

Module MongoDB

Sommaire

Concept

Activation du module

Exemple d'activation du module nommé "Mongodb" sur le module nommé "WebUI" (configuration livrée par défaut par Shinken)
Créer un nouveau module de type MongoDB

Configuration

Exemple de fichier de configuration

Détails des sections composant le fichier de configuration

Identification du module

Accès à la base MongoDB

Configuration de l'URI de connexion et de l'authentification par mot de passe

Connexion directe au serveur MongoDB

Connexion par SSH au serveur MongoDB

Gestion de la reconnexion automatique

Concept

Le module MongoDB est un module qui permet au module WebUI (*l'interface de visualisation*) du Broker d'accéder aux données stockées dans la base de données MongoDB.

- Il permet d'afficher les portails sauvegardés (voir la page [Vue - Les Portails](#)) par l'utilisateur et les favoris des utilisateurs (voir la page [Volet Favoris \(Gestion des Vues \)](#)).
- Il doit donc être listé dans la liste des modules du module WebUI (voir la page [Module WebUI](#))



Ce module est obligatoire pour pouvoir accéder à l'interface de visualisation.



S'il y a plusieurs configurations de WebUI présente dans l'architecture, il faut activer le module MongoDB dans chaque configuration.

- Exemple plusieurs WebUI sur le même broker.
- ou une sur différents Brokers qui n'utilise pas la même définition.

Activation du module

Les modules de type "mongodb" sont des modules qui doivent être activés sur un module de type "webui".

- L'activation du module s'effectue en ajoutant le **nom** du module dans la configuration du **module parent**.
 - Pour cela, il faut ouvrir le fichier de configuration du **module parent** (de type "webui"), et ajouter dans le paramètre **modules**, le nom du module de type "mongodb".
- Il est possible de faire plusieurs modules de type "mongodb".
 - Cela permet, **par exemple**, d'avoir des configurations différentes en fonction des royaumes.
- S'il y a plusieurs modules de type "webui" présents dans l'architecture, il ne faut pas oublier d'activer le module de type "mongodb" dans la configuration de chacun d'eux.
- Contraintes :
 - Activable uniquement sur un module de type "webui" (voir la page [Configuration du module WebUI](#)).
 - Il ne peut y avoir qu'un seul module de type "mongodb" sur un module de type "webui".

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

```
service-shinken-arbiter restart
```

Exemple d'activation du module nommé "Mongodb" sur le module nommé "WebUI" (configuration livrée par défaut par Shinken)

L'exemple suivant

- active le module "Mongodb"
- sur le module "WebUI", dont la configuration est dans le fichier `/etc/shinken/modules/webui.cfg`

Modification dans le fichier du module `/etc/shinken/modules/webui.cfg` :

```
define module {
    [...]
    modules          Module 1, Module 2, Module 3, Mongodb
    [...]
}
```

Puis redémarrage de l'Arbiter

```
service-shinken-arbiter restart
```

Créer un nouveau module de type MongoDB

Pour pouvoir configurer un module de type "MongoDB", il faut faire un nouveau fichier de configuration grâce au fichier d'exemple fourni par défaut.

- Pour commencer, il faut choisir le nom du nouveau module.
 - Pour l'exemple, on l'appelle "Mon-Module-MongoDB".
 - Remplacer dans l'exemple le mot "Mon-Module-MongoDB" par le nom qui a été choisi.
- Puis il faut créer le fichier de configuration :
 - Copier le fichier de définition du module d'exemple `/etc/shinken-user-example/configuration/daemons/brokers/modules/webui/modules/mongodb/modules/mongodb-example.cfg` dans le répertoire de définition des modules `/etc/shinken/modules`.
(Exemple : `/etc/shinken/modules/MongoDB__Mon-Module-MongoDB.cfg`)

```
cp /etc/shinken-user-example/configuration/daemons/brokers/modules/webui/modules/mongodb/modules/mongodb-example.cfg /etc/shinken/modules/MongoDB__Mon-Module-MongoDB.cfg
```

