Modèle windows_by_snmp_services-by-name

Sommaire

Description

Le module Graphite-Perfdata permet d'envoyer et stocker les métriques dans un serveur Graphite (*via Carbon*).

Il est possible de modifier des paramètres (*comme l'adresse du serveur Graphite ou son port*) via le fichier de configuration ci-dessous.

Activation du module

Le module Graphite-Perfdata 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 "`Graphite-Perfdata`" .

Exemple: par défaut, nous livrons un module dont le nom est "`Graphite-Perfdata`":

```
define broker {
    ...
    modules                  Module 1, Module 2, Module 3, Graphite-Perfdata
    ...
}
```

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

```
service shinken-arbiter restart
```

Configuration

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

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

Exemple de fichier de configuration

```
#================================================================================
# Graphite-Perfdata
#================================================================================
# Daemons that can load this module:
# - broker
# This module send metrics into a graphite (carbon) server
#================================================================================


define module {

    #======== Module identity =========
    # Module name. Must be unique
    module_name                 Graphite-Perfdata

    # Module type (to load module code). Do not edit.
    module_type                 graphite_perfdata

    #======== 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 metrology processing load among CPUs.
    # default: 1
    broker_module_nb_workers     1

    #======== Graphite address =========
    # host:  graphite server address (ip or fqdn)
    host                        localhost

    # port:  TCP port of the graphite server used for writing data
    port                        2003

    # webapp_port: TCP port of the graphite server used for reading data
    webapp_port                 80


    #======== realm filtering =========
    # By default, this module will save metrics from all realm and subrealms of the broker realm.
    # You can use realm_store_only to save only the realm you want into the graphite server
    #realm_store_only           Realm1, Realm2, Realm3


    #======== Storage tuning ========
    # For performance purpose, we can globally disable storage of warning and/or error thresholds
    # default: 1 => store_warnig_threshold
    # broker_module_graphite_perfdata_store_warning_threshold 1
    #
    # default: 1 => store error_threshold
    # broker_module_graphite_perfdata_store_error_threshold   1
}
```

Détails des sections composant le fichier de configuration

Identification du module

Il est possible de définir plusieurs instances de module de type Graphite-Perfdata dans votre architecture Shinken .

- Chaque instance devra avoir un nom unique.

Nom	Type	Unité	Défaut	Commentaire

`module_name`	Texte	---	**Graphite-Perfdata**	**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	---	**graphite_perfdata**	Ne peut être modifié.

Options du module

```
define module {
        ...
    #======== 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 metrology processing load among CPUs.
    # default: 1
    broker_module_nb_workers     1


    #======== Graphite address =========
    # host:  graphite server address (ip or fqdn)
    host                        localhost

    # port:  TCP port of the graphite server used for writing data
    port                        2003

    # webapp_port: TCP port of the graphite server used for reading data
    webapp_port                 80


    #======== realm filtering =========
    # By default, this module will save metrics from all realm and subrealms of the broker realm.
    # You can use realm_store_only to save olly the realm you want into the graphite server
    #realm_store_only           Realm1, Realm2, Realm3

    #======== Storage tuning ========
    # For performance purpose, we can globally disable storage of warning and/or error thresholds
    # default: 1 => store_warnig_threshold
    # broker_module_graphite_perfdata_store_warning_threshold 1
    #
    # default: 1 => store error_threshold
    # broker_module_graphite_perfdata_store_error_threshold   1
    ...
}
```

Nom	Type	Unité	Défaut	Commentaire
`broker_module_nb_workers`	Entier	---	**1**	Ce module utilise des workers dans le broker, chaque worker va gérer une part des hôtes/checks. Ce paramètre est utilisé par le broker pour choisir le nombre de workers. Chaque worker va utiliser un CPU, cela permet de répartir la charge du calcul de métrologie sur plusieurs CPU.
`host`	IP | fqdn	---	**localhost**	Adresse du serveur Graphite.

`port`	Entier	---	**2003**	Port du serveur Graphite utilisé pour écrire les données.
`webapp_port`	Entier	---	**80**	Port du serveur Graphite utilisé pour lire les données.
`realm_store_only`	Liste Texte	---		Par défaut, ce module sauvegarde les métriques de tout le royaume du broker et ces sous-royaumes. Vous pouvez choisir de sauvegarder dans uniquement certains royaumes. Le nom des royaumes est renseigné à la suite séparé par une virgule. Exemple: `'Realm1, Realm2, Realm3'`
`broker_module_graphite_perfdata_store_warning_threshold`	Booléen	---	**1**	Permets d'activer ou désactiver la sauvegarde des seuils d'avertissements (*1 pour activer, 0 pour désactiver*).
`broker_module_graphite_perfdata_store_error_threshold`	Booléen	---	**1**	Permets d'activer ou désactiver la sauvegarde des seuils critiques (*1 pour activer, 0 pour désactiver*).

Vérification du bon fonctionnement du module

Vérification de l'état du module

Il est possible de récupérer l'état du module avec la commande suivante :

```
shinken-healthcheck
```

Voir la page suivante pour la description des messages de retour de la commande : Shinken-healthcheck - Vérifier le bon fonctionnement de Shinken Entreprise - Chapitre - Vérification de la configuration de la sauvegarde des données de métrologie

Si le module à un problème de connexion avec Graphite

Il existe une série de vérification que vous pouvez réaliser afin de tester l'accès à votre base Graphite, qui comprend :

- La vérification du processus carbon-cache,
- La vérification du port d'écoute,
- La vérification du firewall,

Voir la page suivant pour la description de comment réaliser ces vérifications : Base de métrologie (Graphite) - Chapitre Vérification de carbon-cache, le demon écrivain