

Importer ses propres fichiers CFG

Sommaire

Contexte

Ce qui est déjà disponible dans l'installation de Shinken

Utiliser la source pré-configurée `cfg-file-nagios`

Créer et utiliser sa propre source et module

Contexte

Création d'une source

Création du répertoire de données de la source

Configuration du fichier de définition des données de la source

Configuration du démon Synchronizer pour déclarer la source

Affichage de la source dans l'UI de configuration

Import des objets

Données globales

Précisions techniques

Clés de synchronisation

Propriétés par défaut utilisé pour la construction des clés de synchronisation

Contexte

Ce document décrit comment importer ses propres fichiers CFG à travers sa propre source.



Important

Attention, nous vous conseillons de créer votre propre source. En effet, la source Nagios (et Shinken) sont des sources qui sont placées dans le répertoire central Shinken (`/etc/shinken`) et peuvent être soumises à des modifications lors de mises à jour. Par contre, la création de nouvelles sources personnalisées se fera dans le répertoire dédié aux utilisateurs (`/etc/shinken-user`) et ce répertoire sera donc traité avec grande vigilance par le moteur de mise à jour, car incluant vos données utilisateur.

Voici les étapes à suivre afin de créer une source qui vous permettra d'importer vos propres CFG:

- [Création d'une source](#)
- [Création du répertoire de données de la source](#)
- [Configuration du démon Synchronizer pour déclarer la source](#)

Ce qui est déjà disponible dans l'installation de Shinken

Pour vous rendre la vie plus facile, quelques taches de configuration ont déjà été faites:

- a. Le type de module "`cfg-file-import`" est déjà prêt à être utilisé,
- b. La possibilité d'utiliser la source pré-configurée "`cfg-file-nagios`" pour tester rapidement un import de cfg.

Utiliser la source pré-configurée `cfg-file-nagios`

Attention, encore une fois, pour une mise en production sur le long terme, il est préférable que vous utilisiez votre propre source, passez alors sur le chapitre suivant.

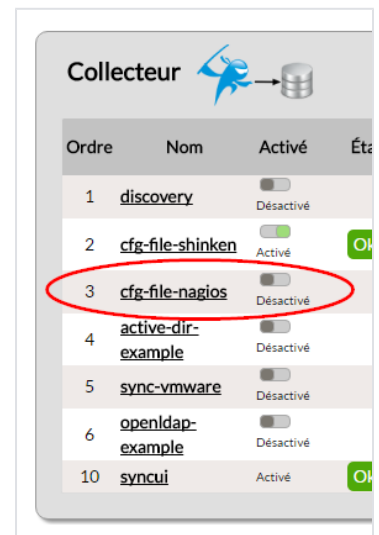
Pour un essai d'import de fichier cfg, l'installation ou la mise à jour de Shinken va mettre en place la source `cfg-file-nagios` déjà configurée :

- `cfg-file-nagios`. Vous pouvez la voir dans la page des sources.

Cette source utilise 2 types de configuration de fichiers

- Les fichiers de définition de la source et de ses modules : `/etc/shinken/sources/cfg-file-nagios.cfg`
- Les fichiers de configuration pour l'import de vos éléments via vos fichiers cfg : `/etc/shinken-user/source-data/source-data-cfg-nagios/definition-source-data-cfg-nagios.cfg`
 - Ce fichier est pré-configuré pour pointer vers votre dossier de configuration nagios. Il vous suffit de saisir le chemin correspondant pour importer facilement et rapidement votre configuration nagios.

Il vous suffira alors d'activer la source pour importer vos éléments définis dans le fichier `nagios.cfg`.



Créer et utiliser sa propre source et module

Contexte

Le fait d'avoir plusieurs sources distinctes peut vous aider si vous avez un grand nombre de fichiers `cfg` et que vous souhaitez pouvoir choisir et séparer les imports. Par exemple, vous avez des fichiers `cfg` relatifs à l'organisation de Paris, et d'autres à l'organisation de Bordeaux (Hôtes, Utilisateurs, ..).

À un instant donné, vous pouvez avoir besoin d'importer les fichiers de Bordeaux uniquement. Ce sera chose possible avec les deux sources distinctes.

Ce chapitre va vous permettre de configurer votre propre source depuis la source exemple : `cfg-file-sample`



A chaque fois que vous avez besoin de personnaliser Shinken, le répertoire à utiliser est `/etc/shinken-user/`

Création d'une source

Placez-vous à présent dans le répertoire des sources Shinken (dans `/etc/shinken/sources`) et copiez la source `Sample` existante (qui a le format d'une source standard sur laquelle vous pouvez vous baser pour créer une source personnalisée) :

```
cd /etc/shinken/sources/  
cp cfg-file-sample.cfg cfg-file-ma-source.cfg
```



Modifiez le fichier `cfg-file-ma-source.cfg`

Supprimez les 4 lignes allant de

Shinken Enterprise

à

End of Shinken Enterprise part

Modifiez

```
source_name      cfg-file-sample  
cfg_path         /etc/shinken-user/source-data/source-data-cfg-sample/definition-source-data-cfg-  
sample.cfg
```

avec

```
source_name      cfg-file-ma-source  
cfg_path         /etc/shinken-user/source-data/source-data-cfg-ma-source/definition-source-data-cfg-  
ma-source.cfg
```

