

# L'Arbiter

## Rôle

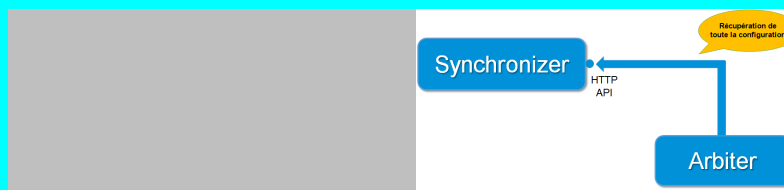
Le démon Arbiter lit la configuration du Synchronizer. Il la divise en N parts ( $N \text{ Schedulers} = N \text{ parts}$ ), et les distribue au démon Shinken Enterprise approprié de manière parallèle. Il gère également la notion de haute disponibilité : si un démon en particulier meurt, il renvoie la configuration gérée par le démon mort à son spare défini qui reçoit les informations des utilisateurs. Il route également les résultats de checks passifs vers le démon approprié. Les résultats de checks passifs sont renvoyés au Scheduler responsable du check.

### **i** Important

Il ne peut y avoir qu'un seul Arbiter master et un seul Arbiter spare maximum par infrastructure.

## Connexion avec le Synchronizer

La communication entre l'Arbiter et le Synchronizer est faite sur le port standard du Synchronizer (7765).



## Connexion avec les autres démons

Ce démon est utilisé pour vérifier et distribuer la configuration aux autres démons, sauf au Synchronizer. Il se connecte sur le port standard des autres démons, et utilise une connexion HTTP ou HTTPS si ceux-ci l'ont prévue.



### **i** Remarque

L'Arbiter est capable de rendre inactif les démons distants désactivés (lorsque la propriété "enabled" est mise 0 dans leurs fichiers cfg)

Exemple: Depuis le serveur central, je décide de désactiver un Poller distant en passant la propriété "enabled" à 0 dans son fichier de définition. Je redémarre l'Arbiter. Le Poller distant sera affiché comme "désactivé" depuis son shinken-healthcheck local.

Il n'est donc pas nécessaire d'éteindre un démon qui n'est plus utilisé sur un machine distante.

## Données

Ce démon stocke la totalité du système de configuration en mémoire. Il a accès à tous les noms de serveurs, adresses, types et commande définis pour les vérifier.

Il stocke également en mémoire les contacts définis, qui doivent recevoir les notifications pour les hôtes et services définis.

## Résumé des connexions de l'Arbiter

Démon source	Destination	Port	Protocole	Note
Arbiter	Synchronizer	7765	HTTP/HTTPS	
Arbiter	Scheduler	7768	HTTP/HTTPS	
Arbiter	Poller	7771	HTTP/HTTPS	
Arbiter	Reactionner	7769	HTTP/HTTPS	
Arbiter	Receiver	7773	HTTP/HTTPS	
Arbiter	Arbiter	7770	HTTP/HTTPS	seulement si il y a un Arbiter esclave, et seulement du maître vers l'esclave.
Arbiter	Broker	7772	HTTP/HTTPS	

## Définition du format

Propriété	Défaut	Description
arbiter_name	N/A	Cette variable est utilisée pour identifier le *nom réduit* de l'Arbiter auquel les données sont associées.
host_name	N/A	Cette variable est utilisée par les démons Arbiters pour définir quel objet 'Arbiter' ils sont (de type Maître ou Spare). En effet, tous ces démons sur différents serveurs utilisent la même configuration, donc la seule différence entre eux est le nom du serveur. Cette valeur doit être égale au nom du serveur (comme avec la commande du hostname). Si rien n'est défini pour cette propriété, le démon Arbiter va utiliser le nom du serveur où il est lancé, mais cela ne sera possible qu'avec un seul Arbiter dans l'architecture Shinken.
address	N/A	Cette directive permet de définir l'adresse d'où l'Arbiter principal peut contacter cet Arbiter (qui peut être lui même). Ça peut être un nom DNS ou une adresse IP.
port	7770	Cette directive est utilisée pour définir le port TCP utilisé par ce démon.
use_ssl	0	Cette variable est utilisée pour définir si l'Arbiter doit être contacté en HTTPS (*1*) ou HTTP (*0*). La valeur par défaut est *0* (HTTP).
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'aller/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.
spare	0	Cette variable permet de savoir si le démon correspondant à la définition de l'Arbiter est un spare ou pas. La valeur par défaut est *0* (maître/non-spare).
modules	N/A	Cette variable définit tous les modules chargés par l'Arbiter qui correspond à cette définition.
enabled	N/A	Cette variable est utilisée pour définir si l'Arbiter est activé ou non.