- Ensuite, il faut modifier le fichier nouvellement créé pour configurer le nouveau module
 - Il faut vérifier que le fichier appartienne à l'utilisateur shinken et qu'il possède le droit d'édition. Si ce n'est pas le cas, il faut effectuer les commandes suivantes :

```
chown -R shinken:shinken /etc/shinken/modules/MongoDB__Mon-Module-MongoDB.cfg
chmod u+w /etc/shinken/modules/MongoDB__Mon-Module-MongoDB.cfg
```

- On change le nom du module en "Mon-Module-MongoDB" dans le fichier `/etc/shinken/modules/MongoDB__Mon-Module-MongoDB.cfg`

```
...
# Module name [ Must be unique ]                                [
MANDATORY ]
#
module_name                Mon-Module-MongoDB
...
```

- Ensuite, il faut ajouter le nouveau module dans le module de type "WebUI" correspondant.
 - Dans notre exemple, on ajoute le module "Mon-Module-MongoDB" au module "WebUI" définie dans le fichier `/etc/shinken/modules/webui.cfg`

```
define module {
    [...]
    modules          Module 1, Module 2, Module 3, Mon-Module-MongoDB
    [...]
}
```

- Pour finir il faut redémarrer l'Arbiter pour que le Broker puisse prendre en compte ce nouveau module.

```
service-shinken-arbiter restart
```

Configuration

La configuration du module se trouve par défaut dans le fichier `/etc/shinken/modules/mongodb.cfg`.

- un exemple dans `/etc/shinken-user-example/configuration/daemons/brokers/modules/webui/modules/mongodb-example.cfg`.

Exemple de fichier de configuration

```
# CFG_FORMAT_VERSION 1 ( SHINKEN : DON'T TOUCH THIS LINE )

=====
# Mongodb
=====
# Modules that can load this module:
# - WebUI
# This module is mandatory for the WebUI module. The WebUI use this definition
# to access the mongodb data (like hive and user shares)
=====

define module {

    # #
    #     MODULE IDENTITY      #
    # #

    # Module name [ Must be unique ]                                [ MANDATORY ]
    #
    module_name                                Mongodb

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

    # #
    #     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
    #
    # mongodb__database__uri                                mongodb://localhost/?w=1&fsync=false

    # Which database contains events data
    #
    #     Default : shinken
    #
    # mongodb__database__name                                shinken

    # username/password to authenticate to MongoDB.
    # Both parameters must be provided for authentication to function correctly.
    #
    # broker__module_webui__module_mongodb__database__username

    #
    # broker__module_webui__module_mongodb__database__password

    # 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 )
#
# mongodb__database__use_ssh_tunnel                                0

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

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

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

# SSH Timeout used to test if the SSH tunnel is viable or not, in seconds.
#
#           Default : 10 ( seconds )
#
# mongodb__database__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 : 5 ( try )
#
# mongodb__database__retry_connection_X_times_before_considering_an_error 5

# Time between each try
#
#           Default : 5 ( seconds )
#
# mongodb__database__wait_X_seconds_before_reconnect            5

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

```

Détails des sections composant le fichier de configuration

Identification du module

Il est possible de définir plusieurs instances de module de type "mongodb" dans l'architecture Shinken.

- Chaque instance devra avoir un nom unique.

Nom	Type	Unité	Défaut	Description
-----	------	-------	--------	-------------

module_name	Texte	---	Mongodb	Il est conseillé de choisir un nom en fonction de l'utilisation du module pour que la configuration soit simple à maintenir. Doit être unique.
module_type	Texte	---	mongodb	Ne peut être modifié.

