Module event-manager-writer

Sommaire

Concept

Le module `event-manger-writer` permet de disposer de la fonctionnalité bac à événements, il gère l'écriture des événements en base de données.

Pour que le bac à événement puisse afficher des événements à jour, il faut absolument que ce module soit activé.

⚠️ Il ne peut y avoir qu'un `event-manager-writer` par base MongoDB.

Donc, par exemple avec 2 brokers sur la même machine :

- soit, vous n'activez le module que sur un Broker
- soit, vous configurez le module pour écrire dans une autre base.

Vous devrez porter une attention particulière sur le volume de votre base d'événements, dû au fonctionnement du module et des informations qu'il stocke.

FONCTIONNEMENT :

- Un élément (*hôte, cluster, checks*) peut avoir :
  - peu de changements d'état, consommant ainsi une place raisonnable,
  - avoir une oscillation de son état (*alternance de pannes et de retour OK*), et ainsi enregistrer beaucoup de changements d'état.
- Le souci est que la place prise par un changement d'état est **variable**, car chaque événement stocké contient le résultat court et le résultat long.
  - **Surveiller la taille de votre base d'événement** avec le check : shinken-broker-module-event-manager-writer.
- Pour un élément donné, seul un changement de statut (*OK, Attention, Critique, Inconnu*) ou un changement de contexte (*Flapping, Downtime, Acknowledged*) va créer une nouvelle entrée dans la liste des événements.

A NOTER

Un cas modifie une entrée déjà enregistrée : Quand le statut d'un élément passe de l'**état non confirmé** à l'**état confirmé**, sans autre changement de son statut ou de son contexte, l'événement le décrivant est mis à jour.

- La colonne **Confirmation de Statut** est modifiée pour indiquer que ce statut a été confirmé.
- la colonne **Date de confirmation de Statut** est modifiée pour indiquer à partir de quand le statut a été confirmé, la colonne **Date du changement** indiquant toujours quand le statut (*en état non confirmé*) a démarré.

Ce module fonctionne de paire avec le module : `event-manager-reader` qui permet à la Webui d'afficher les informations (*voir la page Module event-manager-reader*).

Activation du module

Le module `event-manager-writer` est un module qui peut être activé seulement sur le démon Broker.

- L'activation du module s'effectue en ajoutant le nom de ce module dans le fichier de configuration du démon Broker.
- Pour ce faire, ouvrer le fichier de configuration du Broker à l'emplacement **/etc/shinken/brokers/nom_du_broker.cfg**, et ajouter le nom de votre module "`event-manager-writer`".

Exemple: par défaut, nous livrons un module dont le nom est "`event-manager-writer`":

```
define broker {
    [...]
    modules                  Module 1, Module 2, Module 3, event-manager-writer
    [...]
}
```

Pour prendre en compte le changement de configuration, redémarrer l'Arbiter:

```
service shinken-arbiter restart
```

Configuration

La configuration du module que nous livrons par défaut se trouve dans le fichier **/etc/shinken/modules/event_manager_writer.cfg**

- Vous trouverez aussi systématiquement un exemple dans **/etc/shinken-user-example/configuration/daemons/brokers/modules /event_manager_writer/event_manager_writer-example.cfg**

