Reactionner - GLOBAL - Logs de gestion des modules - chapitre [MODULES-MANAGER]

Sommaire

Import des modules depuis /opt/shinken/modules - chapitre [CODE-LOADING]

Chaque démon (*ou module qui possèdes d'autres modules*) va démarrer son gestionnaire de module "modules-manager" qui va charger chaque module présent dans **/opt/shinken/modules**. Pour qu'un module soit chargé, il faut qu'il soit présent dans un répertoire.

Chargement du code

Le "modules-manager" va regarder si le fichier "module_info.json" placé dans le répertoire du module existe. S'il n'existe pas un log en **WARNING** sera affiché.

```
[YYYY-MM-DD HH:MM:SS] WARNING : [ DAEMON_OR_MODULE_NAME  ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MY_MODULE_DIRECTORY ] Missing 'module_info.json' in directory /opt/shinken/modules
/MY_MODULE_DIRECTORY/module_info.json. Module code will be loaded anyway.
```

Après que le "modules-manager" ait regardé si le fichier "module_info.json" existe, il va regarder si la clé "daemons" est bien présente dans le fichier. Si elle n'existe pas un log en **WARNING** sera affiché.

```
[YYYY-MM-DD HH:MM:SS] WARNING : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MY_MODULE_DIRECTORY ] Missing 'daemons' key in the 'module_info.json' in directory /opt/shinken
/modules/MY_MODULE_DIRECTORY/module_info.json. Module code will be loaded anyway.
```

Si le "modules-manager" n'a pas réussi à lire le fichier à cause d'un problème de formatage, un log en **WARNING** sera affiché.

```
[YYYY-MM-DD HH:MM:SS] WARNING : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MY_MODULE_DIRECTORY ] 'module_info.json' file in directory /opt/shinken/modules/MY_MODULE_DIRECTORY
/module_info.json is malformed. Module code will be loaded anyway.
```

Si le "modules-manager" n'a pas réussi à lire le fichier à cause d'un problème de permissions, un log en **WARNING** sera affiché.

```
[YYYY-MM-DD HH:MM:SS] WARNING : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MY_MODULE_DIRECTORY ] Read permission denied to file 'module_info.json' in directory /opt/shinken
/modules/MY_MODULE_DIRECTORY/module_info.json. Module code will be loaded anyway.
```

Le "modules-manager" annonce qu'il va commencer à charger le code de son module. Un log en **INFO** apparaîtra.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MY_MODULE_DIRECTORY  ] Starting to load the module code from directory /opt/shinken/modules
/MY_MODULE_DIRECTORY.
```

Si le "modules-manager" trouve une exception Python durant le procédé d'import du module, un log en **ERROR** avec l'exception sera affiché et le module ne sera pas importé.

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MY_MODULE_DIRECTORY ] Import module MODULE_NAME failed: EXCEPTION.
```

À la fin la fin de l'import du code le "modules-manager" affiche un résumé de l'import dans un log en **INFO.**

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] A total of 9
Shinken Enterprise modules are available for this daemon/module ( DAEMON_OR_MODULE_NAME ): MODULES_LIST_NAME
(on a total of 40, loaded in 1.844s)
```

Chargement du module en tant que module Python

Lors du chargement du code, le "modules-manager" va vérifier si un module peut être importé en tant que package Python. En cas d'échec un log en **I NFO** sera affiché.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MY_MODULE_DIRECTORY ] Try to import code from /opt/shinken/modules/MY_MODULE_DIRECTORY as python
module.
```

En cas d'échec, 2 situations sont possibles:

- Si le module n'utilise pas les fonctionnalités propre à un module python (*import d'un fichier local, ...*), un log en INFO sera affiché.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MODULE_NAME ] Failed to import the directory /opt/shinken/modules/MY_MODULE_DIRECTORY as a python
module. If this is not a python module, this is not a problem..
```

- Dans le cas contraire, il se peut que l'échec de chargement du module soit lié à un problème d'import de fichier en Python. Les 3 logs en **ERR OR** seront affichés :
    - Le premier log indique que l'import a échoué et que le module ne sera pas chargé.

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MY_MODULE_DIRECTORY] Failed to import the directory /opt/shinken/modules
/MY_MODULE_DIRECTORY as a python module. Python code won't be loaded.
```

- Le second log indique que le fichier '__init__.py' dans le dossier du module est manquant (*sans ce fichier, il est impossible d'importer le module en tant que module Python*).

