run manually but not through EventReporter

Root Cause: EventReporter runs as a Windows service with different permissions than your user account.

Solutions:

1. **Store files in accessible locations** - Avoid system folders like `C:\Windows\System32` - Use generic folders like `C:\temp` or `C:\scripts` - Ensure EventReporter service has read/execute permissions
2. **Check file permissions** - Right-click on the executable file - Go to Properties > Security - Ensure “SYSTEM” and “SERVICE” accounts have execute permissions

Issue 3: Working directory problems

Symptoms: - Program runs but cannot find input/output files - Relative paths in scripts don’t work

Root Cause: The working directory when EventReporter executes the program may be different from expected.

Solutions:

1. **Use absolute paths for all file references** - Instead of: `> temp.txt` - Use: `> C:\temp\temp.txt`
2. **Set working directory in batch files** - Add `cd /d C:\your\working\directory` at the beginning of batch files

Issue 4: Event-specific parameter passing

Symptoms: - Program runs but doesn’t receive expected parameters - Event data is not passed correctly to the external program

Root Cause: EventReporter uses specific replacement characters to pass event data to external programs.

Solutions:

1. **Use correct replacement characters** - `%d` - Date and time in local time - `%s` - Source system IP address or name - `%f` - Numeric facility code - `%p` - Numeric priority code - `%m` - The event message itself - `%%` - Represents a single % sign
2. **Quote parameters properly** - Use quotes around parameters that contain spaces - Example: `"Event occurred: %m"` instead of `Event occurred: %m`

Troubleshooting Steps

1. **Check Windows Event Log** - Open Event Viewer (type “Event Viewer” in Windows search) - Navigate to Windows Logs > Application - Look for EventReporter-related error events
2. **Test with simple commands first** - Start with a basic batch file that creates a text file - Example: `echo Test > C:\temp\test.txt`
3. **Verify the command works manually** - Open Command Prompt as Administrator - Run the exact same command that EventReporter should execute - Ensure it works from the command line first
4. **Check EventReporter service account** - Verify which account EventReporter is running under - Ensure that account has necessary permissions
5. **Test event triggering** - Create a test event that should trigger your Start Program action - Verify the event is being detected by EventReporter - Check if the action is configured correctly

Example Working Configuration

Here’s an example of a properly configured Start Program action for EventReporter:

Command to execute: `C:\scripts\process-event.bat`

Parameters: `"%d" "%s" "%m"`

Batch file content (C:scriptsprocess-event.bat): ``batch @echo off echo Event occurred at %1 from %2 >> C:\temp\event-log.txt echo Message: %3 >> C:\temp\event-log.txt ``

Key points: - Full path to batch file - Quoted parameters to handle spaces in event messages - Absolute paths for output files - Proper use of replacement characters

Additional Tips

- **Timeout settings:** Keep external programs under 5 seconds runtime for best performance
- **Error handling:** Consider adding error checking to your batch files
- **Logging:** Add logging to your scripts to help troubleshoot issues
- **Testing:** Always test Start Program actions with actual Windows events
- **Event filtering:** Ensure your event filters are correctly configured to trigger the action

If you continue to experience issues after following these steps, please contact Adiscon support with: - EventReporter version - Windows version - Exact command being executed - Any error messages from Event Log - Results of manual command testing - Sample event that should trigger the action

Is MariaDB supported by the ODBC action?

This article explains MariaDB support in ODBC database actions.

Question

Is MariaDB supported by the ODBC action?

Answer

Yes, MariaDB is fully supported by the ODBC action and can be used as a direct replacement for MySQL.

Background

MariaDB is a free and open-source alternative to MySQL. It is a fork of MySQL, initiated by the original MySQL developers after Oracle acquired Sun Microsystems (the former owner of MySQL). MariaDB was designed to be binary-compatible with MySQL, which generally makes switching from MySQL to MariaDB very easy.

Key characteristics of MariaDB:

- **Open Source:** MariaDB is consistently Open Source under a license that guarantees free use and further development
- **Binary Compatibility:** Designed to be binary-compatible with MySQL, making migration straightforward
- **Independent Development:** Continuous, independent development separate from MySQL
- **Performance:** Often preferred as an alternative due to sometimes better performance characteristics

Configuration

To use MariaDB with the ODBC action:

1. **Install MariaDB ODBC Driver:** - Download and install the MariaDB Connector/ODBC driver from the official MariaDB website - Ensure you install the correct version (32-bit or 64-bit) to match your Adiscon product installation
2. **Configure System DSN:** - Open the ODBC Data Source Administrator (use the 32-bit version if your product runs in 32-bit mode) - Create a new System DSN - Select the MariaDB ODBC driver - Configure the connection settings (server, database, credentials)
3. **Configure Database Action:** - In your Adiscon product configuration, select the ODBC Database action - Choose the MariaDB System DSN you created - Test the connection using the “Verify Database” button - Create the database tables if needed using the “Create Database” button

Note: The configuration process is identical to configuring MySQL, as MariaDB uses MySQL-compatible drivers and protocols.

Additional Information

For more information about database actions, see the ODBC Database Options documentation in your product's manual.

For MariaDB-specific information, visit the official MariaDB website.

further FAQ you find on adiscon.com :

- FAQ

and on EventReporter :

- EventReporter FAQ: general, licensing and installation and updates

References

The following references provide in-depth information to some very specific things. You may want to review them if you are looking for one of these. Some references are placed on the web and some other are directly contained in this manual. We decided to provide web-links wherever we considered them useful.

- Formats (XML and Database)
- Version History
- The EventReporter Service

Comparison of properties

Available in MonitorWare Agent, EventReporter and WinSyslog

The property replacer is a reference - the actual properties are very depending on the edition purchased. We have just included information on what is available in which products for your ease and convenience.

Properties Available	MonitorWareAgent	WinSyslog	EventReporter
Standard Property	Yes	Yes	Yes
Windows Event Log	Yes		Yes
Syslog Message	Yes	Yes	
Disk Space Monitor	Yes		
File Monitor	Yes		
Windows Service Monitor	Yes		Yes
Ping Probe	Yes		
Port Probe	Yes		
Database Monitor	Yes		
Serial Port Monitor	Yes		
MonitorWare Echo Request	Yes		
System	Yes	Yes	Yes
Custom	Yes	Yes	Yes
NNTP Probe	Yes		
HTTP Probe	Yes		
FTP Probe	Yes		
SMTP Probe	Yes		
POP3 Probe	Yes		

Event Properties

Events have certain properties, for example the message associated with the event or the time it was generated. Each of this properties has an assigned name. The actual properties available depend on the type of event. The following sections describe both how to access properties as well as properties available.

Knowing about event properties is important for building complex filter conditions, customized actions as well as for integrating into a third-party system. Event properties provide a generic way to look at and process the events generated. Thus we highly recommend that you at least briefly read this reference section.

Accessing Properties

Properties are accessed by their name. The component used for this is called the “property replacer”. It is a generic component that allows you to merge properties from the event processed to e.g. the email subject line or a log file line. It is a central component that is used as often in the product as possible. The idea behind the property replacer is that there is often need to specify a value from the event processed.

The property replacer provides very powerful ways to access the properties: they cannot only be accessed as one full property. They can also be accessed as substrings and even be reformatted. As such, the property replacer provides a specific syntax to access properties:

`%property:fromPos:toPos:options%`

The percent-signs (“%”) indicate the start of a special sequence. The other parameters have the following meanings

FromPos and ToPos can be used to copy a substring from a lengthy property. The options allow to specify some additional formatting.

Within the properties, all time is based on UTC regardless if your preferred time is UTC or localtime. So if you want to display localtime instead of UTC, you have to use the following syntax: `%variable:::localtime%`

Property

This is the name of the property to be replaced. It can be any property that a given event possesses. If a property is selected that is empty for the event processed, an empty string is returned.

A property is either an event property, a custom property, a dynamic property or a system property.

If a property is selected that is not present, the result will always be an empty string, no matter which other options have been selected.

FromPos

If you do not want to use the full string from the property, you can specify a start position here. There are two ways to specify the start location:

Fixed Character position

If you know exactly on which position the string of interest begins, you can use a fixed location. In this case, simply specify the character position containing the first character of interest. Character positions are counted at 1.

Search Pattern

A search pattern is specified as follows:

/<search-pattern>/<options>

If a search pattern is specified, the property value is examined and the first occurrence of <search-pattern> is detected. If it is not found, nothing is returned. If it is found, the position where the pattern is found is the start position or, if the option “$” is specified, the position immediately after the pattern.

The search pattern may contain the “?” wildcard character, which represents any character. Other wildcards are not supported with the property replacer.

Please note that a slash inside the search pattern will terminate the search field. So pure slashes cannot be used. However, they can be escaped by prefixing them with a backslash (). The same applies to the ‘?’ character. For example, if you intend to search for `"http://"` inside a search pattern, you must use the following search string: `"/http:////"`.

Default Value

If the FromPos is not specified, the property string is copied starting at position 1.

ToPos

If you do not want to use the full string from the property, you can specify the highest character position to be copied here.

Absolute Position

Specify a simple integer if you would like to specify an absolute ending position.

Relative Position

This is most useful together with the search capabilities of FromPos. A relative position allows you to specify how many characters before or after the FromPos you would like to have copied. Relative positions are specified by putting a plus or minus (“+”/”-”) in front of the integer.

Please note: if you specify a negative position (e.g. -20), FromPos and ToPos will internally be swapped. That is the property value will not be (somehow) reversely copied but they will be in right order. For example, if you specify `%msg:30:-20%` actually character positions 10 to 30 will be copied.

Search Pattern

Search pattern support is similar to search pattern support in FromPos.

A search pattern is specified as follows:

/<search-pattern>/<options>

If a search pattern is specified, the property value is examined and the first occurrence of <search-pattern> is detected. The search is only carried out in the string that follows FromPos. If the string is not found, nothing is returned. If it is found, the position where the pattern is found is the ending position or, if the option “$” is specified, the position immediately after the pattern.

The search pattern may contain the “?” wildcard character, which represents any character. Other wildcards are not supported with the property replacer.

Please note that a slash inside the search pattern will terminate the search field. So pure slashes cannot be used. However, they can be escaped by prefixing them with a backslash (). The same applies to the ‘?’ character. For example, if you intend to search for `"http://"` inside a search pattern, you must use the following search string: `"/http:////"`.

Search Example

A common use case is to combine searches in ToPos and FromPos to extract a substring that is delimited by two other strings. To do so, use search patterns in both fields. An example is as follows: assume a device might generate message in the form `"... error XXX occurred..."` where “…” represents additional message text and XXX the actual error cause. You would like to extract the phrase `"error XXX occurred"`. To do so, use the following property replacer syntax: `%msg:/error/:/occurred/$/%`

Please note that the FromPos is used without the $-option, while in ToPos it is used. If it hadn’t been used in ToPos, only the part “error XXX “ would have been extracted, as the ToPos would point to the last character before the search string.

Similarly, if only “ XXX “ should be extracted, the following syntax might be used:

`%msg:/error/$:/occurred/%`

If you would also like to remove the spaces (resulting in just “XXX”), you must include them into the search strings:

`%msg:/error /:/ occurred/$/%`

Default

If not specified, the ending position will be the last character.

Options

Options allow you to modify the contents of the property. Multiple options can be set. They are comma-separated. If conflicting options are specified, always the last option will be in effect (e.g. specifying “uppercase,lowercase” will lead to lowercase conversion of the property value).

The following options are available with this release of the product:

lowercase

All characters in the resulting property extract will be converted to lower case.

uppercase

All characters in the resulting property extract will be converted to upper case.

uxTimeStamp

This is a special switch for date conversions. It only works if the extracted property value is an ISO-like timestamp (YYYY-MM-DD HH:MM:SS). If so, it will be converted to a Unix-like ctime() timestamp. If the extracted property value is not an ISO-like timestamp, no conversion happens.

uxLocalTimeStamp

This is the same as uxTimeStamp, but with local time instead of GMT.

date-rfc3339

This option is for replacing the normal date format with the date format from RFC3339.

date-rfc3164

This option is for replacing the normal date format with the date format from RFC3164.

date-rfc3164strict

Does the same as date-rfc3164 but when the date is below 10, two spaces will be added between Month and day (Which is defined in rfc3164).

escapecc

Control characters* in property are replaced by the sequence `##hex-val##`, where* hex-val is the hexadecimal value of the control character (at least two digits, may be more).

spacecc

Control characters* in the property are replaced by spaces. This option is most* useful when a message contains control characters (e.g. a Windows Event Log Message) and should be written to a log file.

compressspace

Compresses multiple consecutive space characters into a single one. The result is a string where all words are separated by just single spaces. To also compress control characters, use the compressspace and spacecc options together (e.g.``'%msg:::spacecc,compressspace%'`). Please note that space compression happens on the final substring. So if you use the FromPos and ToPos capabilities the substring is extracted first and then the space compression applied. For example, you may have the msg string"1 2". There are two space between 1 and 2. Thus, the property replacer expression:``%msg:1:3:compressspace%

will lead to `"1 "` ('1' followed by two spaces). If you intend to receive `"1 2"``('1' followed by one space, followed by '2'), you need to use ``%msg:1:4:compressspace%`

or

`%msg:1:/2/$:compressspace%`

In the second case, the exact length of the uncompressed string is not known, thus a search is used in topos to obtain it. The result is then space-compressed.

compsp

Exactly the same as compressspace, just an abbreviated form for those that like it brief.

csv

For example `%variable:::csv%`. This option will create a valid CSV string. For example a string like this:`this is a "test"!` becomes this `"this is a ""test""!"` where quotes are replaced with double quotes.

cef

Convert string content into valid McAfee CEF Format. This means that =``will be replaced with=`` and \ will be replaced with \. **convgermuml**

Converts German Umlaut characters to their official replacement sequence (e.g. “ö” –> “oe”)

localtime

Now you can print the Time with localtime format by using``%variable:::localtime%``

nomatchblank

If this is used, the Property Replacer will return an empty string if the frompos or topos is not found.

replacepercent

This option replaces all % occurrences with a double %%, which is needed for the property replacer engine in case that a string is reprocessed. This is needed because the percent sign is a special character for the property replacer. Once the property is processed, the double``%%``become automatically one``%``. **toipv4address**

Property string will be converted into IPv4 Address format if possible.

toipv6address

Property string will be converted into IPv6 Address format if possible.

crlftovbar

Does the same as date-rfc3164 but when the date is below 10, two spaces will be added between Month and day (Which is defined in rfc3164).

removecc

Removes all control characters from 0x00 to 0x1F

replacechar

Replaces a single character with another single character.

