Le Reactionner

Sommaire

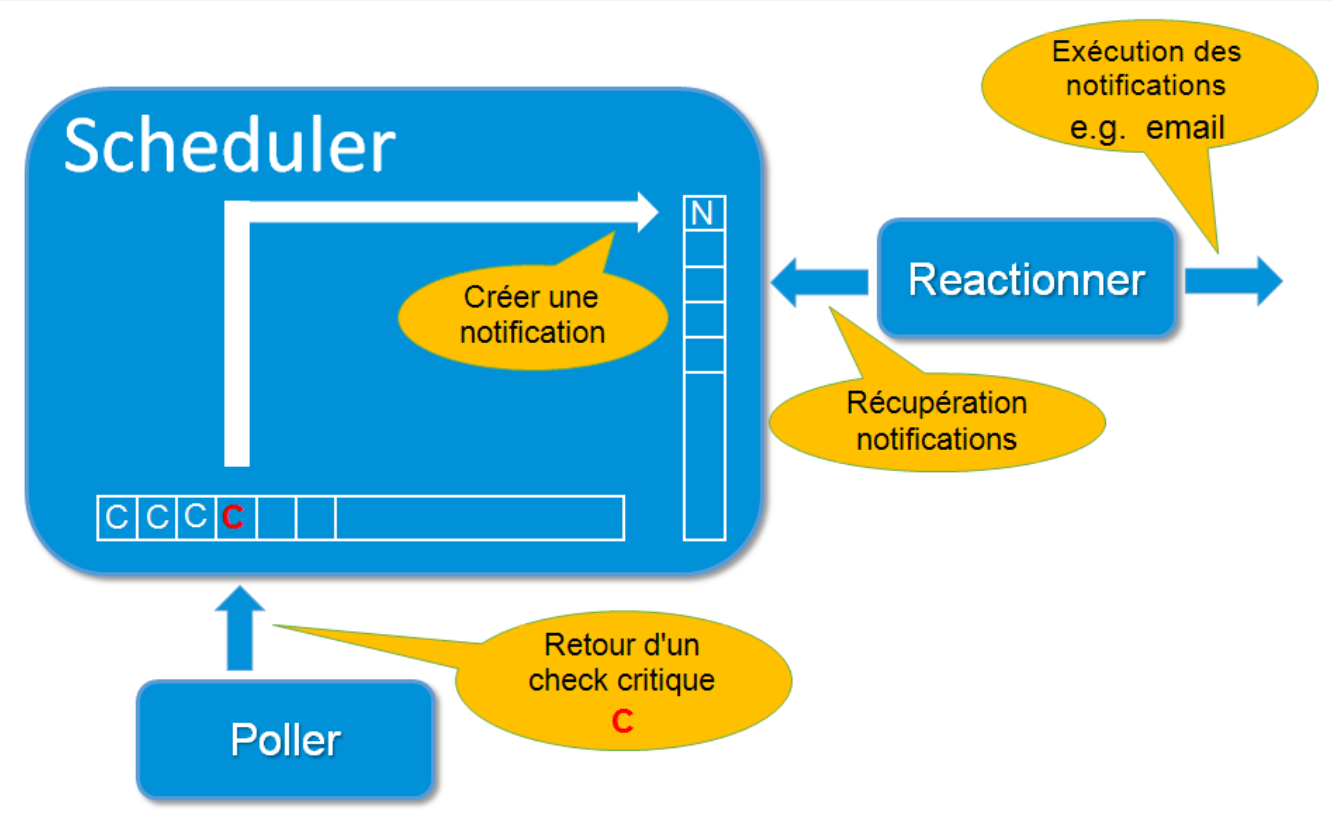
Rôle

Le Reactionner délivre les notifications et lance les gestionnaires d'événements. Cela permet de centraliser les canaux d'échange avec les systèmes externes afin de simplifier les autorisations SMTP ou l'alimentation des sources RSS (seulement un pour tous les hôtes/services). Il peut y avoir plusieurs Reactionners pour du load-balancing (répartition de charge) ou un rôle de spare.