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MY_MODULE_DIRECTORY]  - because of missing file : /opt/shinken/modules
/MY_MODULE_DIRECTORY/__init__.py.
```

- Le troisième log précise quel fichier n'a pas réussi à être importé dans le code du module. Dans notre exemple, c'est le fichier MY_MODULE_DIRECTORY.my_file qui n'a pas été importé.

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MY_MODULE_DIRECTORY] - The line 'import MY_MODULE_DIRECTORY.my_file' The module
won't work in your module.py file.
```

modules-managerINFO

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MY_MODULE_DIRECTORY ] Successfully imported the directory /opt/shinken/modules
/MY_MODULE_DIRECTORY/ as a python module.
```

Vérification de l'existense du dictionnaire Python "properties" du module

Une fois que le "modules-manager" a chargé un module en tant que package Python, il va vérifier si le dictionnaire "properties" est existant dans le module.

Dans le cas contraire, le module ne sera pas importé et le log en **ERROR** suivant sera affiché :

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CODE-LOADING ] [
directory=MY_MODULE_DIRECTORY ] Missing properties dict in module file /opt/shinken/modules
/MY_MODULE_DIRECTORY/module.py. The module won't be loaded.
```

Création du module - chapitre [CREATION]

Une fois, le code python chargé, on demande au "modules-manager" de créer les modules, de les démarrer et de démarrer son ou ces Worker(s).

Création de l'instance du module

Le "modules-manager" va essayer de créer le module :

- Si le code du module ne contient pas la fonction "get_instance()" ou qu'elle ne retourne rien, un log en **ERROR** sera affiché:

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [
MODULE_NAME] [ module-type=MODULE_TYPE ] The module get_instance() call did not return any instance
or does not exist.
```

- Si le "get_instance()" du module rencontre une erreur, le log en **ERROR** sera affiché avec l'exception Python concernée:

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [
MODULE_NAME ] [ module-type=MODULE_TYPE ]  [0.046s] The module creation has failed raising exception:
EXCEPTION. Will retry later
```

- Si le module type ne correspond pas au démon ou au module qui le lance:

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [ MODULE_NAME ] [ module-type=MODULE_TYPE ] The module MODULE_NAME of type MODULE_TYPE is not available for the daemon/module DAEMON_OR_MODULE_NAME.
```

- Si le module a bien été créé alors un log en **INFO** indiquera son temps de création:

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [ MODULE_NAME ] [ module-type=MODULE_TYPE ]  [0.046s] The module is created.
```

Démarrage du module

Une fois le module créé, le "modules-manager" va essayer de le démarrer.

Ce log sera en **INFO**.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [ MODULE_NAME ] [ module-type=MODULE_TYPE ]  Trying to init module.
```

Si le "modules-manager" a réussi à créer son module, mais n'a pas réussi à le démarrer alors 3 logs apparaîtront :

1. Un log en **ERROR** avec l'exception Python rencontrée :

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [ MODULE_NAME ] [ module-type=MODULE_TYPE ] The module has failed to init raising exception: EXCEPTION.
```

2. Un log en **WARNING** indiquant le nombre de fois que le "modules-manager" à essayé de démarrer son module sans succès. Le compteur se remet à zéro dès lors que le démon a réussi à démarrer son module.

```
[YYYY-MM-DD HH:MM:SS] WARNING : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [ MODULE_NAME ] [ module-type=MODULE_TYPE ] Tried to start the module 5 times.
```

3. Un log en **WARNING** indiquant quand le "modules-manager" va essayer de redémarrer son module (*maximum une minute*).

```
[YYYY-MM-DD HH:MM:SS] WARNING : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [ MODULE_NAME ] [ module-type=MODULE_TYPE ] Will retry to start the module at 16:32:18.
```

Un log en **INFO** apparaîtra si le "modules-manager a réussi à démarrer son module.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [ MODULE_NAME ] [ module-type=MODULE_TYPE ] The module is started.
```

Démarrage des modules avec des workers

Une fois le module créé, le "modules-manager" va essayer de le démarrer.

- S'il n'arrive pas à les démarrer, le module redémarrera dans le futur (*max 1 minute*) afin de redémarrer le module et ces Workers.
- Ce log sera en **INFO**.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [ MODULE_NAME ] [ module-type=MODULE_TYPE ] Starting a worker based module.
```

Si le "modules-manager" n'a pas réussi à démarrer le module :

1. Et qu'il a levé une exception lors du démarrage, alors un log en **ERROR** sera affiché disant que le "modules-manager" n'a pas réussi à démarrer le module avec le message de l'exception. Le module redémarrera dans le futur (*max 1 minute*).

   ```
   [YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [
   MODULE_NAME ] [ module-type=MODULE_TYPE ] The worker based module has failed to init raising
   exception: EXCEPTION. Will retry later.
   ```

2. Un log en **ERROR** sera affiché disant que le "modules-manager" n'a pas réussi à démarrer le module et ces Workers, et que le module redémarrera dans le futur (*max 1 minute*).