How ASCII characters are being handled:

Sample:`%msg:$x:$y:replacechar%`

Broken down:

`%msg:$``<- Tells property replacer that a character is being expected (At the moment only for REPLACECHAR Option). ``x``<- The character to search for ``:`

`$``<- Tells property replacer that a character is being expected (At the moment only for REPLACECHAR Option). ``y``<- The character to replace with ``:`

`replacechar%`

How special characters are handled?

Sample: `%msg:$\n:$|:replacechar%`

`%msg:$` <- Tells property replacer that a character is being expected (At the moment only for REPLACECHAR Option). `\n` <- The character to search for special character, possible values: t for tab,

n for newline,

v for verticaltab,

f for formfeed,

r for carriage return

for an actual backslash.``:``

`$``<- Tells property replacer that a character is being expected (At the moment only for REPLACECHAR Option). ``|``<- The character to replace with ``:`

`replacechar%`

* = control characters like e.g. carriage return, line feed, tab, …*

Important: All option values are case-sensitive. So “uxTimeStamp” works while “uxtimestamp” is an invalid option!

Simple Examples

A good example for this is the email subject line, which has severe length constraints. If you would like to have only the first 40 characters of the actual message text in the subject, you could use the replacer: `"%msg:1:40%"`. If you know the first 10 characters of the message are meaningless for you but you would like to see the full rest of the message (no matter how long it may be), you can use a sequence like `"%msg:11%"`.

If you would just like to see the plain message from beginning to end, you can simply omit frompos and topos: `"%msg"`. Of course, all of these sample not only work with the “msg” property, but also with all others like “facility”, or “priority”, or W3C-log header extracted property names.

More complex Examples

If you would like to extract the 50 characters from the message after the word DROP, you would use the following replacer string:`%msg:/DROP/$:+50%`

If you would like to have the first 40 characters in front of the string “- aborted” (including that string):

`%msg:/- aborted/$:-40%`

If you would like to receive everything starting from (and including) “Log:”:

`%msg:/Log/%`

If you would like to have everything between the string “FROM” and “TO” including NONE of the both searchstrings:

`%msg:/FROM/$:/TO/%`

If you would just like to log lowercase letters in your log messages:

`%msg:::lowercase%`

And if you would just like to have the first 50 characters (and these in lower case):

`%msg:50:::lowercase%`

If you need to change a timestamp to a UNIX-like timestamp, you could use this:

`%datereceived:::uxTimeStamp%`

Please see also the focused sample in the topos description.

A real world Sample

We use the following template to generate output suitable as input for MoniLog:

```
%timegenerated:1:10%,%timegenerated:12:19%,%source%,%syslogfacility%,%syslogprio  rity
%,EvntSlog: %severity% %timereported:::uxTimeStamp%: %source%/%sourceproc% (%id%) - "
%msg%"%$CRLF%
```

Please note: everything is on one line with no line breaks in between. This example is from the “write to file” action (with custom file format).**

System Properties

System properties are special sequences that can be helpful. They are available with all event types. They are:

$CRLF

A Windows newline sequence consisting in the characters CR and LF. If you just need CR, you can use `%$CRLF:1:1%` and if you need use LF you can use `%$CRLF:2:2%`

$TAB

An US-ASCII horizontal tab (HT, 0x09) character

$HT

same as $TAB

$CR

A single US-ASCII CR character (shortcut for `%$CRLF:1:1%`) **$LF**

A single US-ASCII LF character (shortcut for``%$CRLF:2:2%``) **$xNN**

A single character, whose value (in hexadecimal) is given by NN. NN must be two hexadecimal digits - a leading zero must be used if a value below 16 is to be represented. The value 0 (%x00) is invalid and - if specified - replaced by the “?” character.

As an example, $CR could also be expressed as %$x0d%.

Please note that only one character can be represented. If you need to specify multiple characters, you need multiple $xNN sequences. An example may be $CRLF which could also be specified as %$x0d%%$x0a% (but not as %$x0d0a%).

$NOW

Contains the current date and time in the format:`YYYY-MM-DD HH.MM.SS`

Please note that the time parts are delimited by `'.'` instead of `':'`. This makes the generated name directly suitable for file name generation.

If you need just parts of the timestamp, please use the property replacer's substring functionality to obtain the desired part. Use``%$NOW:1:4% to get the year,``

`%$NOW:6:7% to get the month,`

…

`%$NOW:1:10% to get the full datestamp,`

`%$NOW:12:20% to get the full timestamp`

$NEWUUID

Creates a new UUID (Universally Unique Identifiers), a unique 128-bit integer represented as a 32 digit hexadecimal number.

Custom Properties

Users can create an unlimited number of custom properties. These can be created with for example the "PostProcess" action (if the product edition purchased supports this action).

Custom properties can theoretically have any name, but Adiscon highly recommends to prefix them with "u-" (e.g. "u-MyProperty" - "u" like "user"). This ensures that no compatibility problems will arise in current and future versions of the software. Adiscon guarantees that it will never use the "u-" prefix for Adiscon-assigned properties.

Custom properties can be used just like regular properties. Wherever you can specify a property, you can also specify a custom property.

Event-Specific Properties

Each network event is represented by a so-called "Event Record" (sometime also named an "InfoUnit", an "Unit of Information"). Data obtained from all services will end up as an event. For example, Windows Event Log data, syslog data, and a file line obtained by the file monitor will all be an event. That kind of generalization make it easy to deal with all of these events in a consistent way.

Each event has a set of properties which in turn have values. For example, there is a property named "source" and it will always contain an indication of which system the event originated on. Obviously, not every event source does support all properties. For example, a syslog message does not contain a Windows Event ID - simply because there is no such thing as an event ID in syslog. So, depending on the type of event, it may contain different properties.

In order to make the product really generally useful, some few properties have been defined in a generic way and are guaranteed to be present in every event, no matter what type it may have. Sometimes this is a "natural" common property, like the "fromhost". Sometimes, though, it may look a bit artificial. An example of the later is the "syslogfacility" property. It is guaranteed to be present in every event - but actually this is a syslog-only thing. The non- syslog event sources either emulate this property (in a consistent manner) or allow the user to configure a syslogfacility that should be used for all events generated by that service. At the bottom line, this will ensure that the property is available in all events and - given proper configuration - that can be extremely helpful for the administrators to set up things in a powerful and generic way.

Standard Properties

As outlined under Event Properties, these are properties present in all types of events. Some event types have only these standard properties. Others have additional properties. Those with additional properties are documented in the other sections. If there is no specific documentation for a specific event type, this means that it supports the standard properties, only.

msgPropertyDescribed

A human-readable representation of the message text. While this is generally available, the exact contents largely depends on the source of the information. For example, for a file monitor it contains the file line and for a syslog message it contains the parsed part of the syslog message.

source

The source system the message originated from. This can be in various representations (e.g. IP address or DNS name) depending on configuration settings.

localhostname

On service startup it is automatically set to the local system computer name. It is read only and can be used if source property is not usable. E.g. if the Source property cannot be translated to IP format because the event log entry was recorded with an old computer name that no longer exists.

resource

A user-assigned numerical value. Does not have any specific meaning. Primarily intended for quick filtering.

CustomerID

A user-assigned numerical value. Does not have any specific meaning. Primarily intended for quick filtering.

SystemID

A user-assigned numerical value. Does not have any specific meaning. Primarily intended for quick filtering.

timereported

The time the originator tells us when this message was reported. For example, for syslog this is the timestamp from the syslog message (if not configured otherwise). Please note that timereported eventually is incorrect or inconsistent with local system time - as it depends on external devices, which may not be properly synchronized.

For Windows Event Log events, timereported contains the timestamp from the event log record.

timegenerated

The time the event was recorded by the service. If messages are forwarded via SETP, this timestamp remains intact.

importance

Reserved for future use.

iut

Indicates the type of the event. Possible values are:

```
1- syslog message
2- heartbeat
3- Windows Event Log Entry
4- SNMP trap message
5- file monitor
8- ping probe
9- port probe
10- Windows service monitor
11- disk space monitor
12- database monitor
13- serial device monitor
```

iuvers

Version of the event record (info unit). This is a monitorware internal version identifier.

Windows Event Log Properties

id

Windows Event ID

severity

severity as indicated in the event log. This is represented in string form. Possible values are:

```
[INF] - informational
[AUS] - Audit Success
[AUF] - Audit failure
[WRN] - Warning
[ERR] - Error
[NON] - Success (called "NON" for historical reasons)
```

severityid

The severity encoded as a numerical entity (like in Windows API)

sourceproc

The process that wrote the event record (called “source” in Windows event viewer).

category

The category ID from the Windows Event Log record. This is a numerical value. The actual value is depending on the event source.

catname

The category name from the Windows Event Log record. This is a string value. The actual value is depending on the event source. This value is a textual representation from the Category ID. This property could contain line feeds, which can be removed by activating the option “Remove Control Characters from String Parameters” in the advanced options of the EventLog Monitor Service.

user

The user name that was recorded in the Windows Event Log. This is “NA” if no user was recorded.

NTEventLogType

The name of the Windows Event Log this event is from (for example “System” or “Security”).

bdata

Windows Event Log records sometimes contain binary data. The Event Log Monitor service can be set to include this binary data into the event, if it is present. If it is configured to do so, the binary data is put into the “bdata” property. Every byte of binary data is represented by two hexadecimal characters.

Please note that it is likely for bdata not to be present. This is because the binary data is seldom used and very performance-intense. (%id%) - “%msg%”%$CRLF%

Windows Event Log V2 Properties

id

Windows Event ID

severity

severity as indicated in the event log. This is represented in string form. Possible values are:

```
[INF] - informational
[AUS] - Audit Success
[AUF] - Audit failure
[WRN] - Warning
[ERR] - Error
[NON] - Success (called "NON" for historical reasons)
```

severityid

The severity encoded as a numerical entity (like in Windows API)

sourceproc

The process that wrote the event record (called “source” in Windows event viewer).

category

The category ID from the Windows Event Log record. This is a numerical value. The actual value is depending on the event source.

catname

The category name from the Windows Event Log record. This is a string value. The actual value is depending on the event source. This value is a textual representation from the Category ID. This property could contain line feeds, which can be removed by activating the option “Remove Control Characters from String Parameters” in the advanced options of the EventLog Monitor Service.

user

The user name that was recorded in the Windows Event Log. This is “NA” if no user was recorded.

nteventlogtype

The name of the Windows Event Log this event is from (for example “System” or “Security”).

channel

The channel property for event log entries, for classic Event logs they match the `%nteventlogtype%` property, for new event logs, they match the “Event Channel”.

sourceraw

This contains the full internal name of the event source for new event logs, for classic event logs it contains the same value as in %sourceproc%.

level

Textual representation of the event log level (which is stored as a number in `%severityid%`). This property is automatically localized by the system.

categoryid

Internal category id as number.

keyword

Textual representation of the event keyword. This property is automatically localized by the system.

user_sid

If available, contains the raw SID of the username (`%user%`) property.

recordnum

Contains the internal event record number. Please note that if the event log has been truncated before, it may not start with 0 or 1 but a higher number.

Syslog Message Properties

rawsyslogmsg

The message as it was received from the wire (unparsed).

syslogfacility

The facility of a syslog message. For non-syslog messages, the value is provided based on configuration. In essence, this is simply an integer value that can be used for quick filtering inside your rules.

syslogfacility_text

The facility of a syslog message. This property is automatically created by using the syslogfacility properly and set to these values: `"Kernel", "User", "Mail", "Daemons", "Auth", "Syslog", "Lpr", "News", "UUCP", "Cron", "System0", "System1", "System2", "System3", "System4", "System5", "Local0", "Local1", "Local2", "Local3", "Local4", "Local5", "Local6", "Local7"`

syslogpriority

The severity of a syslog message. For non-syslog messages, this should be a close approximation to what a syslog severity code means.

syslogpriority_text

The severity of a syslog message. This property is automatically created by using the syslogpriority properly and set to these values: `"Emergency", "Alert", "Critical", "Error", "Warning", "Notice", "Informational", "Debug"`

syslogtag

The syslog tag value, a short string. For non-syslog messages, this is provided based on configuration. In most cases, this is used for filtering.

syslogver

Contains the syslog version number which will be one or higher if a rfc 5424 valid message has been received, or 0 otherwise

syslogappname

Contains the appname header field, only available if the Syslog message was in rfc 5424 format. Otherwise, this field will be emulated by the %syslogtag% property

syslogprocid

Contains the procid header field, only set if the Syslog message was in rfc 5424 format.

syslogmsgid

Contains the msgid header field, only set if the Syslog message was in rfc 5424 format.

syslogstructdata

Contains the structdata header field (in raw format), only set if the Syslog message was in rfc 5424 format.

syslogprifac

Contains combined syslog facility and priority useful to build your own custom syslog headers

Disk Space Monitor

currusage

The currently used disk space.

maxavailable

The overall capacity of the (logical) disk drive.

CPU/Memory Monitor

wmi_type

This variable is a string and can be one of the following variables: cpu_usage, mem_virtual_usage, mem_physical_usage, mem_total_usage.

cpu_number

Number of the current checked CPU.

cpu_load

The workload of the CPU as number, can be 0 to 100.

mem_virtual_load

How much virtual memory is used (MB).

mem_virtual_max

How much virtual memory is max available (MB).

mem_virtual_free

How much virtual memory is free (MB).

mem_physical_load

How much physical memory is used (MB).

mem_physical_max

How much physical memory is max available (MB).

mem_physical_free

How much physical memory is free (MB).

mem_total_load

How much total(Virtual+Physical) memory is used (MB).

mem_total_max

How much total(Virtual+Physical) memory is max available (MB).

mem_total_free

How much total(Virtual+Physical) memory is free (MB).

File Monitor

genericfilename

The configured generic name of the file being reported.

generatedbasefilename

Contains the generated file name without the full path.

Special IIS LogFile Properties

The Logfile Fields in IIS Logfiles are customizable, so there is no hardcoded command for their use.

The property-name depends on its name in the logfile. For example we take this Logfile:

```
#Software: Microsoft Internet Information Services 5.0
#Version: 1.0
#Date: 2005-10-27 14:15:25
#Fields: date time c-ip cs-username s-ip s-port cs-method cs-uri-stem
cs-uri-query sc-status cs(User-Agent)
2005-10-27 14:15:16 127.0.0.1 - 192.168.0.1 443 POST /eCommerce/asdf.php
2005-10-27 14:15:16 127.0.0.1 - 192.168.0.1 443 POST /eCommerce/asdf.php
2005-10-27 14:15:16 127.0.0.2 - 192.168.0.1 443 POST /eCommerce/asdf.php
2005-10-27 14:15:16 127.0.0.2 - 192.168.0.1 443 POST /eCommerce/asdf.php
```

As you can see, in our sample the fields are named: date, time, c-ip, cs-username, s-ip, and so on.

