Alerter

From Integration Wiki

Overview

The Alerter uses information in the Alerter database to determine if something is an alert. The information is placed there either through the Monitors or by a web service call via JCAPS.

The individual alerts do not always follow the same criteria for reporting. For example, the Alerter will report back if it does not hear from a Monitor after a certain time and it can also report if a queue or topic has too many messages in it.

Database Structure

Schemas

alert

- The alert schema stores information regarding the current alerts and their statuses.
- These tables are populated automatically, or via the web site.

archive

- The archive schema stores all of the previous alerts.
- These tables are populated automatically.

config

- The config schema holds the overall configuration values for the Alerter. These include everything from the name of the host that the monitor runs on to the type of alerts. If it affects multiple schemas, it should be in here.
- Most of these tables have to be populated manually, with the exception being the AlertType and AlertLocation tables, which are populated by the web service, if need be.

directory

- The directory schema contains the directory configuration where to look for files that contain alert information. This was specifically written for machines that are not able to use web service clients, but are able to create a file and send it to an outside machine.
- Has to be configured manually.

diskspace

- The diskspace schema contains the configuration on where to look for the diskspace, and when it should report as an alert.
- Has to be configured manually.

egate

- The egate schema contains all of the configuration for eGate. These tables should be very static.
- Has to be configured manually.

heartbeat

- The heartbeat schema contains the configuration for when to report on no heartbeat.
- Has to be configured manually, but if there is no configuration the Alerter will use a default value since this alert is important.

jms

- The jms configuration contains all of the configuration for connecting to the JMS and when to alert on something.
- JMSName and JMSNotification are the only automatically populated tables, everything else has to be done manually.

log

- The log schema contains the configuration of where to look for the log files, what to look for, and when to alert.
- Has to be configured manually.

notification

- The notification schema contains the configuration for notifying people via emails. This includes configurations from email addresses to escalating notifications.
- Has to be configured manually.

General Notes

- The majority of the schemas have foreign-key dependencies to the config schema.
- If there is a primary-key/foreign-key link, the name of PK and FK should match. For example, HostNameID in config.Server references the HostNameID in config.HostNameID.

Configuration

Config Schema

Engine Type

- The type of the specific app server engine.

Environment

- Test, Production, Certification, etc.

Host Name

- The name of the host that the Monitors or app servers are running on.

Server

- Combination of the HostName, EngineType, and Environment. This is probably the most commonly linked to configuration table.

Site

- Linked to from the Source table, and contains the name and contact information displayed in the web page.

Source

For the most part this is automatically populated, the the SiteID and SourceTypeID do need to be populated afterward. It is ok to populate the Source manually.

Source Type

Used by the web page to determine what type of source the Source is. Usually just used to show that the source is a communication to alert the help desk to contact the client.

Directory Schema

Directory Configuration

- The Host Name and the directory to watch for alerts.

Diskspace Schema

Disk Space Configuration

- The Host Name and the path to watch the disk space.

Disk Space Notification

- Populated via the monitor, contains the current disk space in MBs.

Disk Space Threshold

- When to alert for a specific Host Name.

eGate Schema

- Eventually going to be phased out, and since it is static, not going to be discussed here.

Heartbeat Schema

Heartbeat Configuration

- Contains the Host Name and the minutes between sending in a heartbeat.

Heartbeat Notification

- Contains the Host Name and the last time it reported in.

Heartbeat Threshold

- Contains the Host Name and how long to wait before alerting after the last report time in the Notification.

JMS Schema

JMSConfiguration

- The location of all of the JMS servers, along with the Host Names and Sources they are linked to.

JMSConfigurationProperty

- Contains the properties for connecting to the JMS.
  - The Property Name should be one of these, all tied to the same JMSConfigurationID.
    - 'ServerName': The name of the server where the JMS is hosted
    - 'ServerPort': The port of the JMS server.
    - 'JMSServerType': Either 'OPENMQ' or left blank.

JMSName

- Populated automatically; contains the name of the queues and topics.

JMSNoMessageThreshold

- Links to a queue or topic, and if no message has been seen in a specific time, sends an alert.

JMSNotification

- Populated automatically; contains the Host Name, Source, JMS Name, and the index and count of the messages in the queue or topic.

Log Schema

LogConfiguration

- Contains the Host Name and the configuration of where the log file(s) lives.
- The LogFileName can be a specific log, such as 'server.log', or be every log file in a directory, such as '*.log'.

LogNotification

- Populated Automatically; contains the current notifications sent in from the Monitor that is watching the log.

LogThreshold

- The specific Alert Locations and Alert Types to look for, and when to send an alert.

LogWatchValue

- The specific values to watch for in the log, and the type of notification they should send.

Monitor Schema

MonitorConfiguration

Contains the Host Name, and the amount of time before all of the configurations in the Monitor should be refreshed.