Exemple de fichier de configuration

```
#===============================================================================
# event manager
#===============================================================================
# Daemons that can load this module:
#  - broker (to save events information into a mongodb database)
# This module compute and save event for event manager
#===============================================================================

define module {

    #  #
    #      MODULE IDENTITY      #
    #  #

    #  Module name [ Must be unique ]                                    [ MANDATORY ]
    #
    module_name                                  event-manager-writer

    #  Module type [ Do not edit ]                                       [ MANDATORY ]
    #
    module_type                                  event_container

    #  #
    #      MODULE OPTIONS      #
    #  #

    #  Number of day the events are kept in database
    #
    #          Default : 30 ( days )
    #
    # day_keep_data                                 30

    #  #
```

```
#      DATABASE CONNECTION      #
#  #

#   MongoDB parameters

#  MongoDB uri definition . You can find the mongodb uri syntax at
#  https://docs.mongodb.com/manual/reference/connection-string/
#
#          Default : mongodb://localhost/?w=1&fsync=false
#
# uri                                           mongodb://localhost/?w=1&fsync=false

#  Which database contains events data
#
#          Default : event_container
#
# database                                      event_container

#  SSH tunnel activation to secure your mongodb connection
#  That will allow all mongodb to be encrypted & authenticated with SSH
#
#          ...     : Enable  => 1 ( enable ssh tunnel )
#          Default : Disable => 0 ( disable ssh tunnel )
#
# use_ssh_tunnel                                0

#  If the SSH connection goes wrong, then retry use_ssh_retry_failure time before_shinken_inactive
#
#          Default : 1 ( try )
#
# use_ssh_retry_failure                         1

#  SSH user to connect to the mongodb server.
#
#          Default : shinken
#
# ssh_user                                      shinken

#  SSH keyfile to connect to the mongodb server.
#
#          Default : ~shinken/.ssh/id_rsa
#
# ssh_keyfile                                   ~shinken/.ssh/id_rsa

#  SSH Timeout used to test if the SSH tunnel is viable or not, in seconds.
#
#          Default : 10 ( seconds )
#
# ssh_tunnel_timeout                            10

#   AutoReconnect Management

#  When MongoDB require you to reconnect ( For example, It can occur when a new PRIMARY is elected
#  in a MongoDB cluster ), it will raised the MongoDB AutoReconnect exception.

#  How many try to reconnect before module go in error
#
#          Default : 4 ( try )
#
# auto_reconnect_max_try                        4

#  Time between each try
#
#          Default : 3 ( seconds )
#
# auto_reconnect_sleep_between_try              3

#  NOTE: Change these values only if you have a MongoDB cluster and you change the
#        heartbeatTimeoutSecs of your MongoDB replica set
#        The value of auto_reconnect_max_try * auto_reconnect_sleep_between_try must be higher than
#        heartbeatTimeoutSecs in the rs.conf(); of your MongoDB replica set.
```

```
    #  #
    #      WORKERS IN THE BROKER      #
    #  #

    #  This module will use workers in the broker, each worker will manage a shard of all hosts/checks.
    #  This parameter is used by the broker to set the number of workers.
    #  Each worker will use one CPU, which will balance the event processing load among CPUs.
    #
    #          Default : 1 ( worker )
    #
    # broker_module_nb_workers                          1

    #  #
    #      INTERNAL OPTIONS      #
    #  #

    #  INTERNAL : DO NOT EDIT FOLLOWING PARAMETER WITHOUT YOUR DEDICATED SUPPORT

    #  Broker idle time before considering that Shinken is inactive.
    #  Use this if you have Broker loop time that exceeds 30 seconds
    #
    #          Default : 30 ( seconds )
    #
    # time_before_shinken_inactive                      30

}
```

Détails des sections composant le fichier de configuration

Identification du module

Il est possible de définir plusieurs instances de module de type "event-manager-writer" dans votre architecture Shinken.

- Chaque instance devra avoir un nom unique.

Nom	Type	Unité	Défaut	Commentaire
`module_name`	Texte	---	**event-manager-writer**	**Nous vous conseillons de choisir un nom en fonction de l'utilisation du module pour que votre configuration soit simple à maintenir.** Doit être unique.
`module_type`	Texte	---	**event_container**	Ne peut être modifié.

Taille du bac d'événement en nombre de jours

```
    #  #
    #      MODULE OPTIONS      #
    #  #

    #  Number of day the events are kept in database
    #
    #          Default : 30 ( days )
    #
    # day_keep_data                                     30
```

Le paramètre "`day_keep_data`" permet de choisir **le nombre de jours qu'un événement sera gardé dans votre base**.

- Si votre base MongoDB prend trop de place sur le disque, (à monitorer avec le check : shinken-broker-module-event-manager-writer) .
- Il est possible de diminuer le nombre de jours sauvegardés.

Nom	Type	Unité	Défaut	Commentaire
day_keep_data	Entier	jours	**30**	Durée en nombre de jours d'un événement dans le bac à événement.

Accès à la base MongoDB

Cette configuration s'effectue dans le fichier de configuration du module.