To use them as a Property inside our MonitorWareProducts, just use the names from your Logfile and add a “p-” before it:

p-date

The Date on which the Event occurs

p-time

The Time on which the Event occurs

p-c-ip

The IP address of the User which accessed

p-cs-username

The Username of the User which accessed

p-s-ip

The Server IP

p-s-port

The Server Port

p-cs-method

The Client-Server Method (POST,GET)

p-cs-uri-stem

The accessed File including its path

Windows Service Monitor

sourceproc

The name of the service whose status is being reported (from the Windows service registry).

Ping Probe

echostatus

Status returned for the echo request

The status value can be one of the following:

```
0 = IP_SUCCESS
11002 = IP_DEST_NET_UNREACHABLE
11003 = IP_DEST_HOST_UNREACHABLE
11010 = IP_REQ_TIMED_OUT
11013 = IP_TTL_EXPIRED_TRANSIT
11016 = IP_SOURCE_QUENCH
11018 = IP_BAD_DESTINATION
```

roundtriptime

Round trip time for the ping packet (if successful)

Port Probe

responsestatus

The status of the probe.

responsemsg

The response message received (if any)

Database Monitor

Database-Monitor created events are a bit different than other events. The reason is that the database fields themselves become properties - but obviously these are not fixed but depend on what you monitor.

All queried data fields are available as properties via their database field name **prefixed with “db-”**.

An example to clarify: we assume the following select statement is used for the database monitor:

select name, street, zip, city from addresses

There is also an ID column named “ID”. So the event generated by this database monitor will have the following specific properties:

- db-ID
- db-name
- db-street
- db-zip
- db-city

These properties will contain the field values as they are stored in the database. Please note that NULL values are translated into empty strings (“”), so there is no way to differentiate a NULL value from an empty string with this version of the database monitor.

Other than the custom “db-” properties, no specific database monitor properties exist.

Serial Monitor

portname

The name of the port that the data originated from (typical examples are COM1, COM2). The actual name is taken from the configuration settings (case is also taken from there).

MonitorWare Echo Request

responsestatus

The status of the echo request. Possible values:

```
0 - request failed (probed system not alive)
1 - request succeeded
```

If the request failed, additional information can be found in the * msg* standard property.

FTP Probe

ftpstatus

The status of the connection.

ftprespmsg

The response of the connection.

IMAP Probe

imapstatus

The status of the connection.

imaprespmsg

The response of the connection.

NNTP Probe

nntpstatus

The status of the connection.

nntprespmsg

The response of the connection.

SMTP Probe

smtpstatus

The status of the connection.

smtprespmsg

The response of the connection.

POP3 Probe

pop3status

The status of the connection.

pop3respmsg

The response of the connection.

HTTP Probe

httpstatus

The status of the connection.

httprespmsg

The response of the connection.

Complex Filter Conditions

The rule engine uses complex filter conditions.

Powerful boolean operations can be used to build filters as complex as needed. A boolean expression tree is graphically created. The configuration program is modeled after Microsoft Network Monitor. So thankfully, many administrators are already used to this type of Interface. If you are not familiar with it, however, it looks a bit confusing at first. In this chapter, we are providing some samples of how boolean expressions can be brought into the tree.

Example 1

In this example, the message text itself shall be checked. If it contains at least one of three given strings, the filter should become true. If none of the string is found, the boolean expression tree evaluates to false, which means the associated action(s) will not be executed.

In pseudo-code, the filter could be written like this:

```
If (msg  = "DUPADDRES") OR (msg = "SPANTREE") OR (msg = "DUPLEX_MISMATCH) then
   execute action(s)
end if
```

Please note: in the example, we have abbreviated “message” to just “msg”. Also note that for brevity reasons we use the equals (“=”) comparison operator, not the contains. The difference between the equals and the contains operator is that with “contains”, the string must just be part of the message.

In the filter dialog, this pseudo code looks as follows:

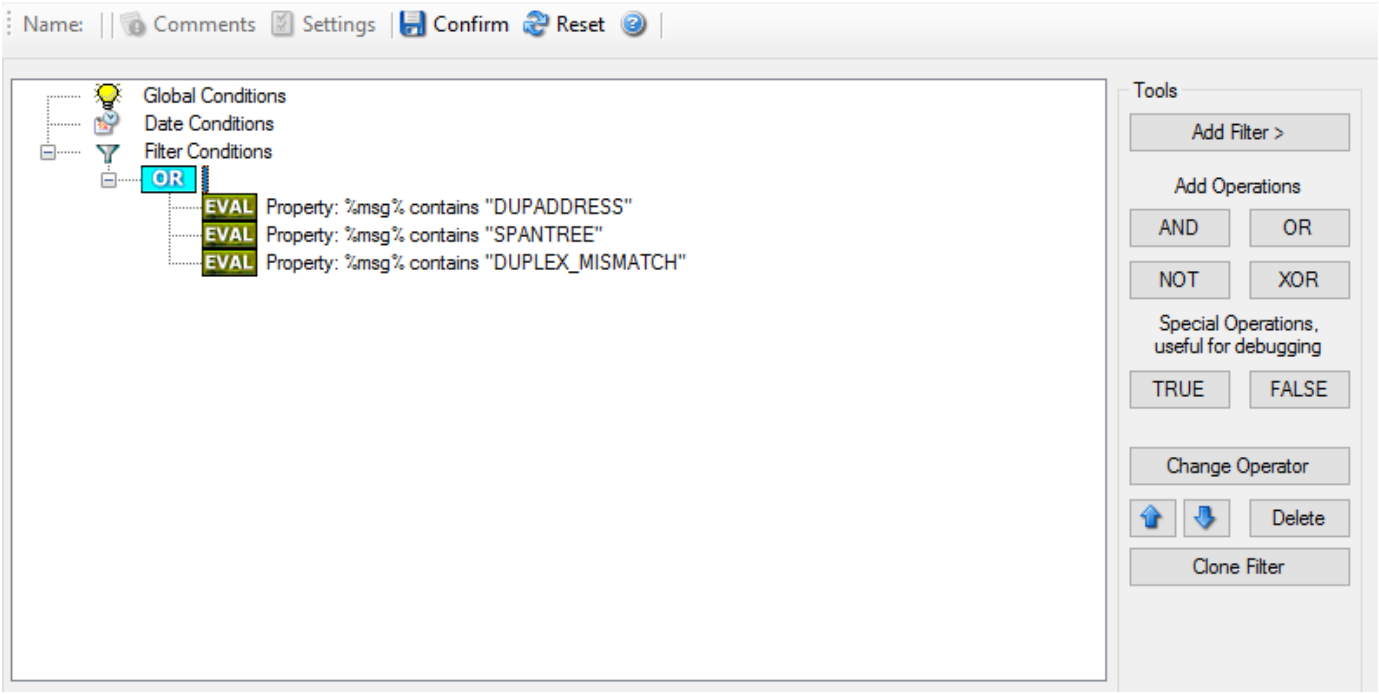


Figure 1 - Example 1

Example 2

Example 2 is very similar to example 1. Again, the message content is to be checked for three string. This time, all of these strings must be present in order for the boolean tree to evaluate to false.

The pseudo code would be as follows (under the same conditions outlined in example 1 above):

```
If (msg  = "DUPADDRES") AND (msg = "SPANTREE") AND (msg = "DUPLEX_MISMATCH) then
   execute action(s)
end if
```

In the filter dialog, this pseudo code looks as follows:

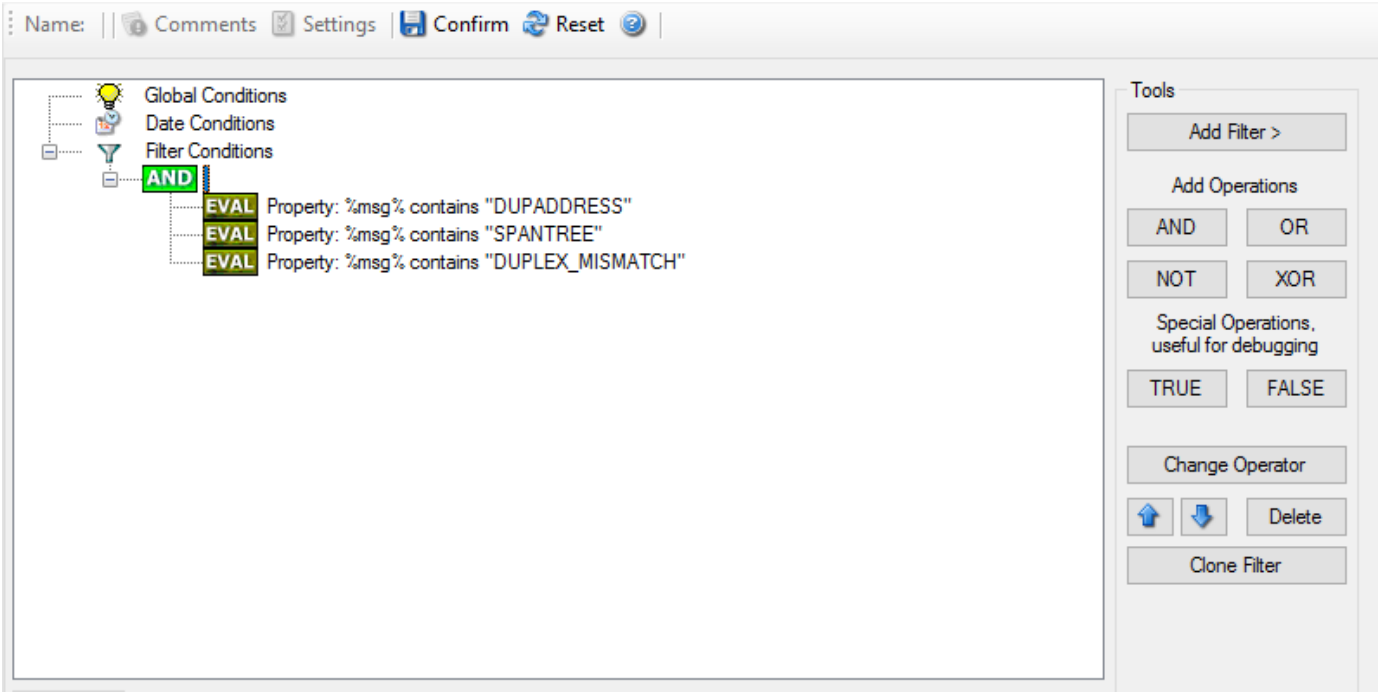


Figure 2 - Example 2

Example 3

This example is a bit more complex version of example 1. Again, the same message text filtering is done, that is if any one of the provided substrings is present, the filter eventually evaluates to true. To do so, the source system must also contain the string “192.0.2”, which can be used to filter on a device from a specific subnet.

An example like this can be used for a rule where the administrator of a specific subnet should be emailed when one of the strings indicate a specific event.

The pseudo code would be as follows (under the same conditions outlined in example 1 above):

```
If ((sourceSys = "192.0.2") And
      ((msg  = "DUPADDRES") OR (msg = "SPANTREE")
         OR (msg = "DUPLEX_MISMATCH))) then
   execute action(s)
end if
```

In the filter dialog, this pseudo code looks as follows:

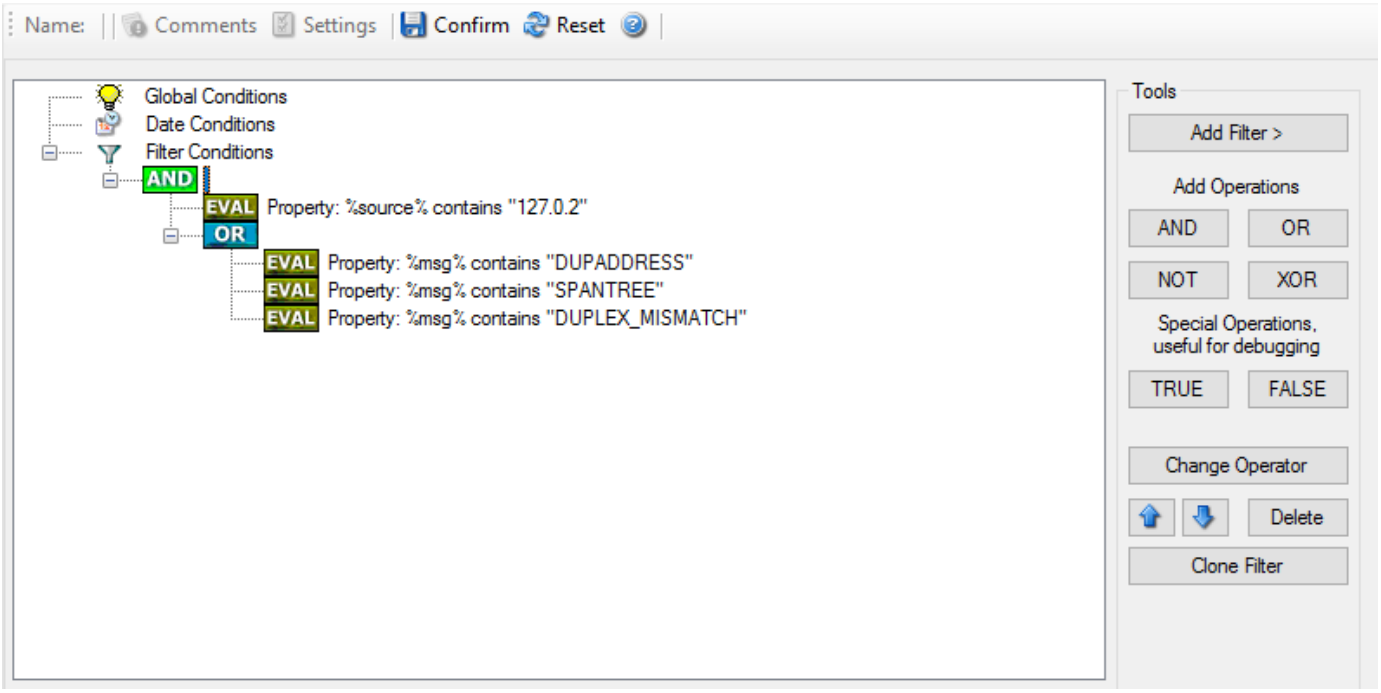


Figure 3 - Example 3

As a side note, you may want to use a range check instead of a simple include for the source system. With a range string check, you can specify that the string must be within a specified column range, in this case obviously at the beginning of the source system IP address.

