# Module named-pipe ( mise en place )

**Sommaire**



Le module Named Pipe, via le fichier "passe plat" FIFO (First In First Out) **shinken.cmd**, va permettre à Shinken de récupérer les entrées ou commandes externes.

## Mise en place du module

- Il faut rajouter la définition du module. Si elle n'est pas présente:
  - Dans le répertoire /etc/shinken/modules/
  - Copier le fichier named-pipe.cfg et modifier les droits :

```
chown shinken:shinken named-pipe.cfg
chmod 644 named-pipe.cfg
```

- Ensuite on va accrocher le module au receiver
  - Ouvrir le fichier /etc/shinken/receivers/receiver-master.cfg
  - Ajouter named-pipe à la ligne modules
    - *modules named-pipe*
- Redémarrer Shinken via la commande *service shinken restart*

### Création de l'élément de supervision en configuration

- Dans l'interface de configuration, il faut à présent créer le check à associer à un hôte, en passif, non actif, et volatile, car si nous souhaitons recevoir une notification dès un changement d'état. L'expiration de l'état de fraîcheur doit également être paramétrée.
  - Voici un exemple avec la syntaxe via fichiers cfg, modèle de check dédié au modèle d'hôte "TRAP-modele" :

```
define service{
        service_description     TRAP
        check_command           check-host-alive
        host_name               TRAP-modele
                is_volatile             1
        passive_checks_enabled  1
                active_checks_enabled          0
                check_freshness                       1
                freshness_threshold               300
        register                0
        check_interval              1
        retry_interval              1
}

define host{
        name                                    TRAP-modele
        register                        0
}
```

- Appliquer le modèle d'hôte à un hôte accessible sur le réseau (le paramètre "adresse" doit être rempli), par exemple un hôte "test-trap"

**Rappels sur le fonctionnement des checks passifs**

La configuration de check présentée ci-dessus est celle d'un check passif. Le statut d'un check de ce type va être mis à jour manuellement via la réception de traps SNMP.

Si aucun statut n'est reçu de manière passive pour ce check au bout de 5mn, Shinken déclenche une vérification manuelle pour éviter d'avoir un statut trop ancien et non représentatif de l'état de l'élément représenté par le check. Ce comportement est configuré via les options "check_freshness" (pour l'activation de ce comportement), "check_command" (pour la commande de vérification lancée par Shinken) et "freshness_threshold" (en secondes, 300 pour 5mn).

**Par contre, ce comportement peut entraîner des changements de statuts du check si l'intervalle d'envoi des statuts vers le check (traps SNMP dans notre exemple) est supérieur à l'intervalle définit par l'option "freshness_threshold".** Il faut donc avoir conscience de ce comportement et régler la valeur de l'option "freshness_threshold" ou bien désactiver cette fonctionnalité pour ce check ("check_freshness 0").

# Script interpréteur des traps

## Pour un hôte

Ajouter le script suivant que l'on appellera **submit_host_result_to_receiver** dans le dossier des plugins Shinken ( **/var/lib/shinken-user/libexec/** ):

```
#!/bin/bash
# Arguments:
# ${1} = host_name (Short name of host that the service is associated with)
# ${3} = return_code (An integer that determines the state of the service check, 0=OK, 1=WARNING, 2=CRITICAL,
3=UNKNOWN).
# ${4} = plugin_output (A text string that should be used as the plugin output for the service check)

# Ensuring we use the correct commands by using their full absolute path
echocmd="/bin/echo"
commandfile="/var/lib/shinken/shinken.cmd"

# get the current date/time in seconds since UNIX epoch
datetime="$(date +%s)"

# create the command line to add to the command file
cmdline="[${datetime}] PROCESS_HOST_CHECK_RESULT;${1};${2};${3}"

# append the command to the end of the command file
${echocmd} "${cmdline}" >> "${commandfile}"
```

On le rend exécutable et on le donne à l'utilisateur Shinken

```
chown shinken:shinken /var/lib/shinken-user/libexec/submit_host_result_to_receiver
chmod +x /var/lib/shinken-user/libexec/submit_host_result_to_receiver
```

Pour tester le script et simuler une réception d'un trap translaté au format Shinken, il suffit d'exécuter la commande suivante qui va faire passer l'hôte en état critique :

```
/var/lib/shinken-user/libexec/submit_host_result_to_receiver HÔTE 2 "test envoi trap - CRITIQUE"
```

