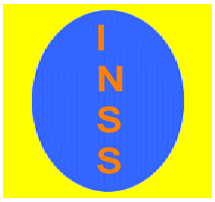

International Network Support & Service – Glas

OPENMAKAO – Nagios plugin

Table of Contents

Overview

OpenMakao is a plugin for Nagios coming from Makao, a plugin for Siebel.
OpenMakao is opensource, in opposite to Makao, which is closed source.
OpenMakao contains all what is not Siebel related from Makao. It is the light version.
Therefore it is mainly Logfile and Port check.
It has also a feature to generate necessary Nagios entries for the cfg files.
You can add also other not OpenMakao related Nagios services, and it will generate all necessary entries for the Nagios cfg files. You just have to define one ini file for each host.
The main features are Siebel related, therefore this is not a big plugin.

If you are interested in the full Makao version for monitoring Siebel, have a look at http://inss.ch.

For interest in the source code of OpenMakao, just request at:
info@inss.ch

OpenMakao is a product of International Network Support & Service – Glas

All configurations will be done within one ini file for each server.

It does not monitor points that are covered by Nagios (standard) plugins.
For complete monitoring that includes the operating system it is recommended to use the Nagios standard plugins (memory, disc space, processes and so on).
For Windows the free NSClient++ is available: (http://nsclient.org/nscp/).

You will find all plugins at the http://exchange.nagios.org homepage.

Note:
Usually monitoring checks are twofold in description. The first description in the chapter „Monitoring points“ describes in general how the check works. Most times you will have a second chapter in „Ini configuration“ how to configure the check in the ini file.

More features or enhancements can be requested: info@inss.ch.

Monitoring points

Port check

This is like a telnet connection testing if the defined port is listening.
If the connection cannot be established (port is closed), the check fails and sends a critical message to Nagios. You can monitor the Siebel Gateway and also Siebel Application server for availability with it (see following chapters).

In other words: if the port is open, it will send OK to Nagios, if the port is closed, it will send critical.
You can monitor also other applications for the port.
Results are only OK or CRITICAL for Nagios.

See chapter [portcheck] for configuration details in the ini file.

Logfiles

OpenMakao scans also the logfiles for errors.
You can configure for each file exactly the tokens for which OpenMakao shall look for.

And if e.g. a well known error occurs that doesn't need to send an alert, you can also define exceptions. You can define several exception tokens like the error tokens.
If OpenMakao finds an error, it will check the exceptions if it shall make an alert or not.
OpenMakao will remember the last line checked between the calls, and continue with next

line to scan for errors. If there were no new lines in the logfile, nothing new will be scanned.
OpenMakao stores the data for that into the data file ending with .dat.
You can define several logfiles.

See chapter [logfile] for configuration details.

Requirements

Operating System: the Plugin is based on Java and therefore OS independent.

Nagios
A Nagios server needs to be available for OpenMakao
(see http://nagios.org).
OpenMakao works as a passive plugin and needs therefore NSCA configured and running at Nagios.
For informations about NSCA look at:
http://nagios.org/download/addons
For downloading NSCA you can use (or look for a more current release, this is current at may 2010):
http://prdownloads.sourceforge.net/sourceforge/nagios/nsca-2.7.2.tar.gz

JAVA JRE
OpenMakao needs to have Java 1.5 JRE or above installed.

OpenMakao needs a folder with write permissions, it is writing itself logfiles.

Installation

Just unzip the content to a folder of your choice, e.g. C:\makao.
The folder needs to have write permission for the user who is running the plugin.
The fresh folder shall contain following files:
ini4j-0.5.2.jar
log4j-1.2.15.jar
log4j.properties
openmakao.jar
README.TXT

Only openmakao.jar, log4j.properties and log4j-1.2.15.jar are mandatory to run OpenMakao.

Run OpenMakao with the command: java -jar openmakao.jar -i
It will generate a sample ini file in the same directory.
It contains already some comments.
Detailed description will follow.