Pour se connecter à la base MongoDB utilisé pour le stockage des données, 2 méthodes sont disponibles:

- **Connexion directe**: Par défaut, mais non sécurisée.
- **Tunnel SSH**: Shinken se connecte à la base MongoDB au travers d'un module SSH pour plus de sécurité

Configuration des paramètres communs aux deux méthodes

```
#   MongoDB parameters

#  MongoDB uri definition . You can find the mongodb uri syntax at
#  https://docs.mongodb.com/manual/reference/connection-string/
#
#          Default : mongodb://localhost/?w=1&fsync=false
#
# uri                                                 mongodb://localhost/?w=1&fsync=false

#  Which database contains events data
#
#          Default : event_container
#
# database                                            event_container
```

Nom	Type	Unité	Défaut	Commentaire
uri	Texte	URL	**mongodb://localhost/?safe=true**	Vous pouvez trouver la syntaxe de l'uri de MongoDB à l'adresse https://docs.mongodb.com/manual/reference/connection-string/
database	Texte	---	**shinken**	Nom de la base de données où sont stockés les données événements

Connexion directe au serveur MongoDB

Par défaut, le module se connecte de manière directe à la base MongoDB pour y lire et écrire les données.

Dans la configuration du module, ceci correspond au paramètre "use_ssh_tunnel" à 0.

Cette méthode de connexion a pour avantage d'être facile à configurer au niveau de Shinken. Par contre, elle oblige à permettre l'accès à la base MongoDB au monde extérieur, et donc s'exposer à des problèmes de sécurité.

La sécurisation de la base MongoDB est bien sur toujours possible (voir Sécurisation des connexions aux bases MongoDB) mais bien plus complexe à mettre en place. La méthode de connexion par SSH est donc préférable pour des raisons pratiques et de sécurité.

Connexion par SSH au serveur MongoDB

```
#  SSH tunnel activation to secure your mongodb connection
#  That will allow all mongodb to be encrypted & authenticated with SSH
#
#          ...     : Enable  => 1 ( enable ssh tunnel )
#          Default : Disable => 0 ( disable ssh tunnel )
#
# use_ssh_tunnel                                    0

#  If the SSH connection goes wrong, then retry use_ssh_retry_failure time before_shinken_inactive
#
#          Default : 1 ( try )
#
# use_ssh_retry_failure                             1

#  SSH user to connect to the mongodb server.
#
#          Default : shinken
#
# ssh_user                                          shinken

#  SSH keyfile to connect to the mongodb server.
#
#          Default : ~shinken/.ssh/id_rsa
#
# ssh_keyfile                                       ~shinken/.ssh/id_rsa

#  SSH Timeout used to test if the SSH tunnel is viable or not, in seconds.
#
#          Default : 10 ( seconds )
#
# ssh_tunnel_timeout                                10
```

Le module peut également se connecter par tunnel SSH à la base MongoDB, pour des raisons de sécurité.

En effet, le paramétrage de MongoDB permet de définir sur quelle interface réseau ce dernier écoute les requêtes.
En n'autorisant seulement interface réseau avec l'adresse 127.0.0.1, cela évite d'ouvrir la base au monde extérieur.

Dans la configuration de la base MongoDB (**/etc/mongod.conf**), assurez-vous que le paramètre "*bind_ip*" est positionné pour n'écouter que sur l'interface locale:

- `bind_ip=127.0.0.1`

Dans cette configuration la base MongoDB écoute que sur l'interface réseau local, pour que le module se connecte, il faut passer par un tunnel SSH.
Pour ce faire activer les options suivantes :

Nom	Type	Unité	Défaut	Commentaire
`use_ssh_tunnel`	Booléen	---	**0**	• 1 : Connection par tunnel SSH • 0 : Connection direct
`use_ssh_retry_failure`	Entier	Nombre d'essais	**1**	Spécifie le nombre supplémentaire de tentatives lors de l'établissement du tunnel SSH si ce dernier n'arrive pas à être établi
`ssh_user`	Texte	Utilisateur unix	**shinken**	L'utilisateur avec lequel le tunnel sera établi
`ssh_keyfile`	Texte	Chemin de fichier	**~shinken/.ssh/id_rsa**	La clé SSH privée présente sur le serveur Shinken qui sera utilisé pour établir le tunnel.