Real-World Examples

To see some real-world examples of where boolean conditions inside filtering are used, please visit these web links:

- Detecting Password Attacks under Windows

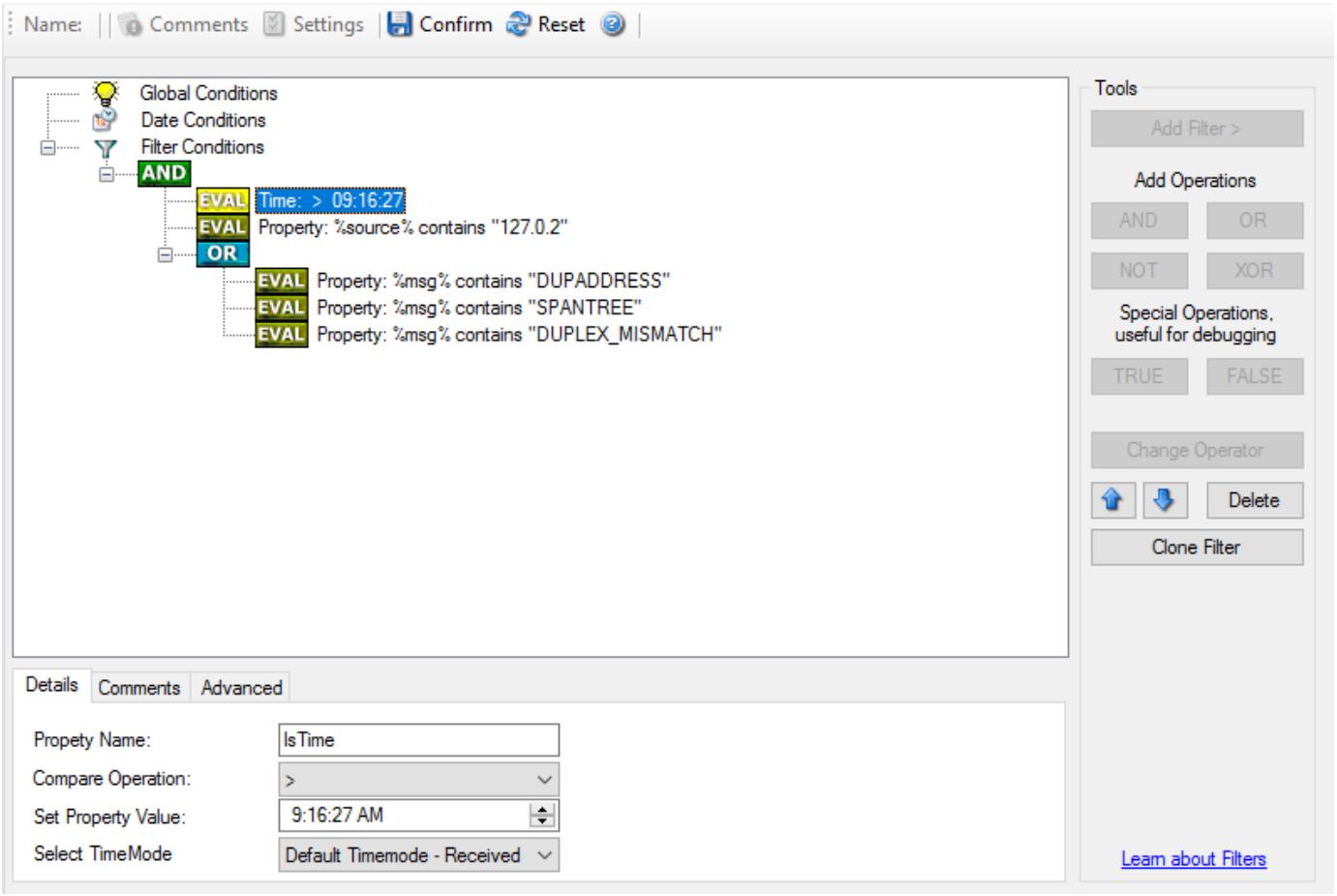
Example 4

In this example, the report is to be filtered in such a way that it shows information only in the case, if the time is greater then certain time with certain event source and one of two event ID's.

In pseudo-code, the filter could be written like this:

```
If (DeviceReportedTime is greater than {9:16:27} AND EventSource is equal to {Print} AND [EventID is equal to {10} OR EventID is equal to {18}]]
```

In the filter dialog, this pseudo code looks as follows:



EventReporter Shortcut Keys

Use shortcut keys as an alternative to the mouse when working in EventReporter Client. Keyboard shortcuts may also make it easier for you to interact with EventReporter. All these shortkeys are usually available in textboxes only. Listed below are the available short keys:

CTRL+S = Save

CTRL+X = Cut

CTRL+C = Copy

CTRL+V = Paste

CTRL+Z = Undo

Note: This is in synchronization with most major Windows applications.

Command Line Switches

There are several command line switches available for using the agent via the command line. To use the agent via the command line you need administrative rights.

- h Shows command line help
- v Shows version information and whether or not the service is installed
- i Install service
- u Remove (uninstall) service
- i Install service with a custom servicename “CustomServiceName”
- u Uninstall a service with a custom servicename “CustomServiceName”
- r Run as console application

- r -o Run ONCE as console application

If you install the service, you can start and stop the service with the “net start” and “net stop” commands. By using the “-r” switch, you run it only on the command line. When you close the command line, the program will stop working.

The “-v” switch gives you information about the version of the service.

You can import Adiscon Config Format (cfg) configuration files via the command line as well. The syntax is quite easy. Simply execute the configuration client and append the name of the configuration file. This could look like this:

Sample for MonitorWare Agent:

mwclient.exe example.cfg

Sample for EventReporter:

CFGEvntSLog.exe example.cfg

Sample for WinSyslog:

WINSyslogClient.exe example.cfg

Sample for Rsyslog Windows Agent:

RsyslogConfigClient.exe example.cfg

or

Sample for MonitorWare Agent:

mwclient.exe “example.cfg”

Sample for EventReporter:

CFGEvntSLog.exe “example.cfg”

Sample for WinSyslog:

WINSyslogClient.exe “example.cfg”

Sample for Rsyslog Windows Agent:

RsyslogConfigClient.exe “example.cfg”

After this is executed, you will see the splash screen of the configuration client and then the import dialogue, which you have to confirm manually.

For doing a silent import, the “/f” (without the quotes) parameter has to be appended. This will look like this:

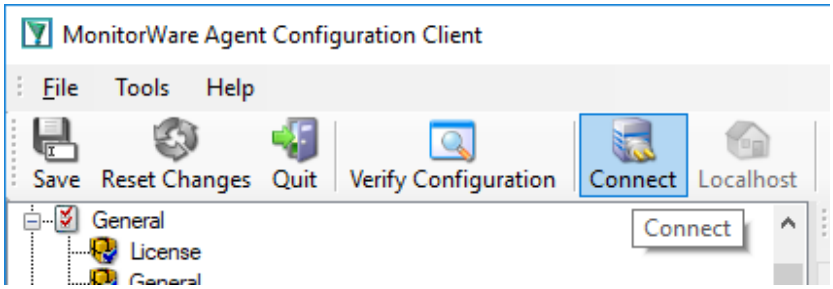
```
mwclient.exe "example.cfg" /f
```

In this case, the filename of the configuration has to be used with the quotes.

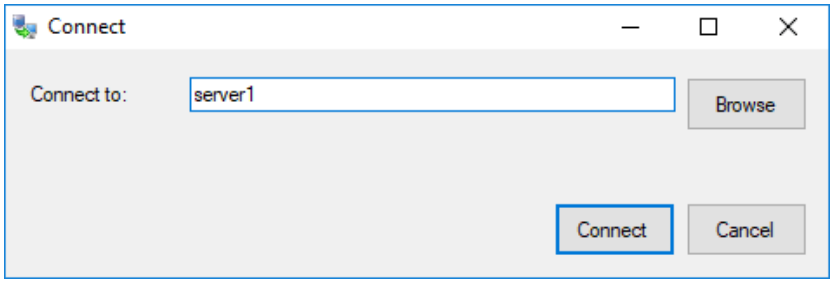
Edition Comparison

EventReporter comes in different editions. Some of them are more feature-richer than others. The manual covers description about the full feature set. In order to remove confusion we have created a Product Comparison Sheet which identifies the differences between different available versions. Click here to see which services, actions and other features are provided by each edition.

Connect to Computer



Click the Connect button in order to access another machine remotely. A window will open up.



Here you can enter the name of the machine you want to configure remotely. You can either directly enter the name into the textfield or you use the Browse button to see a list of available machines in the network. The click on the Connect button, the configuration client will verify access to the remote machine. If the verification is successful, you will be able to proceed with the remote access. Otherwise an error message will be shown.

Please Note: For remote configurations, you must ensure, that the remote machine

is accessible by network and has access rights for the current logged on local user.

System Error Codes

In most of the cases where you will get system error codes, see this list for their meanings: System Error Codes

If you cannot find them there, please contact us via the Customer Service System.

Information for a Mass Rollout

A mass rollout in this context means deploying an Adiscon client (for example MonitorWare Agent, EventReporter, or WinSyslog) to more than a handful of systems in an automated way. The aim is to invest the upfront effort needed to create a consistent “master” configuration once and then reuse it for every target machine. For guidance on differentiating between initial and update rollouts, see the MWAgent FAQ section.

Preparing the Baseline

1. Install the product on a single master system and configure it exactly as desired. Verify that the configuration works before continuing.
2. Export the configuration to a registry file via the configuration client (`Computer` → `Export Settings`).
3. Gather the files required for an engine-only installation:
   - For MonitorWare Agent 8.1 and newer this is typically `mwagent.exe` and `mwagent.pem`.
   - Older releases may also require the Visual C++ runtime and OpenSSL helper files (`Microsoft.VC90.CRT.manifest`, `libeay32.dll`, `ssleay32.dll`, `msvcm90.dll`, `msvcp90.dll`, `msvcr90.dll`).

Automated Rollout Example

Once the master system is prepared, copy the required files to a network share or removable media and automate the rollout with a script similar to the following:

```
copy \\server\share\mwagent.exe C:\some-local-dir
copy \\server\share\mwagent.pem C:\some-local-dir
cd C:\some-local-dir
mwagent -i
regedit /s \\server\share\configParams.reg
net start "AdisconMonitorWareAgent"
```

`configParams.reg` represents the registry export taken from the master system. Because the rollout ships only the engine files, this approach works well for DMZ environments where RPC or file sharing cannot be opened.

Note

`mwagent -i` (or the equivalent command-line switch for other Adiscon products) only registers the Windows service. It assumes the binaries already exist in the current directory, so copy the files before running the command.

Branch Office Rollouts

For branch offices or semi-automated deployments, distribute the prepared package and have the local administrator perform the following steps:

1. Create a directory on the target computer and copy the provided files into it.
2. Run `mwagent -i` from that directory to register the service.
3. Import the exported configuration by double-clicking the `.reg` file (or by running `regedit /s` from an elevated command prompt).
4. Start the Windows service via `net start` or the Services management console. Restarting the entire machine is not required.

Important

The directory that hosts the engine files **is** the installation directory. Deleting it removes the binaries and effectively uninstalls the product.

Updating Existing Rollouts

To upgrade an engine-only installation, update the master system first, export the revised configuration, and distribute the refreshed files using the same process. Uninstallation is unnecessary as long as you overwrite the files in place, but always stop the Windows service before copying the new binaries. For a walkthrough focused on update scenarios, refer to the MWAgent FAQ section.

Glossary of Terms

Unfortunately, in the IT world terms are not necessarily used the same way by all people.**

To clarify what we are talking about, we have created a glossary of terms as we at Adiscon understand them. Of course, we try hard to stay with the mainstream definitions.

This list includes both general IT terms as well as Adiscon-specific terms. We hope this glossary of terms will be useful for all interested parties.

Database

A database is a structural approach to data storage and retrieval. Database systems are optimized for quickly storing and retrieving data. In the MonitorWare products, databases are used to persistently store event data, enabling powerful filtering, searching, and reporting capabilities. MonitorWare supports various database systems including MySQL, Microsoft SQL Server, PostgreSQL, Oracle, and any other database with ODBC support. The database enables you to maintain a centralized repository of all your log data for compliance, forensics, and operational intelligence.

Engine Only Install

An **Engine Only Install** refers to a deployment method where only the core service executable (e.g., `winsyslg.exe`, `mwagent.exe`, `evtlog.exe`) and its essential dependencies are installed, without the full client application and user interface components. This type of installation is particularly useful for:

- **Mass deployments** where you want to minimize the installation footprint

- **Server environments** where GUI components are not needed
- **Automated deployments** where configuration is managed via registry files
- **Security-conscious environments** where you want to reduce the attack surface

In an engine-only install, the service runs with the configuration stored in the Windows Registry, which can be exported from a master installation and imported during deployment. This allows for consistent configuration across multiple systems without requiring the full installation package.

Key characteristics: - Smaller disk footprint (typically just a few MB) - No GUI components or client tools - Configuration via registry import/export - Suitable for headless/server deployments - Can be updated by simply replacing the executable files

This approach is commonly used in enterprise environments where hundreds or thousands of systems need to be monitored with consistent configuration.

EventReporter

EventReporter is Adiscon's solution to forward Windows Event Log entries from all supported Windows versions to a central system.

These central systems can be either WinSyslog , other Syslog daemons (e.g. on UNIX), or MonitorWare Agent. EventReporter is part of adiscon's monitorware line of products.

FTP

FTP stands for File Transfer Protocol. FTP is the best means for moving large files across the Internet. FTP is a client/server protocol that enables a user with an FTP client to log on to a remote machine, navigate the file system of that remote machine, and upload and download files from that machine.

There are two basic types of FTP on the Internet anonymous ftp and private ftp. With anonymous ftp, one logs in as user anonymous, giving one's email address as a password. With private FTP, one logs in with the username and password one has established on that particular system. You are logged into your home directory, with all the file permissions you would normally have there.

HTTP

HyperText Transfer Protocol, the underlying protocol used by the World Wide Web. HTTP defines how messages are formatted and transmitted, and what action Web servers and browsers should take in response to various commands. For example, when you enter a URL in your browser, this actually sends an HTTP command to the Web server directing it to fetch and transmit the requested Web page.

IETF

The IETF is an important Internet standards body. IETF is a short name for "Internet Engineering Task Force". The IETF is responsible for the creation of RFCs. Unlike other, formal standards bodies it is loosely organized. There is no specific membership to the IETF, anyone (knowledgeable) can become an IETF member just by participating on the IETF discussion mailing lists.

The IETF itself provides a good overview over itself at https://www.ietf.org/about/mission/.

IMAP

Internet Message Access Protocol. A protocol allowing a client to access and manipulate electronic mail messages on a server. It permits manipulation of remote message folders (mailboxes), in a way that is functionally equivalent to local mailboxes.