Later appearing files:
OpenMakao will also put new files into the directory.

a) The logfiles according the log4j.properties file. Default are three rollover logfiles.
b) Input files for the servermanager. These are named:
„input_<enterprise>_<applicationserver>.txt there <enterprise> is the name of the Siebel enterprise as defined in the ini file and <applicationserver> is the name of the Siebel application server as defined in the ini file.
c) It will create a dat file with same name as the related ini file just ending with .dat instead of .ini.
It stores information about how many alerts had been sent for one task and the linenumber of logfiles.
d) The send_nsca.cfg file in folder conf. If it doesn't exist, OpenMakao will create it. It will contain a single line:
encryption_method=0
Only 0 and 1 is valid for OpenMakao. You can change this file after it is created once. As long as it exists, OpenMakao will not write a new one.

You don't need another nsca client, this is already build in.

Ini configuration

You use one ini file for every Siebel application server.
You may use several inifiles for several different application server in the same folder. The input file generated for the execution is always named according the environment (format „input_enterprise_applicationserver.txt“).

The ini file is separated into several sections.
A section line follows the standard:
[sectionname]

After editing the ini file and having done the first test run, have a look into the OpenMakao logfile for WARN and ERROR messages. If not correctly edited, it will show this in the logfile.

[environment]

The first section is environment. These data are mandatory.
host = <hostname of siebel server>
The hostname like it is defined in Nagios.
nagiosserver = <hostname nagios server>
Put the hostname or IP dress of the Nagios server.
nagiosport = <nagios port>
Put in the port of the Nagios server, usually 5667.

Additional parameter:
You can add the line „debug = 1“ if you want to get debugging informations for OpenMakao. This is only useful if you have errors during the execution. With debug = 0 you can switch it off (same as skipping the line), debug = 2 or 3 increases the debug level.

[checks]

This section is optional.
If not exists or is completely empty, all checks are enabled.
If not empty, all checks not listed are disabled.

It is mainly for quickly disabling some checks.
If you want to use it, you should put all check sections, and enable/disable with true/false.
If you have one or more lines defined, all not listed will be disabled.
Only sections for Nagios checks are valid here.
Example with all check sections:
portcheck = true
logfile = true

[portcheck]

This section is optional.

You can define as many port check as you want.

For each port check you have to define two lines:
service = <Nagiosservice>
portnr = <portnumber>

Both lines are mandatory for each port check.
The host is the host defined in the environment section of the ini file (line host = <hostname>). For different hosts you will have to create additional ini files.

[logfile]

This section is optional.
You can define several files to be monitored with different words to scan for errors, and also exceptions from that.

Similar as in the tasknumber section every Nagios service needs to have four lines to be defined.
service = <name of Nagios service>
path = <path to file>
errortokens = <list of words to scan for errors, comma separated>
exceptiontokens = <list of words where to ignore the error>

For more files just add in sequence another four lines, all four lines must exist for each Nagios service.
Invalid configurations will appear in the logfile with ERROR.

The line *path* defines the path to the file to be monitored (relative or absolute).

The line *errortokens* define for what OpenMakao shall scan the file.
Separate values with comma.
Example: *errortokens = error,shutdown,fail*

The line *exceptiontokens* defines where it is an exception from an error. Separate values

with comma.
If e.g. the file contains error, but you don't want an alert for Genericerror, you put:
Example: *exceptiontokens = Genericerror*

Errortokens and exceptiontokens are not casesensitive.

OpenMakao remembers the last line where it found an error and stores the data into a dat file (see installation notes).
It will continue with the next not yet scanned error in the file.
If the file appears to be smaller then before it starts from first line assuming it is a new file.
It will alert only once per found error.

[servicenames]

This section is optional and can be deleted after it has been used.
It is not for any Nagios check. Therefore you should remove it for ini files
used for the checks.

Here you can enter any free text in each line for other Nagios services.
It will be only used for the parameter -s (or -service) to generate the Nagios services.
The parameter -s will generate a text file with all configured Nagios services in the ini file.
See parameter description for details of using -s.
The values in this section, one value without = for each line, will be added as Nagios service names to the textfile.