Connexions avec les autres démons

Le Reactionner écoute sur son port 7769, et récupère sa configuration que lui envoie l'Arbiter.

- Sa configuration lui permet de savoir suivant l'architecture, à quel Scheduler ce Reactionner doit se connecter pour récupérer les demandes d'exécution de notification ou de réaction (*Gestionnaire d'événement configuré sur éléments*).
- Il se connectera
  - défacto à tous les Schedulers de son royaume
  - et par défaut aussi à tous les schedulers des sous-royaumes liés au sien (il est possible de désactiver cette connexion au sous-royaume => voir manage_sub_realms)



Visibilité du serveur mail

Le Reactionner lance les commandes de notification afin de notifier les contacts sur les alertes. Les plug-ins de notification sont principalement basés sur le système mail, donc le Reactionner a besoin d'une configuration MTA locale (Mail Transfer Agent). Cette configuration est à la discrétion de l'administrateur système. Par exemple, vous pouvez installer et paramétrer Postfix pour relayer les emails vers un serveur Exchange.

Données

Que ce soit en mode actif ou passif, le Reactionner reçoit les checks (et donc les différentes lignes de commandes à exécuter) des Schedulers.

Il ne connait rien aux objets, il reçoit juste des commandes à exécuter (créées par les Schedulers).

Résumé des connexions du Reactionner

Source	Destination	Port	Protocole	Note
Reactionner	Schedulers	7768	HTTP/HTTPS	

Descriptions des variables

Propriété	Valeur par défaut	Description
reactionner_name	N/A	Cette variable est utilisée pour identifier le nom raccourci du Reactionner auquel les données sont attachées.
address	N/A	Cette directive est utilisée pour définir l'adresse d'où l'Arbiter principal peut joindre le Reactionner. Cela peut être un nom DNS ou une adresse IP.
port	7769	Cette directive est utilisée pour définir le port TCP utilisé par ce démon.
use_ssl	0	Cette variable est utilisée pour définir si le Reactionner doit être contacté en HTTPS (*1*) ou HTTP (*0*). La valeur par défaut est *0* (HTTP).
min_workers	0	Cette variable est utilisée pour définir le nombre de processus Worker utilisé par le Reactionner. Par défaut 0, correspond à 1 Worker par CPU.
processes_by_worker	256	Cette variable est utilisée pour définir le nombre maximal de commandes qu'un Worker est autorisé à exécuter en simultané.
display_statistics_compute_time_if_higher	100	Temps (en ms) de calcul à partir duquel le log affichant le temps de calcul des statistiques passe de DEBUG à INFO.
polling_interval	1	Cette variable est utilisée pour définir le nombre de secondes à attendre avant que le Reactionner ne récupère les checks des Schedulers
timeout	3	Cette variable est utilisée pour définir le temps en secondes avant que l'Arbiter ne considère ce démon comme à l'arrêt. Si ce démon est joignable en HTTPS (use_ssl à 1) avec une latence élevée, nous vous conseillons alors d'augmenter cette valeur de timeout (l'Arbiter aura besoin de plus d'allers/retours pour le contacter).
data_timeout	120	Cette variable est utilisée pour définir le temps en secondes avant de considérer un transfert de configuration ou de données comme échoué.
max_check_attempts	3	Si le ping permettant de détecter la disponibilité réseau du nœud est en échec N fois ou plus, alors le nœud est considéré comme mort. (par défaut, 3 tentatives)
check_interval	60	Intervalle de Ping toutes les N secondes.
modules	N/A	Cette variable est utilisée pour définir tous les modules chargés par le Reactionner.
realm	N/A	Cette variable définie dans quel royaume sera le Reactionner. Si aucun n'est sélectionné, il sera assigné à celui par défaut.
manage_sub_realms	1	Cette variable est utilisée pour définir si le Poller acceptera les tâches du Scheduler venant des sous-royaumes.
reactionner_tags	None	Cette variable est utilisée pour définir les checks que peut lancer le Reactionner. S'il n'y a aucun reactionner_tags spécifié, il prendra tous les checks non qualifiés. Par défaut, il n'y a aucun reactionner_tag, donc il prend tous les checks non qualifiés.
passive	0	Cette variable est utilisée pour définir le mode de connexion. Par défaut 0, le Reactionner se connecte au Scheduler pour récupérer ou retourner les informations. Si 1, le Scheduler se connecte au Reactionner pour pousser ou récupérer les informations.
spare	0	Cette variable est utilisée pour définir si le Reactionner doit être géré en tant que spare (ne chargera la configuration que si le maître tombe). La valeur par défaut est *0*.
enabled	1	Cette variable est utilisée pour définir si le Reactionner est activé ou non.