(ce chemin sera celui du répertoire de vos fichiers `cfg` dans le `shinken-user`)

En ce qui concerne les paramètres des sources, voici leurs descriptions :

Propriété	Exemple	Description
-----------	---------	-------------

source_name	cfg-file-ma-source	Nom de la source affichée dans l'UI de configuration en page d'accueil. Doit être unique et d'une longueur inférieure à 40 caractères, sans quoi le Synchronizer ne démarrera pas.  Contrainte sur le nom (qui empêche le Synchronizer de démarrer) Ce nom ne doit pas dépasser 40 caractères. Caractère interdit " ; , / \ & ? , # , ; et \$ Ce paramètre est obligatoire
order	2	L'ordre de la source sur la page, qui a un impact dans la fusion des données lors des imports de sources. Voir la page du Synchronizer pour plus d'information au sujet des fusions. Cette propriété est éditée directement si vous changez l'ordre depuis l'interface.
import_interval	0	La source sera exécutée automatiquement toutes les <i>import_interval</i> minutes. Si 0, la source ne sera jamais exécutée automatiquement.  Conseil Pour vos tests initiaux, nous vous conseillons de positionner la valeur à 0. Le bouton Force import de la page principale vous permettra de faire l'import manuellement. Activé État Prochain import Forcer l'import ☒ Activé Ok Dans 2 min 
module-type	cfg-file-import	Définition du type de module utilisé par la source.
enabled	1	Activer ou désactiver la source. Cette propriété est éditée directement si vous utilisez le bouton Activer de l'interface.
description	Cette source permet de charger mes fichiers cfg relatifs aux nouvelles commandes	Description de la source
not_stored_properties	< liste des propriétés >	Ce paramètre permet de définir un ou plusieurs propriétés que ne seront pas importés dans shinken. Cela peut être utile pour exclure une propriété ou bien utiliser des propriétés personnalisées utiles pour la gestion de vos fichiers .cfg
properties_used_as_synckey	address	Définit la liste de propriétés qui seront utilisées en plus du nom et du SE_UUID de l'élément pour générer les clés de synchronisation (sync_key). Ce paramètre est optionnel. Si ce paramètre n'est pas présent, sa valeur par défaut vaut propriété "address". S'il est défini à vide, la propriété "address" ne sera pas utilisé comme synckey.
update_cfg_with_staging_se_uuid	1	Ce paramètre permet de vérifier, avant l'import, si un élément (quelque soit son type) existe déjà en Staging. Ce sont les clés de synchronisation (sync_keys) qui sont utilisées pour déterminer si l'élément existe en Staging. Si c'est le cas et que cet élément ne dispose pas de SE_UUID dans la source, alors le fichier sera édité afin d'insérer le SE_UUID de l'élément en Staging. Cela permet de s'assurer que l'élément soit toujours calculé correctement, même si son nom ou son adresse change. <i>Note : pour cela il faut que l'utilisateur Shinken est les droits d'écriture sur les fichiers .cfg.</i>

create_and_write_SEUID_in_file	0	Ce paramètre permet de créer et d'écrire un SE_UUID dans le fichier de la source, avant l'import, sur tous les éléments qui ne possèdent pas de SE_UUID, quelque soit leur type. Si l'élément existe déjà dans l'interface de configuration alors il pourra avoir conflit, il faut donc activer ce paramètre que si vous êtes sûr que les éléments n'existent pas déjà dans l'interface. <i>Note : pour cela il faut que l'utilisateur Shinken est les droits d'écriture sur les fichiers .cfg.</i>
--------------------------------	---	--



Si le paramètre **create_and_write_SEUID_in_file** est utilisé, le SE_UUID sera créé même si un élément correspondant existe déjà dans staging. Le paramètre **update_cfg_with_staging_se_uuid** sera donc inutile.

Création du répertoire de données de la source

Pour créer son propre répertoire dans le répertoire des sources de shinken-user :

```
cd /etc/shinken-user/source-data
cp -r source-data-cfg-sample/ source-data-cfg-ma-source/
mv source-data-cfg-ma-source/definition-source-data-cfg-sample.cfg source-data-cfg-ma-source/definition-source-data-cfg-ma-source.cfg
```

Rajouter les droits sur le répertoire :

```
cd /etc/shinken-user/source-data
chown -R shinken:shinken source-data-cfg-ma-source/
```



Astuce

Une fois le répertoire créé, les fichiers cfg pourront alors être placés directement dans le répertoire source-data-cfg-ma-source/elements ou source-data-cfg-ma-source/packs
Dans tous les cas, tous les fichiers cfg dans le répertoire et sous répertoires de source-data-cfg-ma-source seront traités.

Configuration du fichier de définition des données de la source

Le fichier de définition des données de la source est le fichier qui est désigné par le **cfg_path** dans la configuration de la source. Ce fichier doit être unique, mais il permet de définir plusieurs emplacements à importer.