You don't need to fill this if you are only using OpenMakao checks. Use it only for none OpenMakao Nagios services. Therefore it will not be added to the sample ini file.

You can use it therefore also for Nagios services which are not related to OpenMakao checks. You just need to configure one inifile for each host.
Be aware that the section environment will be read for the name of the host.
For using it, even without any OpenMakao checks, you need to have three sections defined: environment (for hostname), servicenames and also the servicetemplate section.

[servicetemplate]

This section is optional and can be deleted after it has been used.
It is meant for generating the needed Nagios services automatically.
The steps are:
1) Be sure you have filled already complete ini files.
2) Adjust the example service to your needs, at least you will have to put in the value for contact_groups.
3) Run OpenMakao with the parameter -s <inifile1> <inifile2>. You can put several ini files after -s, superheated by a blanc.
4) Look into the created file generatedservices.txt if everything is correct.
5) Copy/paste the services from generatedservices.txt into your Nagios cfg file, usually services.cfg.

6) Restart Nagios to get the new services.

It looks for every section in the inis and put the name into a service, along with hostnames.

Note:
This section is not used for status checks and should be removed after the final configuration is done.

List of names for the Nagios services

Look at the parameter -s before you try to do everything manually.

```
service = in the [logfile] section.
```

For the port check you define yourself the name of the Nagios service with the parameter service = in the **[portcheck]** section of the ini file.

Installation NSCA server

See the official document NSCA_Setup.pdf (also in this package) from nagios.org for that.

Installation NSCA client

Here is not much to do because OpenMakao has the NSCA client already build in.
You will not need the send_nsca executable.
OpenMakao uses the NagiosAppender library as NSCA client.
It is included in the makao.jar file.

OpenMakao, to be precise, the NagiosAppender needs to have a config file with name send_nsca.cfg.
If it doesn't exist, OpenMakao will create the folder and file for that. It is then in the folder „conf“ with the file send_nsca.cfg.

It will contain only a single line:
encryption_method=0
That is enough to work.
If you are using a different encryption_method, you can change the value. As long as the file exists (in folder conf), OpenMakao will not create or overwrite the existing file.
Valid are only the values 0 and 1.
Other values provided by NSCA client are not supported by NagiosAppender.

OpenMakao parameter

Use the -h parameter to get a list with short comments

Usually, in Windows as well as in Linux you can execute OpenMakao like any Java jar file like e.g.:

java -jar makao.jar -h

Parameter -h

Print on the standard output the list of parameters along with a short comment.

Parameter -n <inifile>

This is the most common parameter to use OpenMakao.

It tells OpenMakao to get the parameter from the file in path of the second parameter <file>.

The file must be either from ini file format as described in configuration chapters.

Or you can take the dot dat file. The dat file will be created after the first run and stores data like the content of the ini file, logfile position, parameter values and other data into it.

Once the dat file is created, you can delete the ini file.

To use the dat file instead of the ini will slightly enhance performance because it doesn't need to analyze the inifile.

Restore the ini file with parameter „-exportini“.
The dat file will have same name and path like the ini file just ending with dat instead of ini.

The path can be relative or absolute.

OpenMakao will then first analyze the inifile to know what and how to check.

Depending on configuration it will then start the Siebel servermanager to get the data from Siebel or look into an output file from the servermanager.

It will analyze the data, convert these into Nagios standard format and send it with the build in nsca client as a passive check to Nagios.

In Windows it is best practice to create a batch file and schedule the batch in Scheduled Tasks in Windows.

Example:

java -jar makao.jar -n siebel01.ini

Parameter -i

OpenMakao will generate a sample ini file in the same directory. It will display the absolute path of the created file.

The file will have the name: „sample.ini“.

It contains also some short comments to the sections and parameters.

Rename it to for example the name of the Siebel application server.

You must have one ini file for each Siebel application server.

You might put them all together into the same directory.

OpenMakao will produce other files, containing the name of the application server to avoid using the same file for different servers.

Parameter -s <inifile1> <inifile2> <inifile3>

Once you have filled out every inifile for each Siebel application server, you will have to create the Nagios services. As this is not very amusing OpenMakao can automate that.