## Exemple de définition

Dans le répertoire `/etc/shinken/arbiter/`, voici un exemple de définition qui permet la définition de l'Arbiter (à placer dans un fichier `.cfg`) :

```
#####
# ARBITER
#####
# Description: The Arbiter is responsible for:
# - Loading, manipulating and dispatching the configuration
# - Validating the health of all other Shinken daemons

#####
# IMPORTANT: If you use a spare arbiter you MUST set the host_name on each
# servers to its real DNS name ('hostname' command).
#####

define arbiter {

    #=====  

    # Daemon name and address =====  

    # Daemon name. Must be unique  

    arbiter_name      Main-arbiter

    # hostname used by the arbiter daemon to know which arbiter definition it must use as  

    # the same configuration with two arbiter definition is set on both master and slave  

    # arbiter  

    host_name         nodel

    # IP/fqdn of this daemon  

    address           nodel.mydomain ; DNS name or IP

    # Port (HTTP/HTTPS) exposed by this daemon  

    port              7770

    # 0 = use HTTP, 1 = use HTTPS  

    use_ssl           0

    #=====  

    # Daemon connection timeout and down state limit =====  

    # The arbiter connection timeout and down state limit are useful for another arbiter to  

    # know when to consider this arbiter as DEAD so the spare can take the lead.  

    # timeout: how many second 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 bandwidth 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

    #=====  

    # Master or spare selection =====  

    # 1 = is a spare, 0 = is not a spare  

    spare             0

    #=====  

    # Modules to enable for this daemon =====  

    # Available:  

    # - synchronizer-import : [mandatory] will allow to get configuration from the synchronizer  

    modules           synchronizer-import

    #=====  

    # Enable or not this daemon =====  

    # 1 = is enabled, 0 = is disabled  

    enabled           1
}
```

## Remarques sur la compilation et l'envoi de la configuration

Un des rôles de l'Arbiter est de récupérer la configuration des démons, hôtes et checks et de la transmettre aux démons appropriés.

Lorsque la configuration comporte un nombre important d'hôtes et de checks, la compilation de la configuration peut être longue. De même, si on dispose d'une architecture comportant plusieurs royaumes, il est possible que le démon de certains royaumes soient situés dans des zones géographiques ou configurations réseau où les débits et latences de connexion sont élevés. Ces connexions réseaux moins optimisées peuvent occasionner des ralentissements sur l'ensemble des royaumes lors de l'envoi de la configuration des hôtes/checks vers les démons.

> V02.05.00 => En *Série*

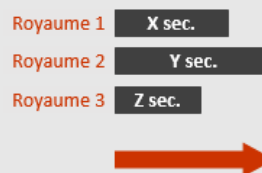
( Envoie de  $X + Y + Z$  )



> Temps: **SOMME ( X, Y, Z )** secondes

> V02.06.00 => En *Parallèle*

( Envoie de  $X + Y + Z$  )



> Temps: **MAX ( X, Y, Z )** secondes

Pour limiter l'impact de ces problèmes, depuis la V02.06.00, l'Arbiter effectue la compilation et l'envoi de la configuration de manière parallèle pour chaque royaume. Un ralentissement occasionné par une connexion ralentie vers un royaume n'aura alors pas d'effet sur l'envoi de la configuration aux autres royaumes de l'architecture Shinken.