   ```
   [YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [
   MODULE_NAME ] [ module-type=MODULE_TYPE ] The worker based module failed to init. Will retry later.
   ```

Si le module et ces Workers se sont bien démarrés alors ce log en **INFO** apparaîtra.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CREATION ] [ MODULE_NAME ] [
module-type=MODULE_TYPE ] The worker based module has started.
```

Création d'un nouveau processus [FORK_CLEANUP] / [HTTP_CLIENT]

La création d'un nouveau processus (*fork*) nécessite l'arrêt des requêtes faites via le module Python PyCurl (*utilisé par HTTP_CLIENT de Shinken*), pour éviter tout risque de blocage (*deadlock*) dans le processus créé (sur les fonctions de résolution de nom de la libc, ou la gestion des contexte SSL de la bibliothèque OpenSSL par exemple).

Les logs suivants indiquent

- le temps nécessaire à l'interruption des requêtes faites via PyCurl.
- quelles requêtes ont été interrompues pour être rejouées plus tard.

```
[YYYY-MM-DD HH:MM:SS] INFO   : [ DAEMON_OR_MODULE_NAME ] [ FORK_CLEANUP     ] [ BEFORE ] [ PID ] [
HTTP_CLIENT ] interrupting current PyCurl transfers to allow process creation
[YYYY-MM-DD HH:MM:SS] WARNING: [ DAEMON_OR_MODULE_NAME ] [ HTTP_CLIENT ] PyCurl query [ URL ] has been
interrupted and paused to allow process creation
[YYYY-MM-DD HH:MM:SS] INFO   : [ DAEMON_OR_MODULE_NAME ] [ FORK_CLEANUP     ] [ BEFORE ] [ PID ] [
HTTP_CLIENT ] PyCurl is now idle, after X.XXXs
[YYYY-MM-DD HH:MM:SS] INFO   : [ DAEMON_OR_MODULE_NAME ] [ HTTP_CLIENT ] PyCurl query [ URL ] has been
replayed successfully ( status:XXX )
```

Changement de configuration ou d'état du module - chapitre [UPDATE]

Ce chapitre ne concerne pas le démon Synchronizer ni le démon Arbiter.

Rajout d'un nouveau module

Lorsqu'un démon ou un module reçoit un nouveau module, le "modules-manager" affiche un log en **INFO** expliquant qu'il va lancer le nouveau module.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] [ MODULE_NAME ]
Launched as a NEW module as it was added to configuration.
```

Quand le "module-manager" a fini de créer le ou les module(s) ajouté(s) dans la configuration du module ou du démon, alors un récapitulatif des actions faites sera affiché en log **INFO**.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] Module modifications:
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] - WebUI -> started (
new )
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] - sla -> started ( new
)
```

Suppression d'un module dans un démon

Lorsqu'un module est enlevé de la configuration du démon ou de son module, le "modules-manager" affichera un log en **INFO** informant que l'on supprime le module.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] [ MODULE_NAME ]
Removing module because it has been removed from configuration.
```

Si le module possède des Workers, le "modules-manager" va arrêter tous ses Workers. Un log en **INFO** sera affiché.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] [ MODULE_NAME ]
Stopping all workers.
```

Le module "modules-manager" indique avec un log en **INFO** qu'il va essayer d'arrêter le module.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] [ MODULE_NAME ] Trying
to stop module.
```

- Si l'arrêt du module a échoué en appelant la fonction quit() du module, un log en **ERROR** sera affiché.

  ```
  [YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] [ MODULE_NAME
  ] Stopping module failed. The quit() function has failed with error: EXCEPTION.
  ```

- Si l'arrêt du module a échoué avec une exception Python, un log en **ERROR** sera affiché.

  ```
  [YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] [ MODULE_NAME
  ] Stopping module has failed raising exception: EXCEPTION.
  ```

Lorsque que le "module-manager" a réussi à arrêter son module, il l'indique avec un log en **INFO**.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] [ MODULE_NAME ]
Stopped module successfully.
```

Si le "modules-manager" a essayé de supprimer le module, mais qu'il n'a pas été trouvé dans sa liste de modules en cours d'exécution alors un log en **WARNING** sera affiché.