Prerequisite is that you have filled out all inifiles and one (any) inifile contains the adjusted [servicetemplate] section. It will take the first servicetemplate it founds in all the files.

It doesn't make sense to do that one after another because OpenMakao summarizes the result for all files (hostnames per servicenames).

Therefore you should run it once with all ini files.

OpenMakao will use the template for that what is defined in the section [servicetemplate], at least you should have defined the contact_groups. The parameter <hostname> will be replaced by OpenMakao itself and will contain the name you defined in the environment section.

Be aware that services disabled in the [checks] section will be skipped.

It will add free text Nagios services from the section [servicenames] without changes (useful for additional none OpenMakao checks). You have to add the section manually, it is not part of the sample ini file because it is not related to OpenMakao checks.

If successful, it will display a message and the name of the file where it stored the generated services. The name of the created file will be: „`generatedservices.txt“` in the same directory.

Parameter -version

This parameter prints the current version of OpenMakao to the screen.
It will display in the demoversion how long the version will be valid.
Short version is -v.

Troubleshooting

OpenMakao is a complex plugin and therefore it is essential to look for errors and warnings in the logfile to ensure that it is correctly configured.
Especially for the first runs always have a look into the OpenMakao logfile in the same directory.

OpenMakao Logfile

OpenMakao writes a logfile with log4j. The output format can be configured in the log4j.properties file. Look for the documentation of log4j library for details.
OpenMakao is doing some checks in the ini file to see if the configuration is valid.

A message will then appear in the OpenMakao logs: Ini is valid.

It marks start and end of each check, you will then see to which check an occurring error is related.

Not every error occurring in the logfile lets the complete task fail. Anyway, if you find an error in the logfile, try to resolve the issue. If not possible contact info@inss.ch.
The task for a Nagios service is completed with the line (example):
Will send for host windowsone to nagios 192.168.130.128 at port 5667
Message sent. Service: TxnMerge, code: 0, description: Running
Message sent. Service: tasks_TxnMerge, code: 0, description: No task with error.
The line Message sent indicates also that it could connect to a server.

Debug

You can configure the debug level in the environment section of the ini file.
Add the line:
debug = 1
and OpenMakao will write debug informations into the logfile.
Usually you will not need them. Errors and warnings are important and will be written also without debugging into the logfile.

With values 2 – 4 you will get some more debug messages.

Alerts do not appear in Nagios log

First check if the messages had been sent by OpenMakao. A line with „Message sent“ will be there for every configured Nagios service. If there are Java exceptions like:

```
log4j:ERROR NagiosAppender::send_nsca(), Exception thrown
trying to deliver log4j record to Nagios: nagios server=
```

you have to check the [environment] section of the ini file.

If the message had been sent by OpenMakao but the messages still don't appear in the log of Nagios, have a look if the NSCA logfile got any messages. If that also doesn't have messages check if the server might have a different encryption level as the client.
In OpenMakao this is configured in the folder „conf“ in the fielsend_nsca.cfg file.
It will be created if it does not exist with default to: encryption_method=0.
If the server is configured with encryption_method=1, you need to adjust that file accordingly at the client. The file can be changed, OpenMakao will only recreate it if it does not exist. Once created, OpenMakao doesn't change it anymore.
Other encryption_method than 0 or 1 are not supported by OpenMakao.

Some Messages got lost

If it happens that message from the client get lost at Nagios, it might be that Nagios got to many messages at the same time. Usually this should not happen, but OpenMakao sends, depending on configuration, a lot of messages to Nagios.
If you have scheduled them to run at the same time, it might happen that Nagios cannot digest them fast enough, and some will get lost.
The solution is to configure the schedule of the checks to different times.

Status UNKNOWN in Nagios

If you get the status UNKNOWN in Nagios although the status seems to be clear, it might be that you are using another standard language in Siebel. Only DEU and ENU are built in.

Add new languages like described in the chapter „Additional languages“.

You can also redefine the status with the same option, if you would need other Nagios status. It will overwrite the built in values.