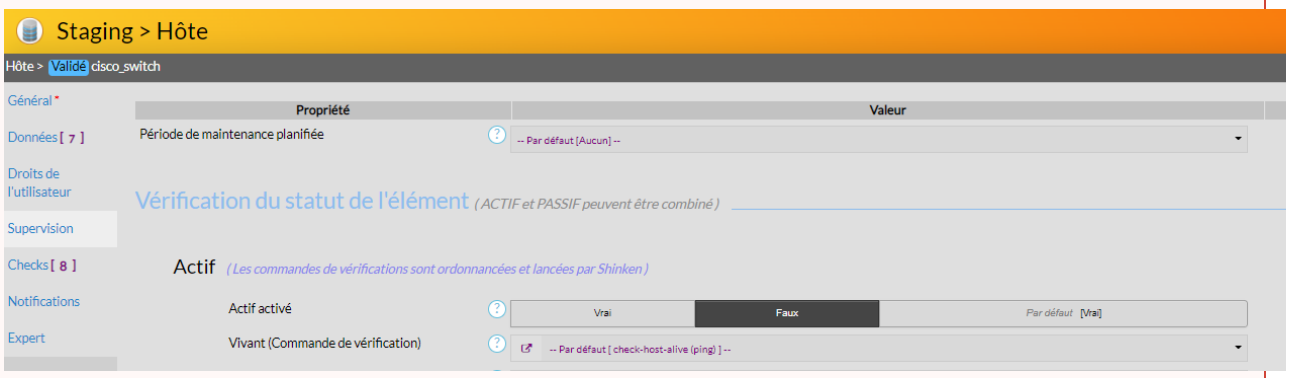
Les arguments sont :

- $1 = Le nom de la machine concerné par la trap
- $2 = Le code de retour ( 0=OK, 1=WARNING, 2=CRITICAL, 3=UNKNOWN )
- $3 = Un message texte correspondant à la sortie de la commande.

L'hôte devrait passer en critique, et si au bout de la période du seuil de fraîcheur, aucun nouveau trap n'a été reçu, alors la commande check-host-alive fera repasser le check à OK ( si bien sûr l'hôte est accessible via le réseau ).

**Attention**

Si vous ne voyez pas votre hôte passer en critique, il est possible que vous n'ayez pas désactivé la vérification active sur votre hôte. Pour modifier cela, rendez-vous dans l'interface de configuration, cliqué sur votre hôte et dans supervision, mettez à Faux l'actif activé comme sur l'image ci-dessous :




## Pour un check

Ajouter le script suivant que l'on appellera **submit_check_result_to_receiver** dans le dossier des plugins Shinken ( **/var/lib/shinken-user/libexec/** ):

```
#!/bin/bash
# Arguments:
# ${1} = host_name (Short name of host that the service is associated with)
# ${2} = svc_description (Description of the service)
# ${3} = return_code (An integer that determines the state of the service check, 0=OK, 1=WARNING, 2=CRITICAL,
3=UNKNOWN).
# ${4} = plugin_output (A text string that should be used as the plugin output for the service check)

# Ensuring we use the correct commands by using their full absolute path
echocmd="/bin/echo"
commandfile="/var/lib/shinken/shinken.cmd"

# get the current date/time in seconds since UNIX epoch
datetime="$(date +%s)"

# create the command line to add to the command file
cmdline="[${datetime}] PROCESS_SERVICE_CHECK_RESULT;${1};${2};${3};${4}"

# append the command to the end of the command file
${echocmd} "${cmdline}" >> "${commandfile}"
```

On le rend exécutable et on le donne à l'utilisateur Shinken

```
chown shinken:shinken /var/lib/shinken-user/libexec/submit_check_result_to_receiver
chmod +x /var/lib/shinken-user/libexec/submit_check_result_to_receiver
```

Pour tester le script et simuler une réception d'un trap translaté au format Shinken, il suffit d'exécuter la commande suivante qui va faire passer le service en état critique :

```
/var/lib/shinken-user/libexec/submit_check_result_to_receiver HÔTE CHECK 2 "test envoi trap - CRITIQUE"
```

Les arguments sont :

- $1 = Le nom de la machine concerné par la trap
- $2 = Le nom du check ( doit correspondre au nom donnée dans la définition du check Shinken. dans cet exemple : TRAP )
- $3 = Le code de retour ( 0=OK, 1=WARNING, 2=CRITICAL, 3=UNKNOWN )
- $4 = Un message texte correspondant à la sortie de la commande.

Le check devrait passer en critique, et si au bout de la période du seuil de fraîcheur, aucun nouveau trap n'a été reçu, alors la commande check-host-alive fera repasser le check à OK ( si bien sûr l'hôte est accessible via le réseau ).

**Attention**

Si vous ne voyez pas votre check passer en critique, il est possible que vous n'ayez pas désactivé votre check en mode actif. Pour modifier cela, rendez-vous dans l'interface de configuration, cliqué sur votre check et dans supervision, mettez à Faux l'actif activé comme sur l'image ci-dessous :