```
[YYYY-MM-DD HH:MM:SS] WARNING : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] [ MODULE_NAME ]
Trying to remove the module MODULE_NAME but it was not in running modules list: LIST_CURRENT_RUNNING_MODULES
```

Où *LIST_CURRENT_RUNNING_MODULES* est la liste des modules en cours d'exécution.

Quand le "modules-manager" éteint les modules, si un module ne réussit pas à s'éteindre alors un log en **ERROR** sera affiché, suivi de la traceback de l'exception.

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] [ MODULE_NAME ] The
module MODULE_NAME has failed to stop raising exception: EXCEPTION
```

Quand le "module-manager" a fini d'éteindre le ou les module(s) supprimé(s) dans la configuration d'un module ou d'un démon, alors un récapitulatif des actions faites sera affiché en log **INFO**.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] Module modifications:
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] - Livestatus ->
stopped ( removed )
```

Changement de configuration d'un module (et sa relance)

Le démon ou le module a détecté que la configuration de son module en cours d'exécution a changé. Le "modules-manager" indique avec un log **INFO** qu'il va redémarrer le module afin de prendre en compte la nouvelle configuration.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] [ MODULE_NAME ]
Restarting module because its configuration has changed.
```

Quand le "module-manager" a fini de redémarrer le ou les module(s) supprimé(s) dans la configuration du module ou du démon, alors un récapitulatif des actions faites sera affiché en log **INFO**.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] Module modifications:
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] - Livestatus ->
restarted ( configuration change )
```

Le module n'a pas réussi à se mettre à jour

Le démon ou le module n'arrive pas à mettre à jour son module. Le "modules-manager" informe qu'il faut contacter le support avec un log en **ERROR**.

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ UPDATE ] [ MODULE_NAME ]
Cannot update 'MODULE_NAME' module. Please contact your Support.
```

Arrêt du module - chapitre [SHUTDOWN]

Lorsqu'un démon s'éteint, il va d'abord demander au "modules-manager" d'éteindre ses modules.

Le démon éteint ses modules

Ce log en **INFO** indique que le "modules-manager" va commencer à éteindre tous ses modules.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ SHUTDOWN ] Start to shutdown
all modules.
```

Le module s'éteint

Ce log en **INFO** indique que le "modules-manager est en train d'arrêter le module.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ SHUTDOWN ] Stopping module
MODULE_NAME.
```

Si une exception Python a été remontée par le "modules-manager" lors de l'arrêt du module, un log apparaîtra en **ERROR** avec le nom du module et le message de l'exception Python.

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ SHUTDOWN ] Stopping module
failed. The quit() function has failed with error: EXCEPTION.
```

Si le "modules-manager" a essayé de supprimer un module, mais qu'il ne l'a pas trouvé dans sa liste de modules en cours d'exécution alors un log en **WARNING** sera affiché.

```
[YYYY-MM-DD HH:MM:SS] WARNING : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ SHUTDOWN ] Trying to remove
the module MODULE_NAME but it was not in running modules list: LIST_CURRENT_RUNNING_MODULES
```

Le démon a fini d'éteindre ses modules

Le "modules-manager" informe avec un log **INFO** qu'il a terminé d'arrêter tous les modules.

```
[YYYY-MM-DD HH:MM:SS] INFO : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ SHUTDOWN ] Stopping all modules
ended.
```

Arrêt inopiné du module - chapitre [CRASH]

Le module s'est arrêté de façon inattendu

Lorsqu'un module n'est plus en cours d'exécution et qu'il s'est éteint de façon inattendue, le "modules-manager" affichera un log en **ERROR**.

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CRASH ] [ MODULE_NAME ] [
module-type=MODULE_TYPE ] The module MODULE_NAME has gone down unexpectedly!
```

Lorsqu'un ou plusieurs Worker(s) d'un module ne sont plus en cours d'exécution et qu'ils se sont éteint(s) de manière inattendue, le "modules-manager" affichera un log en **ERROR**.

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CRASH ] [ MODULE_NAME ] [
module-type=MODULE_TYPE ] The module MODULE_NAME worker(s) has gone down unexpectedly!
```

Le module a un comportement anormal

Lorsqu'un module externe a une taille de liste d'attente (*de commande interne ou de retour de commande interne*) **plus élevée** que la taille maximale définie dans le fichier de configuration **ini** du démon (*max_queue_size*), un log sera affiché en **ERROR.**

- QUEUE_SIZE = Taille (*en nombre d'éléments*) de la liste d'attente. Éléments possibles : Commande(s) ou retour(s) de commande(s).
- QUEUE_MAX_SIZE = Taille (*en nombre d'éléments*) maximale de la Queue définie dans le fichier **ini** du démon (*paramètre max queue size*).

```
[YYYY-MM-DD HH:MM:SS] ERROR : [ DAEMON_OR_MODULE_NAME ] [ MODULES-MANAGER ] [ CRASH ] [ MODULE_NAME ] [
module-type=MODULE_TYPE ] The external module MODULE_NAME queue size is too high ( QUEUE_SIZE >
QUEUE_MAX_SIZE )!
```