IMAP includes operations for creating, deleting, and renaming mailboxes; checking for new messages; permanently removing messages; searching; and selective fetching of message attributes, texts, and portions thereof. It does not specify a means of posting mail; this function is handled by a mail transfer protocol such as SMTP.

IPv6

Adiscon Products officially support IPv6. The IPv6 support was introduced with the following versions:

- MonitorWare Agent 8.0
- WinSyslog 11.0
- EventReporter 12.0

Support for IPv6 is available in all network related facilities of the engine. All network related actions will automatically detect IPv6 and IPv4 target addresses if configured. You can also use DNS resolution to resolve valid IPv6 addresses. Network related Services can either use IPv4 or IPv6 as internet protocol. In order to support both protocols, you will need to create two services. The only exception is the RELP Listener, which uses IPv4 and IPv6 automatically if available.

Millisecond

A millisecond is a thousand of a second. It is abbreviated as “ms”. As such, 500ms mean half a second.

Inside the adiscon’s monitorware line of products, many timers are expressed in milliseconds as a fine control over the services and actions is provided to the administrator.

Actions

Actions tell the Product (i.e. MonitorWare Agent or EventReporter or WinSyslog or any of the combinations) what to do with a given event. With actions, you can forward events to a mail recipient or Syslog server, store it in a file or database or do many other things with it. There can be multiple actions for each rule. These actions are described in the following section.

Write to File

The message is written to a plain text log file.

Write to Database

The message will be written to the specified ODBC database. This database format will be used by the MonitorWare Console that becomes available later. Therefore, if you intend to use the console, we recommend adding at least one rule that persists data to the database.

Write to EventLog

The message will be written to the application event log. Please note that the agent intentionally does not try to make the message look like it was generated on the local system. This could be very confusing. Instead, it is written inside the message part with standard values for event source and type.

Forward via Email

The message will be forwarded via email. Please note that each message will generate one email message. Messages are not combined to fit into a single mail. The Send Mail Action includes a timeout feature (m_nTimeoutValue) that provides control over message delivery timing.

Forward via Syslog

The message will be forwarded to a syslog daemon. UDP and TCP forwarding is supported.

Forward via SETP

The message will be forwarded via the custom SETP protocol. This is typically used in environments where data from different agents will be consolidated in a central place. SETP allows to transfer all InformationUnits exactly as they are. As such, the central repository can store an exact picture of the whole network.

Net Send

The message will be forwarded via the Windows “net send” functionality. Please note that the Windows function is not very reliable and requires the user to be logged in. As such, we recommend using “Net Send” only in combination with other actions.

Start Program

The message will be passed to an external process. The command line is specified in the action modifier.

Play Sound Action

This action allows you to play a sound file.

Send to Communications Port

This action allows you to send a string to an attached communication device, that is it sends a message through a Serial Port. It can send any message to a configured Serial or Printer port.

Set Status

This action allows you to create new properties of your own choice in the incoming messages. There is an internal Status List within the product which you can use for more complex filtering. You can set property over the Set Status action and you can add filter for them. They are more or less helpers for building complex rule constructions.

Set Property

With the “Set Property” action, some properties of the incoming message can be modified. This is especially useful if an administrator would like to e.g. rename two equally named devices.

Call RuleSet

This Action simply calls another RuleSet in some existing RuleSet. When this Action is encountered, the Rule Engine leaves the normal flow and go to the called RuleSet (which may contain many rules as well). It executes all the rules that have been defined in that called RuleSet. After the execution of all of them, it will return to its point from where it left the original flow.

Discard

Please see the rules description below for a complete discussion. Effectively, the message will be discarded and any further processing of this information unit be stopped as soon as a “Discard” action is found.

Post-Process Event Action

The post process action allows you to re-parse a message after it has been processed e.g. Tab Delimited format. Such re-parsing is useful if you either have a non-standard syslog format or if you would like to extract specific properties from the message.

Filter Conditions

Filter conditions are used inside the rule engine. They help to decide when a rule is to be carried out. Filter conditions are considered to match of the outcome if the configured comparison operation is “TRUE”. Available filter conditions are listed down below:

- Global Conditions
- General Conditions
- Date / Time
- InformationUnit Type
- Syslog

- Event Log Monitor
- NT Service Monitor
- Disk Space Monitor
- SNMP Traps
- SerialPort Monitor
- Custom Property

Global Conditions

Global Conditions apply to the rule as whole. They are automatically combined with a logical “AND” with the conditions in the filter tree. These are:

- Treat not found Filters as TRUE*

If a property queried in a filter condition is not present in the event, the respective condition normally returns “FALSE”. However, there might be situations where you would prefer if the rule engine would evaluate this to “TRUE” instead. With this option, you can select the intended behavior. If you check it, conditions with properties not found in the event evaluates to “TRUE.

- Fire only if Event occurs* - This is kind of the opposite of the “Minimum Wait

Time”. Here, multiple events must come in before a rule fires. For example Ping is not a very reliable protocol, so a single ping might be lost. Thus, it may not be the best idea to restart some processes just because a single ping failed. It would be much better to wait for repetitive pings to fail before doing so.

Exactly this the “Fire only if Event occurs” filter condition is made for. It waits until a configured amount of the same events occurs within a period. Only if the count is reached, the filter condition matches and the rule can fire.

- Minimum Wait Time* - This filter condition can be used to prevent rules from

firing to often. For example, a rule might be created to check the status of a port probe event. The port probe probes an smtp server. If the event is fired and the rule detects it, it will spawn a process that tries to restart the service. This process will take some time. Maybe the SMTP gateway need some more time to fully start up so that the port probe might fail again while the problem is already taken care of. The port probe as such will generate an additional event. Setting a minimum wait time will prevent this second port probe event to fire again if it is – let us say – within 5 minutes from the original one. In this case, the minimum wait time is not yet reached and as such, the rule will not match. If, however, the same event is generated 5 hours later (with the mail gateway failing again), the rule will once again fire and corrective action taken.

Date Conditions

Rule processing can be bound to a specific or the installation date. By default a Rule will always be processed.

General Filter Conditions

This set includes filters which are related to Non-Event Log specific settings. These are:

- Source System* - This is the system a message is originated from. It can be

used to check for authorized systems to pass messages to the MonitorWare Agent.

- Message Content* - The message content filter condition is very powerful. It

evaluates to true if the specified content is found anywhere in the message. As there is implicit wildcarding, there is no need to specify extra wildcards.

- CustomerID* - CustomerID is provided for customer ease. For example if someone

monitors his customer’s server, he can store different CustomerIDs in each agent. This is user configurable.

- SystemID* - SystemID is of type integer and is to be used by our customer. In

addition, it is user configurable.

- Status Name and Value* - These filter type corresponds to set status action.

Date / Time

This filter condition is used to check the time frame and / or day of week in which an event occurred.

- Time* - This filter condition is used to check the period in which an event

occurred. For example, a syslog message from a Cisco router saying that it dialed up is normal if it occurs during office hours. If it occurs at night, so, it is an alerting signal and an administrator might receive notification of this event (while he might otherwise decide to discard it). This can be done with the time setting.

- Weekdays* - This is closely equivalent to the time filter condition, except that

it is applied on a per-day basis. So it can be used to detect for example events occurring on weekends and act differently on them.

Information Unit Type

This is based on the type of service that generated the information unit. So with this setting rules can be created that act only on e.g. syslog messages or NT event reports.

Syslog

Syslog related filters are grouped here:

- Syslog Facility* - For syslog information units, this is the actual syslog

facility. If that filter condition is used on non-syslog originated information units, it will be a value mapped on a best effort basis to a syslog facility.

- Syslog Priority* - For syslog information units, this is the actual syslog

priority. If that filter condition is used on non-syslog originated information units, it will be a value mapped on a best effort basis to a syslog priority.

- Syslog Tag* - The syslog tag value, is a short string. This is provided for

non-syslog messages based on configuration. In most cases, this is used for filtering.

Event Log Monitor

Event Log Monitor related filters are grouped here:

- Event ID* - For Event Log Monitor information units, this is the actual NT event

log ID. For others, this value is undefined. We recommend using it with Event Log Monitor information units only.

- Event Type* - For Event Log Monitor information units, this is the actual NT

event log. If enabled, the event must have the configured event type or the rule will not match. This filter condition should only be used with event log information units only.

- Event Source* - For Event Log Monitor information units, this is the actual NT

event log source. For others, this value is undefined. We recommend using it with Event Log Monitor information units only.

- Event Severity* - For Event Log Monitor information units, this is the actual NT

event log severity. For others, the value is mapped on a best effort basis or not available. We recommend using it with Event Log Monitor information units only.

- Event Category* - For Event Log Monitor information units, this is the actual NT

event log category. If enabled, the event must have the configured event category or the rule will not match. This filter condition should only be used with event log information units.

- Event Categoryname* - This value contains the Category value as string if it can

be resolved. Otherwise it will contain the category number.

- Event User* - For Event Log Monitor information units, this is the actual NT

event log user. If enabled, the event must have the configured event user or the rule will not match. This filter condition should only be used with event log information units.

NT Service Monitor

The NT Service Name is used to check if vital operating services are running continuously. By default these services set to “automatic” startup. If the value returned is not true then corrective measures can be taken e.g. alerts can be generated.

DiskSpace Monitor

A flexible dialog allows to generate filters on disk free space – both with an absolute or relative value. Multiple comparisons can be done.

SNMP Traps

Using SNMP Traps MonitorWare Agent can be used to manage and monitor all sorts of equipment including computers, routers, wiring hubs etc. A trap is generated when the device feels it should do so and it contains the information that the device feels should be transmitted. Related filters are grouped here:

- Community* - It corresponds to the respective SNMP entity.
- Enterprise* - It corresponds to the respective SNMP entity.
- Generic name* - It corresponds to the respective SNMP entity.
- Version* - It corresponds to the respective SNMP entity.
- Uptime* - It corresponds to the respective SNMP entity.

Serial Port Monitor

The serial port monitor allows you to monitor devices attached to local communications ports.

Custom Property

As the name suggests it is a “Custom Property”. Internally in MonitorWare Agent all values are stored in properties. For example the main message is stored in a property called “msg”. By using this dialog you can access properties which are dynamic (Like those from SNMP Trap Monitor when using v2 protocol).

Information Units

Information units contain the data gathered by the services. As soon as a service detects a reportable event, it creates a new information unit. The information unit contains a textual representation of the event (for example a syslog message) as well as information about the event itself. For example, it contains the system that the event was originated from and the date and time it was received.

Which data is contained in the information unit depends on its type. However, there are a number of common data elements present in all information units. Most of these elements can be used as filter conditions in the rule engine. Information unit specific data elements are not eligible as filter conditions. However, there are data elements (properties) which are defined to be present in all information units even though they seem to be specific to a service type. One example is syslog priority. These values are present in each information unit type simply because priority is a good abstraction for other types, too. Such generally available properties are mapped if they are not directly supported by the service type. In the example, an Event Log Monitor maps the event log severity to the syslog priority.

There is a direct one-to-one relation between service type and information unit type. Each service type has its own information unit type.

Inside the rule base, the information unit type itself can be used as a filter condition. This facilitates creating rules that check information unit type specific properties only if they originated from the specific service type (e.g. check syslog priority only if the information unit was generated by a Syslog server).

MonitorWare Agent - Services

Services inside the MonitorWare Agent gather the data that is processed by rules. Each service type reflects a specific set of code inside the MonitorWare Agent. For example, a Syslog Service represents an instance of a Syslog server and an NT Event Log Monitor Service represents an instance of an NT Log Monitor (periodically pulling out log information).

Typically, there can be multiple instances of the same service running, as long as their configuration parameters do not conflict. For example the syslog service: there can be multiple syslog servers on a given system as long as they listen to different ports. Consequently, there can be multiple instances of the syslog service be created. For example, there could be three of them: two listen to the default port of 514, but one with TCP and one with UDP, and a third one listens to UDP port 10514. All three coexist and run at the same time.

The following services are supported:

Syslog server

Implements a Syslog server. It can be set to listen to any valid port. UDP and TCP communication is supported.

Passive Syslog Listener

The Passive Syslog Listener Service is a TCP based Listener Service that sends messages from a Syslog Queue to any remote host, that connects to it. Connections can be secured with TLS including certificate based authentication. Additionally, a greeting and response message can be configured as well.

RELP Listener

Apart from the fact that a different communication protocol is used, the RELP listener corresponds functionally to the syslog listener. The RELP listener automatically monitors all available IP addresses, including IPv4 and IPv6. This is due to the Librelp implementation method.

SETP server

Implements an SETP Server. It is used for reliable receiving event notifications.

Event Log Monitor

Monitors Windows event logs. As soon as new events are detected, these are forwarded to MonitorWare processing. This service is similar to the Adiscon EventReporter functionality.

Database Monitor

The Database Monitor read a table from an ODBC data source and generates InfoUnits out of it. These InfoUnits have properties (names by the table fields) which are filled dynamically depending on which field your table has. Each property can be used like other properties with in the MonitorWare Agent.

In short it is used to Monitor Database tables. It periodically checks a database table for new records and if it finds them, generates an event from each record. A table that shall be monitored by the Database Monitor must have an integer ID field that auto-increments.

SerialPort Monitor

The SerialPort Monitor allows you to monitor devices attached to local communications ports. Actually, this is not limited to serial (RS232) devices - devices connected via e.g. LPT ports can also be monitored as long as the device provides a proper interface to the port device.

For example - uses for the serial port monitor may be interfacing to data loggers, “strange” log sources (e.g. PBX call logs) or out-of-band log retrieval (e.g. setting a router to log to the serial port instead to the network and then picking the data from that serial line). Out-of-band log retrieval can also be used to hide the fact that logging is actually taking place.

SNMP Trap Receiver

SNMP Trap Receiver allows you to receive SNMP messages. A rough description of a Trap is that it is somewhat like a Syslog message, just over another protocol (SNMP). A trap is generated when the device feels it should do so and it contains the information that the device feels should be transmitted. It also contains some (few) standard items, as the version, community etc.

File monitor

Monitors text files. As soon as new lines at the end of the file are detected, these are forwarded to MonitorWare Agent for processing. They can be forwarded either one line at a time or in fixed chunks as set by the administrator.

Heartbeat

This service generates a special information type. Its primary purpose is to notify an upstream system that the MonitorWare Agent set for heart beating is still alive. So the upstream system can be configured to raise alarms (or corrective action) if it does not receive heartbeats from the downstream system.

Ping Probe

The ping probe pings a configured (remote) system on a schedule. If no ping response (echo reply) is received within a configured interval, an event is generated. This way, MonitorWare can check if a remote system is responding, at least at the IP stack level.