Notification Schema

Chain

- Contains the individual chain, aka, the current level of escalation. This contains how long an alert should stay in a chain, and the Chain Notification Group of who should get the email.

ChainLink

- Contains how the different Chains link together.

ChainNotificationGroup

- Contains the name of the Group and the Email Group Members.

ChainNotificationGroupRotation

- If there is a rotating on-call, this is used to override a specific Chain Notification Group for a specific time.

ChainQualifier

- This matches up a specific Current Alert to a Chain and Schedule.

CurrentAlertChain

- What alert is currently matching the Chain Qualifier.

CurrentMessage

- The Current Alert and who is being notified about it.

EMailGroupMember

- The emails of the people who should be notified, along with how often they should be notified.

EMailGroupMemberLink

- This is where Groups are created, able to link multiple EMails Group Members together.

EMailToSend

- The current emails that are being sent.

Holiday

- The holiday schedule. If a holiday is seen, it is treated like a weekend alert in the Alerter.

MessageType

- The type of message to send out, from a full HTML email to a short SMS style message.

Schedule

- When to actually alert someone about an alert. Not all alerts should contact someone 24/7.

Alert Set-up

Must Be Set-up

These are the tables that have to be set up for everything to work.

1. config.HostName
    - The Alerter knows of an '[ANY]' HostName, and it is recommended that this is created first.
2. config.EngineType
3. config.Environment
4. config.Server

Setting up Disk Space

1. diskspace.DiskSpaceConfiguration
    - Link to config.HostName, give a valid path, and the amount of time between polling the disk space.
2. diskspace.DiskSpaceThreshold
    - Set the config.HostName (match the diskspace.DiskSpaceConfiguration link) and set the size in megabytes that it should alert.

If the size of the watch path goes below the size in megabytes, an AlertType of 'Low Disk Space' will be sent with the AlertLocation being the Host Name.

Setting up the Heartbeat

1. heartbeat.HeartbeatConfiguration
    - Link the config.HostName and set the minutes between refreshes.
2. heartbeat.HeartbeatThreshold
    - Link the config.HostName and set the amount of time to wait for a heartbeat before sending an Alert.

If an Alert is sent, it will be of AlertType 'No Recent Heartbeat' with the AlertLocation being the Host Name.

Setting up the JMS

1. jms.JMSConfiguration
    - Link to the config.HostName, to config.Source (may have to create a Source, name it something human readable, such as 'JMS Server 1'), the SystemName is a human readable name of where the server lives, and the minutes between refreshes.
2. jms.JMSConfigurationProperty
    - For each JMS server, always use the same JMS Configuration ID that was previous created by the above step. The PropertyName of

'ServerName' should have a PropertyValue of the name of the machine where the JMS server runs, the PropertyName of 'ServerPort' should be the port to connect to the JMS, and the PropertyName of 'JMSServerType' should be 'OPENMQ' for all JCAPS 6.

There are two types of alerts that are current sent. One AlertType is 'Index not moving' with the AlertLocation being the SystemName, this occurs if the index has stayed the same between two polling periods. The other AlertType is 'Too many messages' with the AlertLocation of the SystemName, this occurs if there are >500 messages in the queue or topic.

1. jms.JMSNoMessageThreshold
    - Not really needed, but set it up if you want to be notified if a message has not been received on a queue or topic in a period of time.

This will send an AlertType of 'Have Not Received a Message' with the AlertLocation of the SystemName.

Setting up the Log

1. log.LogConfiguration
    - Link to the config.HostName, set the LogName to what you want to see in the Alerts, set the LogDirectory to the directory where the log(s) are, and set the LogFileName.
        - The LogFileName can be a specific log, 'server.log', or a bunch of logs, '*.log'.
2. log.LogWatchValue
    - Link to the config.HostName, set the ValueToWatchFor for a specific value in the log to look for, link to the config.AlertLocationID, and link to the config.AlertType.
        - This means that you can create very specific AlertTypes for different values in the log.
3. log.LogThreshold
    - Use the AlertLocation and AlertType from above and also set the Source that should be reported, along with the number of Notifications to be seen before making an Alert.
        - This is to keep the number of false positives from occuring, for example, if a site disconnects and you roll the transaction back, an alert shouldn't be sent if this occurs once every so often. If the transaction roll's back constantly though, this should be an Alert.

This will through the AlertLocation and AlertType specified in the log.LogThreshold.

Setting up the Notifications

The notifications contain a lot of configuration files because it is able to do a lot, from escalation of alerts to knowing when it is a holiday.

- notification.Schedule
  - Determine what a valid schedule should be. It is ok to have multiple, for example, we currently have 24/7 alerts and 9AM to 5PM only on work day alerts.

- notification.MessageType
  - The only valid MessageTypes the Alerter knows about are 'EMPI-HTML', 'EMPT-Text', 'HTML', 'N/A', 'ShortText', and 'Text'.