`ssh_tunnel_timeout`	Entier	Secondes	**10**	Spécifie le timeout en secondes de la vérification du tunnel SSH avant que la connexion vers MongoDB soit effectuée

Pour configurer les clés SSH à utiliser, voir la page Création automatique et gestion de la clé SSH de l'utilisateur shinken

Gestion de l'auto reconnexion avec un cluster MongoDB

```
#   AutoReconnect Management

#  When MongoDB require you to reconnect ( For example, It can occur when a new PRIMARY is elected
#  in a MongoDB cluster ), it will raised the MongoDB AutoReconnect exception.

#  How many try to reconnect before module go in error
#
#         Default : 4 ( try )
#
# auto_reconnect_max_try                          4

#  Time between each try
#
#         Default : 3 ( seconds )
#
# auto_reconnect_sleep_between_try                3

#  NOTE: Change these values only if you have a MongoDB cluster and you change the
#        heartbeatTimeoutSecs of your MongoDB replica set
#        The value of auto_reconnect_max_try * auto_reconnect_sleep_between_try must be higher than
#        heartbeatTimeoutSecs in the rs.conf(); of your MongoDB replica set.
```

Définitions

- **Primaire:** nom de MongoDB pour désigner un serveur maître, le serveur sur lequel il est possible de faire des requêtes d'écriture dans la base.
- **Election** : processus de MongoDB pour choisir un nouveau membre Primaire si le membre Primaire devient inaccessible

Voir : Haute disponibilité de la base MongoDB

Dans le cas de l'utilisation d'un cluster MongoDB, lorsque le membre Primaire devient inaccessible une nouvelle élection est déclenchée ce qui provoque une coupure temporaire de l'accès à la base.

Dans le but de ne pas interrompre le service, le module SLA va se reconnecter automatiquement au cluster MongoDB.
Pour ce faire il va faire un nombre d'essais égaux au paramètre "`auto_reconnect_max_try` " avec une pause de X secondes entre chaque essai (correspondant au paramètre "`auto_reconnect_sleep_between_try`").

Par défaut pour MongoDB le temps maximum avant qu'un membre Primaire soit considéré comme indisponible et qu'une nouvelle élection ait lieu est de 10 secondes.
Voir : "heartbeatTimeoutSecs" donné par la commande rs.conf(); dans un shell de MongoDB.

Nom	Type	Unité	Défaut	Commentaire
`auto_reconnect_max_try`	Entier	Nombre d'essais	**4**	Nombre d'essais de reconnexion à la base
`auto_reconnect_sleep_between_try`	Entier	Secondes	**3**	Temps entre chaque essai en seconde

Les valeurs par défauts du fichier laisse 12 secondes, ce qui est amplement suffisant avec la configuration par défaut de MongoDB.

Il est conseillé de ne pas modifier ces valeurs.

Utilisation des workers du module event-manager-writer

```
#  #
#      WORKERS IN THE BROKER      #
#  #

#  This module will use workers in the broker, each worker will manage a shard of all hosts/checks.
#  This parameter is used by the broker to set the number of workers.
#  Each worker will use one CPU, which will balance the event processing load among CPUs.
#
#          Default : 1 ( worker )
#
# broker_module_nb_workers                          1
```

Le paramètre "`broker_module_nb_workers`" va déterminer combien de fois le module va se cloner pour gérer le flux de donnée à enregistrer afin de repartir cette charge sur plusieurs CPU.

Il est possible de changer ce paramètre si l'utilisation CPU du processus : "*NOM DU BROKER* [- Module: event-manager-writer][Worker: 0]" est trop élevé.

Nom	Type	Unité	Défaut	Commentaire
`broker_module_nb_workers`	Entier	X workers	**1**	Nombre de workers qui traitent le flux de donnée pour sauvegarder les événements dans la base MongoDB. (le traitement est réparti sur les workers)

Ne pas dépasser le nombre de core cpu de la machine : cela serait contre-productif pour les performances.

Options internes

```
#  #
#      INTERNAL OPTIONS      #
#  #

#  INTERNAL : DO NOT EDIT FOLLOWING PARAMETER WITHOUT YOUR DEDICATED SUPPORT

#  Broker idle time before considering that Shinken is inactive.
#  Use this if you have Broker loop time that exceeds 30 seconds
#
#          Default : 30 ( seconds )
#
# time_before_shinken_inactive                      30
```

Ces paramètres sont dédiés au fonctionnement interne au module, il est fortement recommandé de ne pas les modifier sans votre support dédié.

Nom	Type	Unité	Défaut	Commentaire
`time_before_shinken_inactive`	Entier	Secondes	**30**	Temps d'inactivité du Broker avant de considérer que Shinken est inactif. Utilisez cette option si vous avez un temps de boucle du Broker qui dépasse 30 secondes.