Port Probe

This is similar to the ping probe, but works at the application level. It can be used with any TCP based service. Basically, the MonitorWare Agent goes out and periodically tries to connect to a specific TCP port on a specific (remote) machine. If the connection request fails, an event is generated. As such, failing services (like database or mail servers) can be detected.

Optionally, the port probe can send a single greeting string if the connection was established and check if a response is sent by the remote system. For example, a SMTP mail gateway can be probed by connecting to port 25 and then sending a “HELO” sequence. The system should respond with a “HELO” message. Many protocols have such command sequences. Thus, they can be very effectively probed. Again, if the system does not provide the expected response, the port probe will generate a notification event.

NT Service Monitor

The NT Service Monitor checks if vital system services and applications are running and generates an alert if not.

Disk Space Monitor

Disk Space Monitor continuously checks all hard drives for available and used space. It can be used to generate long term reports as well as alerts or corrective action on low space conditions.

Associated rulesets

Each instance of a service has an associated ruleset. This allows easy creation of customized rulesets on a per service basis. Of course, all services can also operate on a common ruleset.

All services are executed as multiple threads inside the MonitorWare Agent. From the operating point of view, there is only one system service called the “MonitorWare Agent”. If the service configuration of the MonitorWare Agent is modified, the MonitorWare system service needs to be restarted in order to activate the new configuration. Later releases will have some options to automate this task.

Monitor Ware Line of Products

Adiscon’s MonitorWare line of products includes monitoring and operations management tools. It consists of several components, each of which can be used either individually or as a complete solution. As of this writing, the following products are available:

- EventReporter
- MonitorWare Agent
- WinSyslog
- Rsyslog Windows Agent

There is also an open source syslog library available for programmers wishing to integrate syslog into their C/C++ programs:

- Liblogging

New products are continuously being added - please be sure to check https://www.adiscon.com/products/ from time to time for update

NNTP

NNTP stands for Network News Transport Protocol. This protocol is used by client and server software to carry USENET postings back and forth over a TCP/IP network.

When you are using any of the common software like modern web browsers or newsreaders, you are taking benefit of the NNTP connection to participate in newsgroups.

POP3

POP3 (Post Office Protocol 3) is the most recent version of a standard protocol for receiving e-mail. POP3 is a client/server protocol in which e-mail is received and held for you by your Internet server. Periodically, you (or your client e-mail receiver) check your mail-box on the server and download any mail. It is also built into modern web browsers and email clients.

Registry File

A **Registry File** (with .reg extension) is a text file that contains a snapshot of Windows Registry entries. In the context of Adiscon products (WinSyslog, EventReporter, MonitorWare Agent), registry files are used to export and import complete product configurations.

Registry files enable: - **Configuration backup** - Save your entire product configuration - **Mass deployment** - Apply the same configuration to multiple systems - **Configuration sharing** - Share configurations between team members - **Disaster recovery** - Quickly restore configurations after system failures

The registry file can be created through the product's client interface using the "Export Settings to Registry File" option, and imported silently using: `regedit.exe /s configuration.reg`

This makes registry files an essential tool for enterprise deployments and configuration management.

RELP

RELP is the "Reliable Event Logging Protocol". It assures that no message is lost in transit, not even when connections breaks and a peer becomes unavailable. The current version of the RELP protocol has a minimal window of opportunity for message duplication after a session has been broken due to network problems. In this case, a few messages may be duplicated (a problem that also exists with plain tcp syslog).

RELP addresses many shortcomings of the traditional plain tcp syslog protocol. For some insight into that, please have a look at https://rainer.gerhards.net/2008/04/on-unreliability-of-plain-tcp-syslog.html. Please note that RELP is currently a proprietary protocol. So the number of interoperable implementations is limited.

Note that for reliable operation where messages should be preserved over a service shutdown, queue cache mode must be activated.

Repository

In the context of the MonitorWare product line, a repository typically means a database. You may also find this term used with products that use file based logging. Repository has some relation to a data warehouse or data mart as known by data mining professionals.

Resource ID

The Resource ID is an identifier used by the adiscon's monitorware line of products. It is a simple, administrator assigned string value. It can be used to correlate different events - even from different source - to a specific resource.

For example, on a Windows server running Microsoft Exchange, all Exchange events could be assigned to a resource id of "Exchange Server".

In MonitorWare Agent and WinSyslog support for Resource IDs is limited. The field is present and can be persisted to the database or stored in XML files, but besides this there is no value in it.

RFC 3164

RFC 3164 is a IETF document. It describes how syslog messages have been seen in traditional implementations. RFC 3164 is not a standard but rather a descriptive ("informational" in IETF terms) document. It does not demand a specific behavior but rather documents what has been seen. Some existing implementations of real-world syslog use different formats.

RFC 3164 is just the first step towards a newer and better syslog standard. A standard already produced by this working group is rfc 3195, which describes how syslog can be sent reliably over a tcp connection.

Adiscon supports RFC 3164 messages. There are a number of switches in each product to take care of those implementation that do it slightly different.

The formal specification for RFC 3164 can be found in the IETF RFC repository.

RFC 3195

RFC 3195 is an IETF standard. It specifies how syslog messages can reliably be transmitted via a tcp connection. RFC 3195 optionally allows for message encryption and authentication of sender and receiver. However, it has not receive any importance in practice. Servers are hard to find.

adiscon's monitorware line of products implement the core RFC 3195 protocol (actually, Adiscon was the first one to do this on the Windows platform). Under UNIX Rsyslog and SDSC syslog are known to support RFC 3195. Our liblogging project enables your own applications to "talk" 3195.

The formal specification for RFC 3195 can be found in the IETF RFC repository .

During its creation, RFC 3195 was known as "syslog-reliable". Many people still use this name to refer to it.

RFC 5424

RFC 5424 is a IETF document.

This document describes the syslog protocol, which is used to convey event notification messages. This protocol utilizes a layered architecture, which allows the use of any number of transport protocols for transmission of syslog messages. It also provides a message format that allows vendor-specific extensions to be provided in a structured way.

A standard already produced by this working group is rfc 3195, which describes how syslog can be sent reliably over a tcp connection.

Adiscon supports RFC 5424 messages. There are a number of switches in each product to take care of those implementation that do it slightly different.

The formal specification for RFC 5424 can be found in the IETF RFC repository.

The Rule Engine

Overview

This paper explains you the Rule Engine that is employed in some of the MonitorWare Line of Products namely MonitorWare Agent, WinSyslog, and EventReporter 6.0 (and higher)

What is the Rule Engine

Rule Engine is actually an engine present in the above mentioned MonitorWare Line of Products using which you can define certain filters and the actions that are to be carried out if the defined filter condition matches with the real time condition.

Rule Engine revolves around four basic concepts:

- Information Unit
- Information Services
- RuleSets
- Queue Manager

In order to understand the complete Rule Engine, you need to understand the above mentioned four concepts. The details of these are written below

1. Information Unit (Info Unit)

“Information Unit” or “Info Unit”, as we call it, is the basic building block of Rule Engine. Info Unit is basically an object that contains all the information about a specific event which includes:

- Message
- Which application generated this event
- When this event was generated
- Syslog Facility
- Syslog Priority
- Info Unit Type (it tells which Info Service has generated this Info Unit)
- etc

The following figure will give you an idea about an Info Unit:

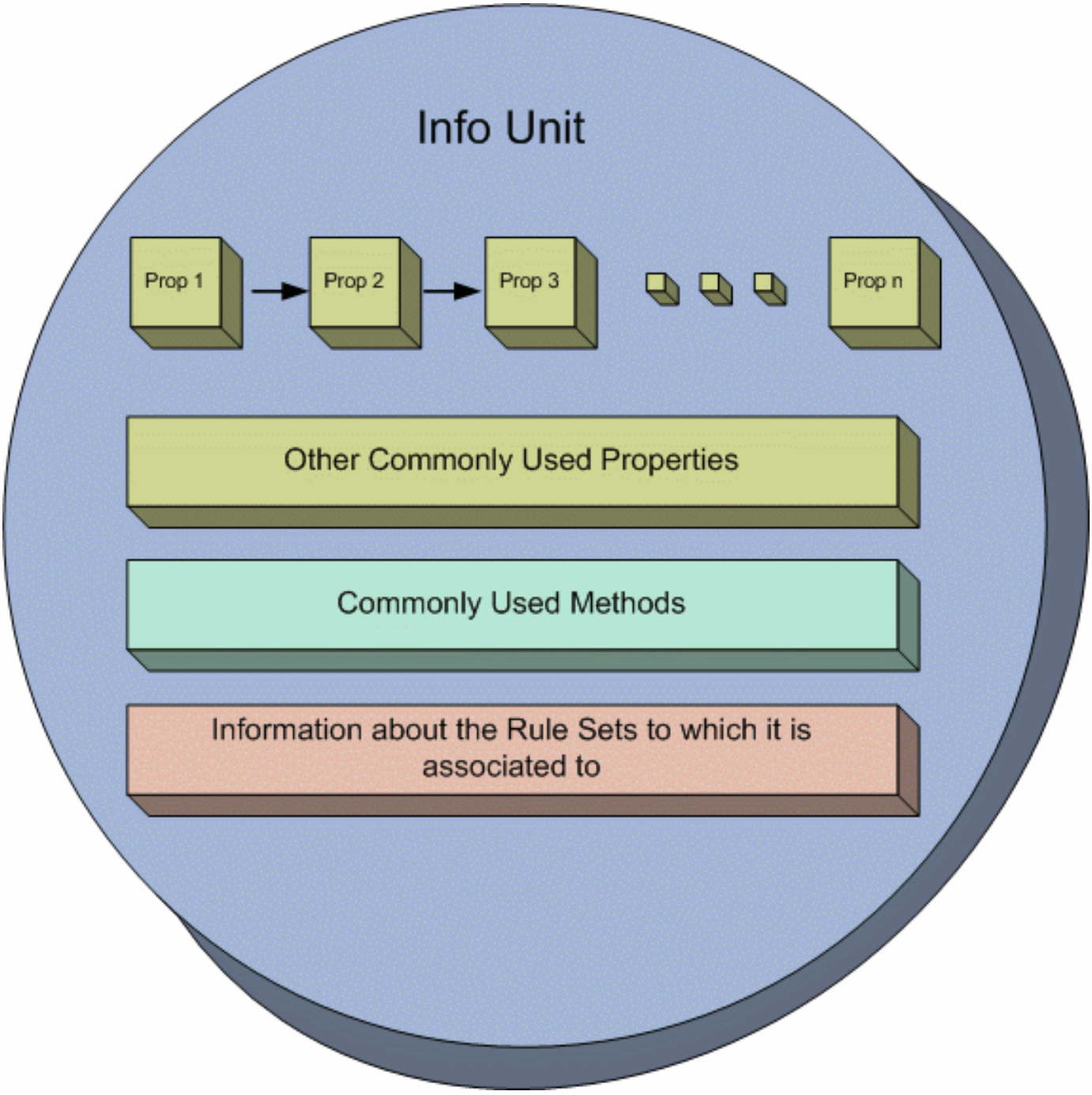


Figure 1: Conceptual Diagram of an Info Unit

As the figure illustrates, an Info Unit contains most of the properties (mentioned above in bullets) in the form of a list. In addition to this list, it also has some commonly used properties separately stored in it (for efficiency reasons). Apart from the properties, an Info Unit also has some methods which allow it to write it or to construct itself etc. Information about the RuleSets (will be explained in the coming sections) is also contained within each Info Unit so that it exactly knows which rules will be applied on it.

2. Information Services (Info Service)

“Information Services” or “Info Services”, as we call them, generate Info Units. Each Information Service will generate its own Info Units. The important thing to note over here is that each Info Unit has the same format but can have different properties and rulesets associated with it. For example, if an architect makes a building plan then it becomes a template. Now he can use this template to construct as many buildings as he likes but each one can have different properties (they can differ in color scheme, window styles etc). Exactly in the similar way, an Info Unit is actually a template from which each Info Service makes a specific object of Info Unit that might differ in properties from another Info Unit object.

Examples of Info Services

There can be a number of different examples on Info Services. Following are some of the examples:

1. Syslog server

It receives the messages that are forwarded to it and for each message (or event) it generates an Info Unit out of it.

2. Event Log Monitor

It picks up the events from the Window’s Event Log and for each event it constructs an Info Unit.

3. Ping Probe

It pings a specified device and if doesn’t find a response from the other side, it generates an Info Unit with desired information.

Important Note

One thing to note about Info Services is that there can a number of different Services running on the same machine. You can even run the different instances of the same Info Service (but with different properties naturally). In either case, each Info Service will generate its own Info Unit.

3. RuleSets

As the name suggests, a RuleSet is a set of Rules. A “Rule” consists of the following two things

- Filter Condition
- Actions

You might have noticed that the point 1 written above is singular and point 2 is plural which clearly means that you can define only one Filter condition for one rule but can define as many actions as you like. The filter condition can however contain as many Boolean operators as you like.

Filter Condition

Filter Condition is a combination of different Boolean operators which will evaluate to a Boolean answer. In simple words, the result of a filter condition can either be True or False.

Actions

Actions are all those events which are fired when a filter condition evaluates to a True value. As mentioned above, a Rule can have more than one actions associated with it which means that if a filter condition evaluates to a true value then all of the actions associated with that rule will execute. If the filter condition, on the other hand, evaluates to a false value then all of the defined actions will be skipped.

Note that other than normal actions, there are three special kinds of Actions that are worth mentioning here:

- Discard Action (Explained Later)

- Include Action (Explained Later)
- Actions that can alter the contents of Info Units permanently

4. Queue Manager

Queue Manager simply maintains a queue of all of the Info Units that have been forwarded to it by different Info Services.

Overall Picture

This section will explain you that how the different components are related to each other and how does the whole process work. The picture shown below gives an idea about how things are working. As you can see that we have four different stages through which the events are processed.

Info Services picks up the events and convert them into Info Unit. Note that each Info Service has its own Info Unit. These Info Units are passed to the Queue Manager. The job of the Queue Manager, as mentioned above and as clear from the diagram, is to simply make a queue of these Info Units that it has received from various Info Services. The Rule Engine picks up the Info Units from this Queue Manager, applies the rules on these Info Units (as mentioned above, each Info Unit has the information about which rules should be applied on it) and if necessary carries on the actions. The rule engine keeps on repeating this process while there are some Info Units present in the Queue.