- notification.Holiday
  - The listing of holidays, it is ok if nothing is in here. If something qualifies though, the Alerter will treat this date like a weekend.

- notification.EMailGroupMember
  - Set this up with the emails of everyone who should be notified at some point. It should be self explanatory.
  - If you want to setup a group, insert all of the email addresses into this table. Create one more entry for the group, and set the 'IsGroup' value to true. Now, please look at 'notification.EMailGroupMemberLink'.

- notification.EMailGroupMemberLink
  - This links the EMailGroupMember groups to individual EMailGroupMembers.
  - The GroupID is the EMailGroupMember group ID, the MemberID are the individual EMailGroupMemberIDs that should be in the group.

- notification.ChainNotificationGroup
  - Think of this as group for the EMailGroupMember. Set up the name of the group or people in ChainNotificationGroupName, and link to the EMailGroupMember.

- notification.ChainNotificationGroupRotation
  - This is used if there is an on-call rotation. Set the ChainNotificationGroupID to the correct ID, and set up a rotating EMailGroupMemberID, with the correct dates, for which to send.

- notification.Chain

        - This is where the escalation chains are configured, for example, if at first the Alert should go to the help desk, then 30 minutes later to the help desk supervisor, then 30 minutes later to a developer, etc...
        - Add the individual ChainNotificationGroups, and how long the Alert should stay in this chain.

- notification.ChainLink
    - This is where all of the Chains are linked up. This determines that if a chain should escalate, what group should it escalate to next.

- notification.ChainQualifier
    - Possibly the most modified table in notification. This is where all of the alerts qualify to go out to the chains.
    - This is a bunch of keys to other tables, with the only exception being the DelayMinutes, which is used to delay an alert for a specific amount of time.
    - The HostNameID, AlertLocationID, AlertTypeID, and SourceID can all reference a value named '[ANY]' in their tables. This has to be added manually, but will work as a wild card. You could then set up an alert for 'Index not moving' on any host, and alert location, and any source to be alerted to a specific group of people.

- notification.CurrentAlertChain
    - Populated automatically, so not going to worry about it here.

- notification.CurrentMessage
    - Populated automatically, so not going to worry about it here.

- notification.EMailToSend
    - Populated automatically, so not going to worry about it here.

Adding a new Site

Follow these steps when adding a new site to the Alerter:

Gather the Site Information

- If the site already exists (look it up in the site table), then skip this step.
- Compile a list of all queues, topics, and services from each CAPS Connectivity Map for that Site. Only include site specific items.

Here is an example. This is the list of items from the Rogue Site:
svcRogue_HL7Out
svcRogue_HL7In
qRogue_Out
qRogue_In
svcRogue_Normalize
svcRogue_Denormalize
qRogue_Denormalize

- Next get the Site Name and contact information from the task given by the project coordinator.

Another Rogue example:
Site Name: Rogue - Ventanna
Contact: Mr Bob - 509-755-8892

Alerter DB's

- Alerter = CAPS 5.0 & egate
- Alerter2 = CAPS 6.0

Add to the Site Table

- Now create an entry in the config.Site table in the Alerter2 database for this Site and include the contact information.

Add to the Source Table

- Then create entries for each item in your list in the config.Source table in the Alerter2 database. The SiteID should be the one from the previous step.
- Lastly add a SourceTypeID of 1 to your entries in the config.Source table for each "Outbound" entry. This will cause the alert to include the contact information and the words "Please have them reset their side."

In our example list those would be:
svcRogue_HL7Out
qRogue_Out

Specifying a Special Schedule

- If you need to alter the default chain or schedule information you need to make an entry into the notification.ChainQualifier table.

- You can do this by getting the config information from the alert details and then looking up the keys in the corresponding table.
    - For example:
    - HostNameID is found in the config.HostName table, after opening it I can see that this: 'Server: egate2' gives hostnameid = 10

Glossary

- **Alert Location**: This is a general location of where the Alert took place. An example of this is Communication, Processing, or Routing.

- **Alert Type**: This is a more specific type of Alert and should be more specific than Alert Location. An example of this is 'Exception Found', 'Transaction Rolled Back', or 'Unable to Connect to Site'.

- **Engine Type**: This refers to the type of engine installed on the server. An example of this is eGate or CAPS.

- **Environment**: The level of services running on the server. For example, production, test, and development are all different levels of environments.

- **Heart Beat**: The Monitor sends a message every so often that it is still running, this is considered the heart beat.

- **Host Name**: Usually the name of the server, but can be an artificial name for testing, or for running multiple instances of the Monitor on one machine.

- **Monitor**: The Windows service or Linux daemon that monitors specific items on the server. The Monitor, true to its name, only monitors and is designed to be stateless.

- **Server**: The physical, or virtual, machine. The server in the Alerter is comprised of the Host Name, Engine Type, and Environment.

- **Source**: This is the actual thread, process, or log that is causing the Alert. This should be very specific.