Exemple de définition

Dans le répertoire **/etc/shinken/reactionners/**, voici un exemple de définition qui permet la définition du Reactionner (à placer dans un fichier CFG) :

⚠️ Il est conseillé d'éditer les fichiers .cfg avec l'encodage utf-8

```
#================================================================================
# REACTIONNER
#================================================================================
# Description: The reactionner is responsible for:
# - Executing notification actions
# - Executing event handler actions
#================================================================================

define reactionner {


    #======== Daemon name and address =========
    # Daemon name. Must be unique
    reactionner_name          reactionner-master

    # IP/fqdn of this daemon (note: you MUST change it by the real ip/fqdn of this server)
    address                   localhost

    # Port (HTTP/HTTPS) exposed by this daemon
    port                      7769

    # 0 = use HTTP, 1 = use HTTPS
    use_ssl                           0


    #======== Daemon performance sizing =========
    # Number of worker process to launch. 0 = Auto -> 1 per CPU
    min_workers               0

    # Number of maximum commands a worker is allowed to launch in the same time
    processes_by_worker       256

    # How many seconds to wait between the poller to get jobs from the scheduler
    polling_interval          1


    #======== Daemon connection timeout and down state limit =========
    # timeout: how many seconds to consider a node don't answer
    timeout                   3

    # data_timeout: how many second to consider a configuration transfert to be failed
    # because the network brandwith is too small.
    data_timeout              120

    # max_check_attempts: how many fail check to consider this daemon as DEAD
    max_check_attempts        3

    # Check this daemon every X seconds
    check_interval            60

    # Time (in ms): if the time to compute stats (for shinken checks) is higher
    #               than this time, then the log will be in INFO level
    # Default: 100  (ms)
    # display_statistics_compute_time_if_higher    100

    #======== Modules to enable for this daemon =========
    # Available:  None for this daemon
    modules


    #======== Realm and architecture settings =========
    # Realm to set this daemon into
    realm                     All

    # manage_sub_realms 1 = take data from the daemon realm and its sub realms
    #                   0 = (default) take data only from the daemon realm
    # NOTE: this can be set to 1 on top level realms to allow the reactionner
    #       to take jobs (notifications & event handlers) from schedulers on lower
```

```
    #       realms too.
    #       But please note that if there is a reactionner in the lower realms, then
    #       BOTH reactionners will take jobs from the scheduler.
    manage_sub_realms        0

    # Reactionner tags are managed tags. The hosts, checks and commands can have specific tags, and
    # this reactionner will only take them if the element tag is list in the reactionner
    # Reserved tag name:  None  =  untaggued element (default value)
    #reactionner_tags       None

    # passive:  0 = reactionner will connect to the scheduler to take/returs jobs
    #           1 = scheduler will connect to this reactionner to push/take back jobs
    passive                  0

    # spare: 0 = this reactionner is always active
    #        1 = this reactionner will be activated by the arbiter is another reactionner is dead
    spare                    0


    #======== Enable or not this daemon =========
    # 1 = is enabled, 0 = is disabled
    enabled                  1

}
```