Module ws-arbiter (mise en place)

Sommaire

Mise en place du module

- Il faut vérifier que la définition du module est bien présente:
  - Dans le répertoire /etc/shinken/modules/
  - ouvrir le fichier ws-arbiter :
  - Vérifier que les paramètres correspondent à vos attentes:
    - Adresse de connexion
    - Activation du SSL
    - Authentification HTTP

```
#===============================================================================
# ws-arbiter   (webservice)
#===============================================================================
# Daemons that can load this module:
# - receiver
# - arbiter
# This module is a webservice that can be used to send checks to Shinken Enterprise
# as POST HTTP(s)
#===============================================================================


define module {

    #======== Module identity =========
    # Module name. Must be unique
    module_name               ws-arbiter
    # Module type (to load module code). Do not edit.
    module_type               ws_arbiter


    #======== Listening address =========
    # host: IP address to listen to.
    #       note: 0.0.0.0 = all interfaces.
    host                      0.0.0.0

    # port to listen
    port                      7760

    # HTTPs part, enable if you want to set the listening for HTTPS instead of default HTTP.
    # disabled by default. Set your own certificates.
    use_ssl                   0
    ssl_cert                  /etc/shinken/certs/server.cert
    ssl_key                   /etc/shinken/certs/server.key


    #======== HTTP authentification =========
    # You can use HTTP basic authentification method for this module.
    # If username is set to anonymous and password is commented, then
    # no authentification will be required.
    username                 shinken
    password                shinken
}
```

- Ensuite on va accrocher le module au receiver

        - *modules ws-arbiter*
    - Ouvrir le fichier /etc/shinken/receivers/receiver-master.cfg
    - Ajouter le nom du module (ws-arbiter) à la ligne modules
- Redémarrer l'Arbiter via la commande *service shinken-arbiter restart* (pour qu'il propage cette nouvelle configuration)

Exemple de l'hôte et du check qui vont permettre de vérifier la bonne réception des traps

- 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 "TRAP-modele" 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

# get the current date/time in seconds since UNIX epoch
datetime=`date +%s`

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

```
# Beware to update user/password and shinken-srv address
curl -u user:password -X POST -d "time_stamp=$datetime&host_name=$1&return_code=$2&output=$3" http://shinken-srv:7760/push_check_result
```

Important

Penser à modifier les valeurs suivantes dans le script :

- user et password : user et password depuis la configuration de votre module receiver-module-webservice
- shinken-srv : adresse de votre serveur receiver où se situe le module receiver-module-webservice

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"
```

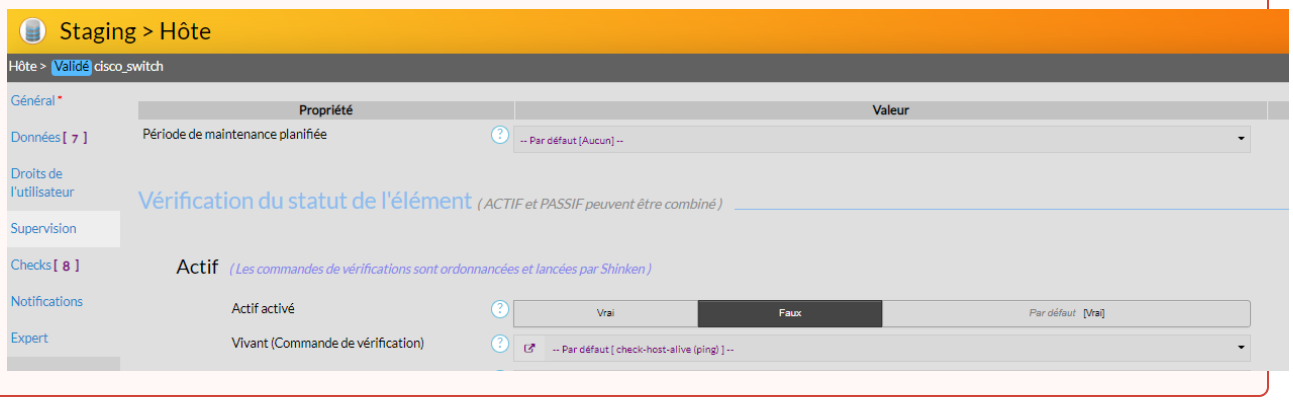
Les arguments sont:

- $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

# get the current date/time in seconds since UNIX epoch
```

```
datetime=`date +%s`

# 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)

# Beware to update user/password and shinken-srv address
curl -u user:password -X POST -d
"time_stamp=$datetime&host_name=$1&service_description=$2&return_code=$3&output=$4" http://shinken-srv:7760
/push_check_result
```

Important

Penser à modifier les valeurs suivantes dans le script :

- user et password : user et password depuis la configuration de votre module receiver-module-webservice
- shinken-srv : adresse de votre serveur Receiver où se situe le module receiver-module-webservice

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 la vérification active sur votre check. 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 :