Accès à la base MongoDB

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

Pour se connecter à la base MongoDB utilisée 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 de l'URI de connexion et de l'authentification par mot de passe

```
# #
# 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
#
# mongodb__database__uri                mongodb://localhost/?w=1&fsync=false

# Which database contains events data
#
# Default : shinken
#
# mongodb__database__name                shinken

# username/password to authenticate to MongoDB.
# Both parameters must be provided for authentication to function correctly.
#
# broker__module__webui__module__mongodb__database__username

#
# broker__module__webui__module__mongodb__database__password
```

Nom	Type	Unité	Défaut	Description
uri	Texte	URL	mongodb://localhost/?safe=true	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 stockées les données SLA.

broker__module_webui__module_mongodb__database__username	Texte	---		Utilisateur pour l'authentification avec mot de passe à la base MongoDB. Utile uniquement si l'activation par mot de passe a été activé (voir la page MongoDB - activation de l'authentification par mot de passe)
broker__module_webui__module_mongodb__database__password	Texte	---		Mot de passe de l'utilisateur utilisé pour l'authentification avec mot de passe à la base MongoDB. Utile uniquement si l'activation par mot de passe a été activé (voir la page MongoDB - activation de l'authentification par mot de passe)

Connexion directe au serveur MongoDB

Par défaut, le module se connecte de manière directe à la base MongoDB définie avec les paramètres communs listés ci-dessus pour y lire et écrire les données.

Dans la configuration du module, on sait que la connexion se fait de manière directe lorsque le paramètre "use_ssh_tunnel" est 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 sûr toujours possible (voir la page [Sécurisation des connexions aux bases MongoDB](#)) mais bien plus complexe à mettre en place. La méthode de connexion par SSH est ainsi 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 de données au monde extérieur.

Dans la configuration de la base MongoDB (**/etc/mongod.conf**), il faut vérifier 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	Description
<code>use_ssh_tunnel</code>	Booléen	---	0	• 1 : Connexion par tunnel SSH. • 0 : Connexion directe.
<code>use_ssh_retry_failure</code>	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.
<code>ssh_user</code>	Texte	Utilisateur unix	shinken	L'utilisateur avec lequel le tunnel sera établi.
<code>ssh_keyfile</code>	Texte	Chemin de fichier	~shinken/.ssh/id_rsa	La clé SSH privée présente sur le serveur Shinken qui sera utilisé pour établir le tunnel.
<code>mongo_timeout</code>	Entier	secondes	10	Spécifie le timeout en secondes de la connexion à MongoDB.

Pour configurer les clés SSH à utiliser, voir la page [Création automatique et gestion de la clé SSH de l'utilisateur shinken](#).

Gestion de la reconnexion automatique

```
# 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 : 5 ( try )
#
# mongodb__database__retry_connection_X_times_before_considering_an_error 5

# Time between each try
#
#         Default : 5 ( seconds )
#
# mongodb__database__wait_X_seconds_before_reconnect 5

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

La reconnexion automatique permet au module de se reconnecter à Mongo dans le cas où :

- Il y a une perte de connexion suite à un problème réseau ou à un redémarrage de mongo.
- 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.



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.

Élection : processus de MongoDB pour choisir un nouveau membre Primaire si le membre Primaire devient inaccessible

Voir : [Haute disponibilité de la base MongoDB \(mise en place d'un cluster\)](#)

Dans le but de ne pas interrompre le service lorsque l'un de ces cas survient, le module "mongodb" va se reconnecter automatiquement. Pour cela, il va faire un nombre d'essais égal au paramètre "mongodb__database__retry_connection_X_times_before_considering_an_error" avec une pause de X secondes entre chaque essai (correspondant au paramètre "mongodb__database__wait_X_seconds_before_reconnect").



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	Description
mongodb__database__retry_connection_X_times_before_considering_an_error	Entier	Nombres d'essais	5	Nombre d'essais de reconnexion à la base.

mongodb__database__wait_X_seconds_before_reconnect	Entier	Secondes	5	Temps entre chaque essai en seconde.
--	--------	----------	---	--------------------------------------

Les valeurs par défauts du fichier laissent 25 secondes, ce qui est amplement suffisant avec la configuration par défaut de MongoDB.



Il est conseillé de ne pas modifier ces valeurs.