Manager’s Queue

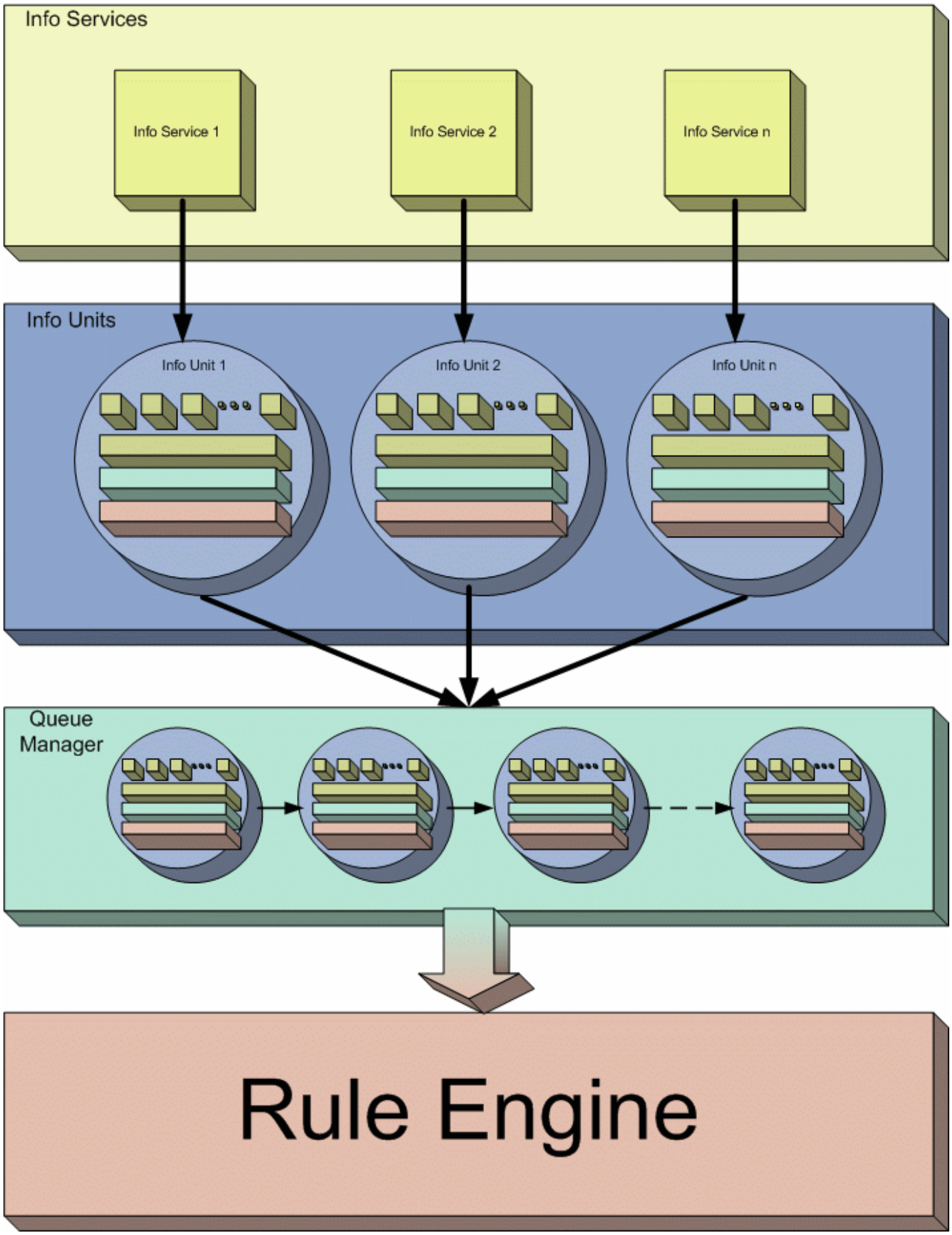


Figure 2: Overall Process

How Does the Rule Engine Work

Having explained the overall picture of the whole process, let’s specifically talk about Rule Engine. The following figure explains it in detail:

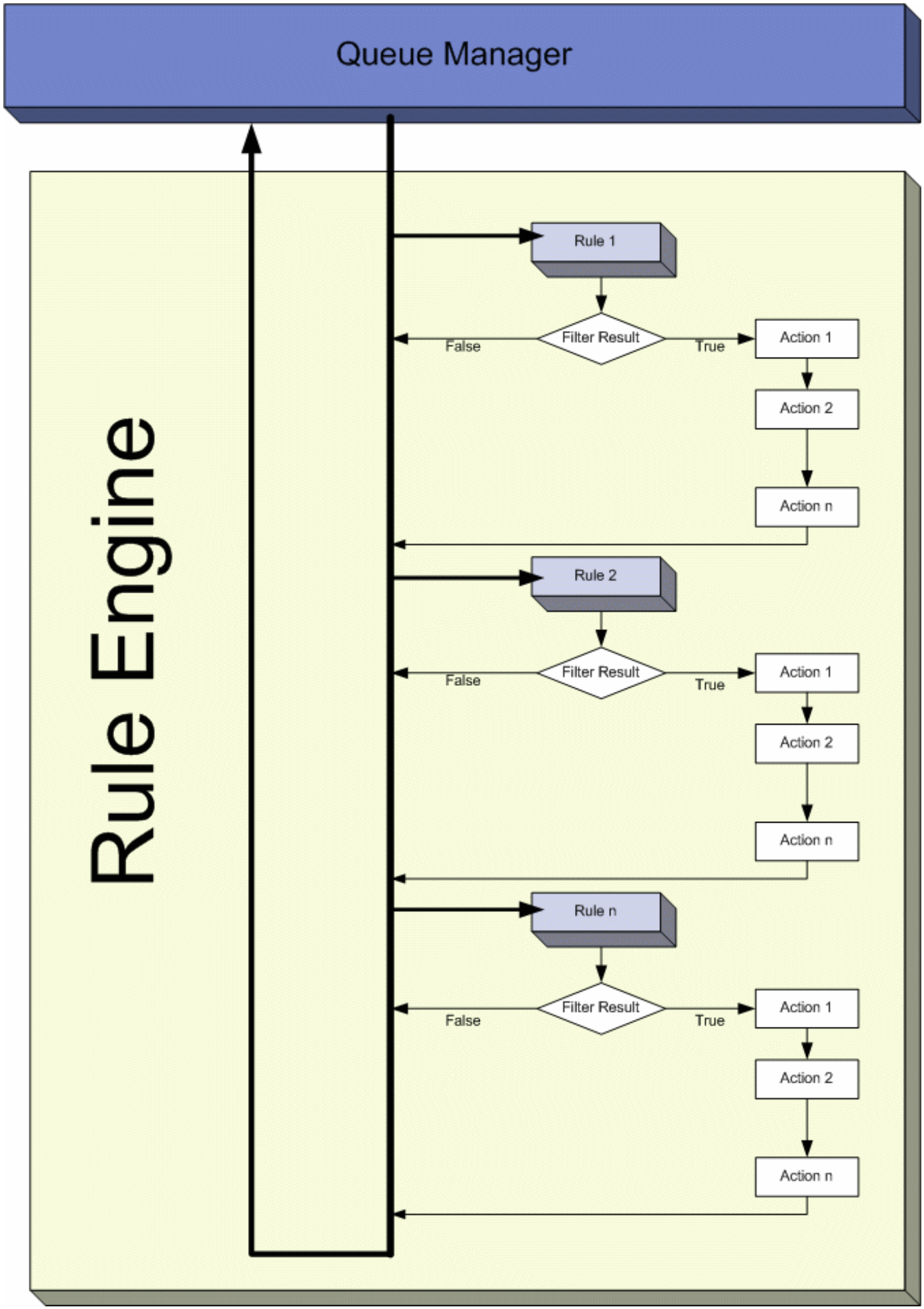

Queue Manager
Rule Engine
Rule 1
Filter Result
False
True
Action 1
Action 2
Action n
Rule 2
Filter Result
False
True
Action 1
Action 2
Action n
Rule n
Filter Result
False
True
Action 1
Action 2
Action n

Figure 3: Working of Rule Engine

As you can see in the figure, the Rule Engine picks up Info Units one by one from the Queue Manager. Since each Info Unit has the information about its Rule sets, it will apply the rules on it in the same order in which they were defined. As you can see above, it will pick up the first Rule and evaluates its Filter Condition. If that Filter Condition is evaluated to false, all the actions associated with that rule will be skipped and it will pick up the second Rule. If the Filter Condition, on the other hand, evaluates to True, it will execute all the actions that are associated with this Rule in the order in which they were defined. After the execution of all these actions, it will pick up the next rule in the current ruleset. Once all the rules have been executed, the current Info Unit (that was handed over to Rule Engine by the Queue Manager) will be destroyed and the Rule Engine will go to the Queue Manager to pick up the next Info Unit if there exists one.

The above picture has been drawn for normal flow of executions. There can be 2 conditions when the flow will not follow the diagram shown above. These conditions arise in response to 2 special kinds of actions that are called Discard Action and Include Action.

Discard Action

A Discard Action immediately destroys the current Info Unit and any action of any Rule that has been defined after the Discard Action will not be executed at all. Let’s take a simple example to clarify it further.

Let’s say that Action 2 of Rule 1 in the picture above is a Discard Action. If the Filter Result of Rule 1 is evaluated to true, then Action 1 will be executed. As Action 2 is a Discard Action, immediately the current Info Unit will be destroyed (which means that now the Rule Engine will skip all the Rules and all the actions associated with them) and the Rule Engine will go back to the Queue Manager to pick up the next Info Unit in the Queue.

Include Action

An Include Action simply includes another RuleSet in some existing RuleSet. When this Action is encountered, the Rule Engine leaves the normal flow and go to the included ruleset (which may contain many rules as well). It executes all the rules that have been defined in that included RuleSet. After the execution of all of them, it will return to its point from where it left the original flow. Let’s take an example to clarify it a little further.

Let’s say that the Action 1 or Rule 1 is an include action. If the Filter Condition result of Rule 1 evaluates to true, it will execute the Action 1. Since Action 1 is the include action in this example, it will go to the included ruleset and will execute its Filter Condition. If that filter condition evaluates to true, it will execute all of its actions and will return to Action 2 of Rule 1 (of normal flow) and if on the other hand, the filter condition of the included ruleset evaluates to false, it will skip all of its actions and will come back to the Action 2 of Rule 1 (of normal flow).

Note that there is no limit on including the rules which means that a rule that has been included in another rule may contain another rule in it which might contain another rule in it and so on.

Suggestions for Defining Complex RuleSets

While defining a complex RuleSet, it might be a good idea to follow the stages defined below.

Edit Stage # - Actions Stage 0 - Discard unwanted events Stage 1 - Post Process Stage 2 - Discard unwanted events Stage 3 - All Actions Stage 4 - Individual Actions

As mentioned above, the rules and actions will be executed in the order in which you will define them. So it’s very important that you define the actions in a way such that you achieve the desired results as well as achieve them with efficiency. For example, if you haven’t defined any filter which we call as No Filter (it always evaluates to true) and if the first action that you have defined is the Discard Action, then there is no meaning of defining any action after this first action because the first action will always be executed and it will always discard the complete Info Unit.

Here is the explanation of the above mentioned stages.

Stage 0

In this stage, you can discard those events that you are not interested in. You can use the Discard Action explained above to discard the events.

Stage 1

In this stage, we recommend to Post Process the incoming Info Units. Once the Info Unit has been handed over to the Rule Engine from the Queue Manager, you can actually change the contents of the Info Units to make them more meaningful.

Stage 2

In this stage, you might want to again discard those events that you are not interested in. Simply use the Discard Action.

Stage 3

In this stage, you will apply the actions that will apply to all of the Info Units coming (to be more specific, you will apply those rules over here for which you have selected “No Filter” as the filter condition.

Stage 4

In this stage, you will create the rules for which you have specific filter conditions.

To sum it up, we recommend doing most generic things first and least generic things later or in other words, do the generic things first and the specific things later. Note that this section suggests only the typical scenario but it can vary from depending upon the needs. For example, you might want to perform some actions on some specific events after stage 1 and before stage 2.

Rules

Rules are the workhorse of the MonitorWare Agent. All actions and processing carried out is configured by the rules defined. Rules are configured by the client and processed by the so-called “rule engine” inside the MonitorWare Agent service.

You might already know something similar to the MonitorWare Agent rule engine. Rule engines and rule bases are an extremely powerful tool and in widespread use in the industry. Examples of rule bases can be found at Checkpoint’s Firewall One Firewall Rule Base or Cisco Routing filter - just to name a few.

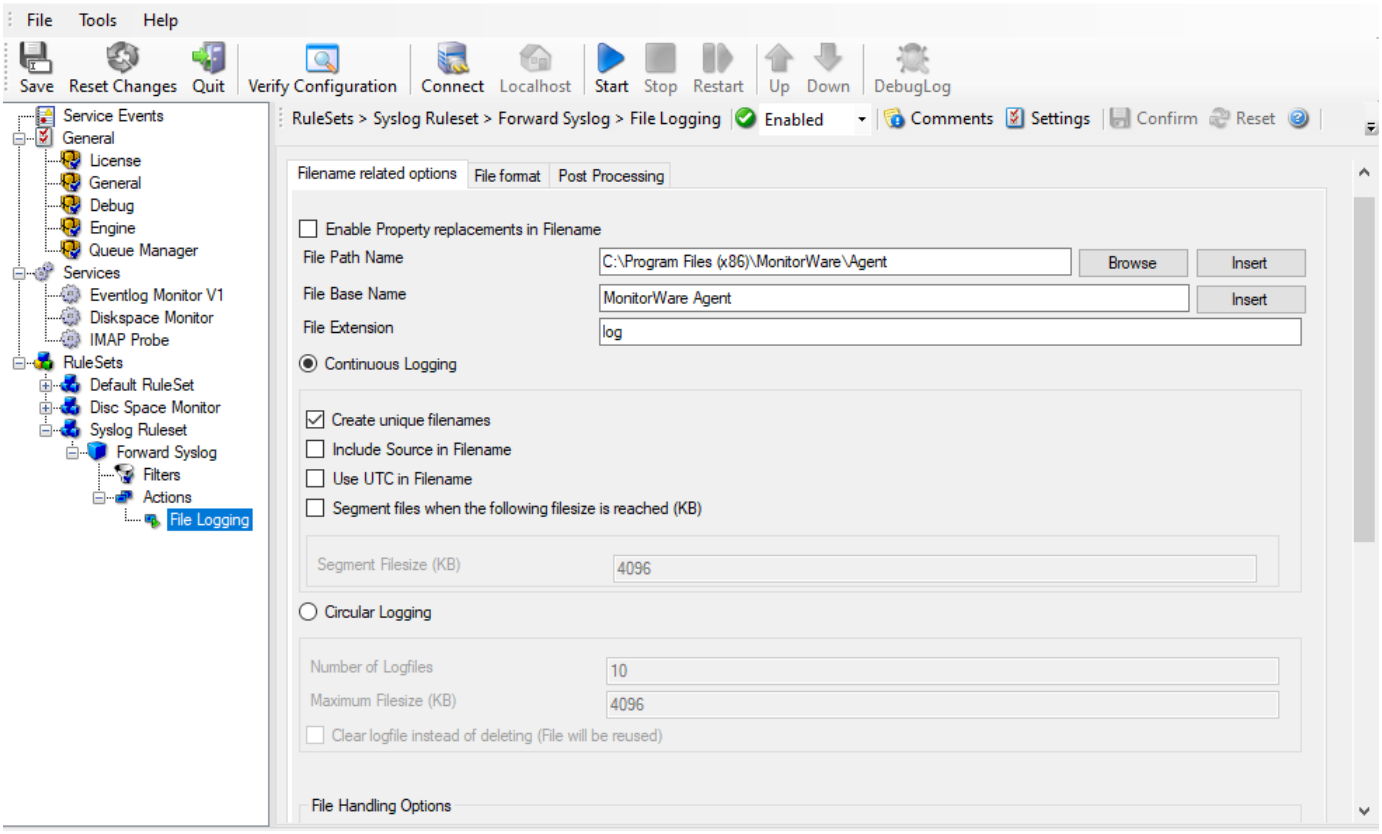
The rule base consists of the rules as configured in the client. The rule engine is the process carrying out the rules. A rule base can contain no, one or an unlimited number or rules. However, if there is no rule at all defined, no action will ever be carried out by the agent. Consequently, the client will issue a warning message in this case.

A rule has a description, associated match conditions, and actions. The match conditions are called “filter conditions”. These specify when a rule is to be carried out. Again, there can be no, one, or many filter conditions for a single rule. If there are no filter conditions, the rule will always match. This is useful in many cases. If there is more than one filter condition, all filter conditions need to match in order for the rule to match (logical AND).

Actions associated with a rule specify what to do when the associated rule matches (and only the associated rule). Actions carry out the actual processing of messages. For example, actions include logging a message to a flat file or database, sending it via email or forwarding it to syslog daemon or another MonitorWare Agent. There can be no, one, or an unlimited number of actions associated with a rule. However, if no action is associated, the rule will not have any effect. Consequently, the client will issue a warning when writing the rule base. Rules without actions can be useful to temporarily disable a rule with complex filter condition. If there are multiple actions, they are not guaranteed to be carried out in any specific order. If you definitely need an action to be carried out before another one, you currently need to define two rules.

Actions can be modified with action modifiers. These are the strings attached to a specific action. Action modifiers allow customizing a specific behavior of this action. It modifies only this action and only this one, other actions of the same type are not affected - regardless if the appear in the same rule or a very different one. The use of the action modifier depends on the type of action. For example, with syslog forwarding it is the host the syslog message is to be forwarded to. With ODBC database logging it is the DSN and so on. If there is no action modifier, the values configured in the client’s configuration tabs will be used. They are also used for all values that cannot be modified via the action modifier (e.g. the SMTP server address for email forwarding).

Below find a screenshot of a rule base with a number of rules, filter conditions and action modifiers:



Sample Rule Base

Now that we know the elements, how are rules being processed. It is easy. Rules are strictly processed from top to bottom, or from number one to the last one. For each rule the filter conditions are checked to see if they match. If they do, all associated actions are carried out. Then, the rule engine advances to the next configured rule. Once again, it checks if it matches and - if it does - carries out the actions associated with that rule. Then the next rule is

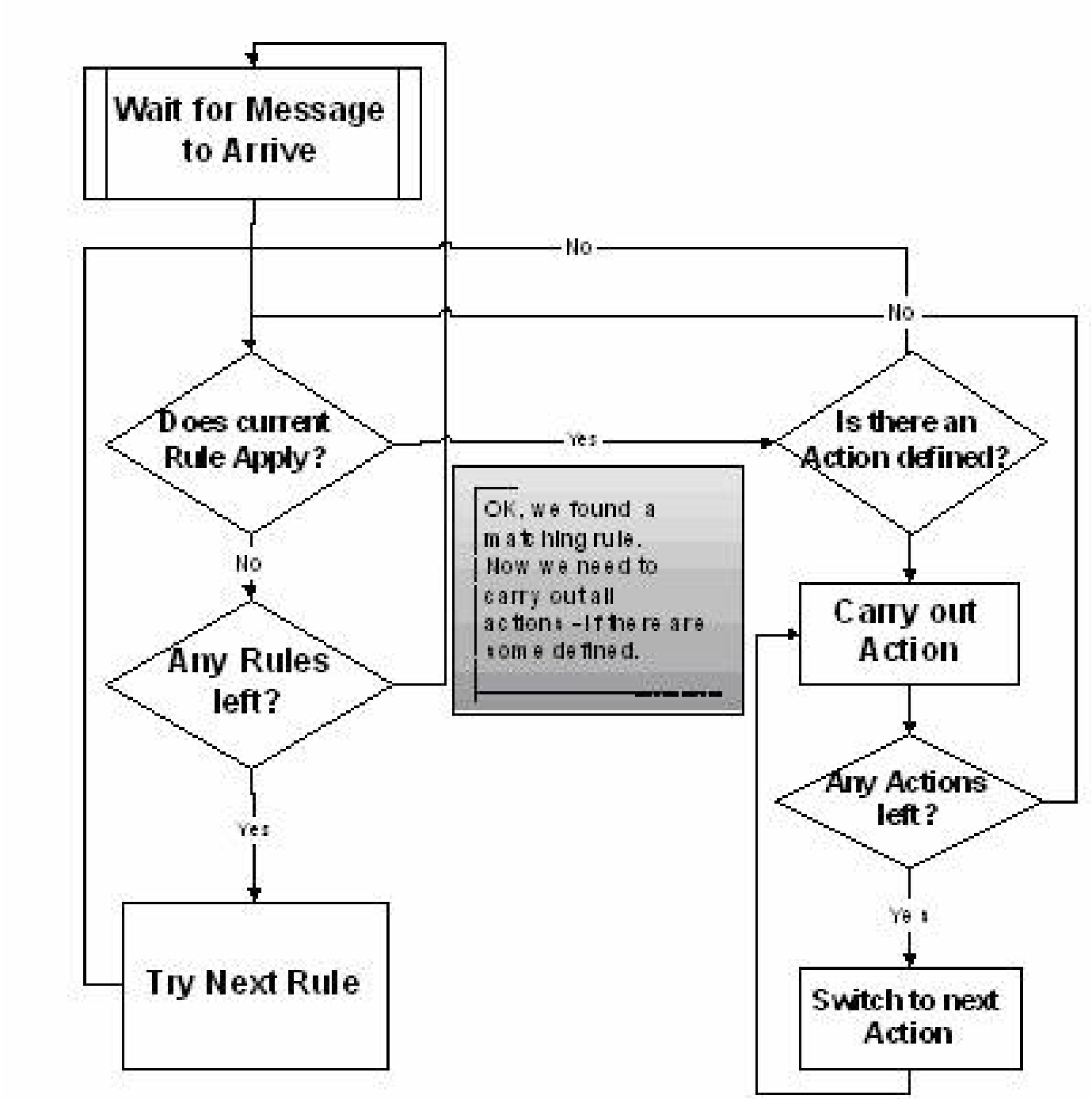
processed and so on. The rule engine stops when there are no more rules to be evaluated. It also stops if a rule contains a “discard” action.

The “discard action” is a very special and powerful action. It does not actually carry out any processing. In fact, it disables all further processing for a message as soon as it is found by the rule engine. So what is the discard action good for? It is used to handle common situations where a number of well know messages - unimportant messages - should be filtered out so that the other rules do not need to take care of these messages. In many other products using rules bases, this is called the “block rule”. Please note that with Adiscon’s rule engine, there can be multiple block rules at multiple layers of the rule base giving you additional flexibility.

One last thing to mention: the rule base is applied to every message arriving at the MonitorWare Agent. By design, there is no way to modify the behavior of the rule base for the next message to be arrived. This ensures an always consistent processing of incoming messages. However, there can be multiple rule bases. Each rule base is associated with a service. Only the rule base associated with the service generating the message will be processed.

While building and testing your rule base, please keep in mind that the MonitorWare Agent service needs to be restarted to load a modified rule base. The reason is that the service does not re-read the rule base to save system resources.

There is an online seminar available on the rule engine and its processing. If you are interested in a more in-depth view, you might want to visit it at rule engine.



Rule Engine Flowchart

For those interested in more in-depth information on how the rule engine works, this flowchart might be helpful:

There is an online seminar available on the rule engine and its processing. If you are interested in a more in-depth view, you might want to visit it at rule engine.

SETP

SETP is the “Simple Event Transfer Protocol”. SETP allows reliable delivery of events between SETP supporting systems. EventReporter, WinSyslog, and MonitorWare Agent support SETP. EventReporter works as SETP Client Only. As such, it can forward events generated and gathered by them to central or intermediary SETP servers. WinSyslog EnterPrise Edition works as SETP client and server, only. The MonitorWare Agent can operate both as a SETP server and client and as such also as a relay. It plays a vital role in a complex, distributed environment.

SETP was developed for MonitorWare. It allows synchronous communication between SETP clients and servers. With SETP, an event can be forwarded exactly as it was on the original event generating system. For example, if a syslog message is received on a remote system, that exact syslog message can be forwarded via as many SETP relays as is configured. During that relaying, no information from the original message is altered or lost. As such, each of the relays as well as the final SETP server will see the original source address, time stamps and message.

Furthermore, SETP guarantees reliable delivery. It is based on TCP, so each of the SETP peers know exactly that the communication partner can successfully receive and process the message. SETP guarantees that new events are only forwarded after the previous ones were successfully received and processed. SETP also checks for on the wire errors. Due to its characteristics, SETP can successfully be used in barely or occasionally connected environments like radio connected systems.

The SETP design is influenced by many industry standard movements, most notably the BEEP protocol and XML. However, SETP is optimized to have a very lightweight footprint. As such, it can be implemented even in low powered devices with little overhead.

SMTP

The “Simple Mail Transfer Protocol”. This is an Internet standard for sending email messages. Virtually all major email systems are either based on SMTP or at least offer gateways to SMTP capable systems.

SMTP is used for sending email. It cannot be used to pick up email messages. For this purpose, protocols like POP3 or IMAP4 are required.

SMTP is highly standardized. As such, a standard email client can work with all SMTP compliant servers. In the public Internet, almost all providers offer SMTP compliant mail servers for their customer’s use.

SNMP

SNMP stands for Simple Network Management Protocol. A set of standards for communication with devices connected to a TCP/IP network, like routers, hubs and switches. A device is said to be SNMP compatible if it can be monitored and/or controlled using SNMP messages.

SNMP messages are known as PDU’s - Protocol Data Units. Devices that are SNMP compatible contain SNMP ‘agent’ software to receive, send, and act upon SNMP messages. Software for managing devices via SNMP are available for every kind of commonly used computer and are often bundled along with the device they are designed to manage. Some SNMP software is designed to handle a wide variety of devices.

Syslog

Syslog is both a protocol and a system for logging messages in IP networks. Originally developed for Unix systems, it has become the de facto standard for system logging across multiple platforms. Syslog uses UDP port 514 by default (though TCP and TLS variants exist) and follows formats defined in RFC 3164 (traditional) and RFC 5424 (structured). Messages include facility, severity, timestamp, hostname, and message content. All Adiscon products support both sending and receiving syslog messages.

Syslog Facility

Syslog Facility is one information field associated with a syslog message. It is defined by the Syslog protocol. It is meant to provide a very rough clue from what part of a system the message originated from. Traditionally, under UNIX, there are facilities like KERN (the OS kernel itself), LPD (the line printer daemon), and so on. There are also the *LOCAL_0* to *LOCAL_7* facilities, which were traditionally reserved for administrator and application use.

However, with the wide adoption of the syslog protocol, the facility field contents has become a little less clear. Most syslog enabled devices nowadays allow configuring any value as the facility. So it is basically left to distinguish different classes of syslog messages.

The facility can be very helpful to define rules that split messages for example to different log files based on the facility level.

TCP

A reliable IP transport protocol. TCP communication ensures that no packets are lost in transit. As such, it is most useful in low-bandwidth or unreliable environments. Examples are slow WANs or packet radio networks.

Here you find information about Performance Tests and Results

UDP

A non-reliable IP transport protocol. It provides best effort delivery. Typically, in LAN environments UDP packets are never lost. However, in WAN scenarios or with heavily loaded LANs, UDP packets might be lost.

Here you find information about Performance Tests and Results

Upgrade Insurance

UpgradeInsurance is Adiscon's software maintenance plan. It offers free major upgrades as well as priority support. UpgradeInsurance is available for all Adiscon products and can be purchased for a period between 1 and 5 years.

Click here for more Information about Upgrade Insurance.

UTC

UTC is the so-called "universal coordinated time". UTC was formerly referred to as "GMT" (Greenwich Mean Time) and is the basis of the international time zone system. For example, New York, USA is 5 hours behind UTC. So if it is 12 noon in New York, the UTC time is 5pm.

The MonitorWare line of products often uses UTC. UTC has the fast advantage of providing one consistent time notation, even if devices are across multiple time zones. This is extremely valuable if a central location is to consolidate events from senders in multiple time zones.

Using UTC might not be appropriate if a whole system is contained within a single time zone. As such, most time parameters inside the MonitorWare line of products can be configured to work with local time instead of UTC.

Copyrights

This documentation as well as the actual MonitorWare Agent product is copyrighted by Adiscon GmbH, Germany. To learn more about other Adiscon products, please visit https://www.adiscon.com/en/products.

We acknowledge using these following third party tools. Here are the download links:

Openssl-3.2.1: https://www.openssl.org/source/openssl-3.2.1.tar.gz **Liblogging 0.7.1:** https://github.com/Rsyslog/liblogging/archive/refs/tags/v0.7.1.tar.gz **Librelp 1.11.0:** https://github.com/Rsyslog/librelp/archive/refs/tags/v1.10.0.tar.gz **Libfastjson-0.99.8:** https://github.com/Rsyslog/libfastjson/archive/refs/tags/v0.99.8.tar.gz

Liblognorm 0.3.5 https://github.com/Rsyslog/liblognorm/archive/refs/tags/v0.3.5.tar.gz

Microsoft, Windows, and the Windows logo are trademarks, or registered trademarks of Microsoft Corporation in the United States and/or other countries.

Other mentioned trademarks are for reference only. They belong to their respective owners.

- **genindex**

Index

W