Il présente trois paramètres :

paramètre	Exemple	Description
cfg_dir	cfg_dir=.	Il permet d'indiquer à Shinken de lire le contenu d'un dossier et de ses sous-dossiers. Tous les fichiers se terminant par .cfg seront lu et importé par cette source.
cfg_file	cfg_file=mес_hotes.cfg	Il permet d'indiquer un seul fichier à lire. Ce n'est pas obligé de se terminer par .cfg , mais il doit en conserver la syntaxe
resource_file	resource_file=mon_infrastructure	Il permet d'indiquer un seul fichier à lire. Ce n'est pas obligé de se terminer par .cfg , mais il doit en conserver la syntaxe

Ces paramètres peuvent être cumulés afin de lire plusieurs répertoires ou plusieurs fichiers.

Voici un exemple de ce fichier de définition qui permet de lire les dossiers et fichiers suivants :

- elements_prod_Paris
- elements_prod_Marseille
- infrastructure_reseau

Si d'autres dossiers ou fichiers sont présents dans le répertoire de données de la source, ils ne seront pas lus dans cet exemple.

Exemple de fichier de définition

```
cfg_dir=elements_prod_Paris
cfg_dir=elements_prod_Marseille
cfg_file=infrastructure_reseau
```

Configuration du démon Synchronizer pour déclarer la source



Modifiez le fichier `/etc/shinken/synchronizers/synchronizer-master.cfg`

A la fin de la ligne "sources", ajoutez votre source.

Exemple :

```
sources          syncui, cfg-file-shinken, active-dir-example, sync-vmware, cfg-file-nagios, discovery,
openldap-example, cfg-file-ma-source
```

Redémarrez le synchronizer pour qu'il prenne acte de la nouvelle source :

```
service shinken-synchronizer restart
```

Affichage de la source dans l'UI de configuration

Une fois le démon redémarré, votre nouvelle source personnalisée doit apparaître :

Son paramétrage sera alors en place : son état (activée ou désactivée), son ordre, sa fréquence en automatique ou en manuel.

Collecteur 									
Ordre	Nom	Activé	État	Prochain import	Forcer l'import	Éléments	Résultat	Le dernier import	
1	cfg-file-shinken	☐	Désactivé						
2	cfg-file-ma-source	☒	Ok	Dans 1 min		0	 Le fichier de configuration <code>/etc/shinken-user/source-data/source-data-cfg-ma-source/definition.cfg</code> a été correctement chargé.	Il y a 3 min	

Import des objets

Depuis la page d'accueil de l'interface de configuration, vous pouvez alors utiliser le bouton **"Force import"** en cliquant sur le bouton 

Dans le panel des éléments, vos nouveaux éléments doivent apparaître. Vous pourrez alors les importer dans votre base.



Depuis une page spécifique de source, vous pouvez également appuyer sur le bouton  (il ne sera disponible que si la source est active). Une pop-up sera alors affichée jusqu'à la fin de l'import, ensuite la page sera rechargée pour être mise à jour.

Données globales

Les données globales peuvent être configurées dans `/etc/shinken/resource.d/`.

- Ce dossier contient des fichiers définissant des données globales pour différents sujets (email, mysql, oracle, ...) accessibles par vos objets globalement dans Shinken.

- Vous trouverez plus de détail sur l'utilisation des données globales dans le chapitre [Les Variables \(Remplacement dynamique de contenu - Anciennement les Macros \)](#)

Pour une organisation plus facile et une configuration plus propre, les données globales peuvent également être configurées dans les sources.

- Le dossier d'exemple d'une source cfg est situé dans ***/etc/shinken-user/source-data/source-data-cfg-sample*** et contient un dossier global-data dans lequel les définitions des données globales peuvent être placées (un fichier échantillon est également présent dans ce dossier)
- Lorsque la source est importée, ces fichiers de données globales seront copiés dans ***/etc/shinken/resource.d/module-source_name***

Ce mécanisme vous permet également de surcharger les valeurs définies dans ***/etc/shinken/resource.d*** avec des fichiers définis dans les sources.

Par contre ces fichiers contenant des définitions de donnée globale ne sont pris en compte **qu'après le rechargement du Synchronizer**.

Précisions techniques

Clés de synchronisation

Les clés de synchronisation sont des valeurs utilisées lors de l'étape du mélange des sources (Voir [Modules de Sources \(imports \)](#) et de [Taggers \(qualification \)](#)) qui permet de choisir quel élément de cette source se mélange avec quel élément d'une autre source (Voir [Le mélange des sources & les clés de synchronisation \(sync-key\)](#)).

Le paramètre `properties_used_as_synckey` de la source permet d'ajouter les propriétés qui serviront à créer les clés de synchronisation (Voir [Module de source de type cfg-file-import](#)).

Propriétés par défaut utilisé pour la construction des clés de synchronisation

Propriété	Type d'élément	Info
Nom de l'élément	Tous les éléments	Cette propriété ne peut pas être retirée des propriétés utilisées pour faire les clés de synchronisation
_SE_UUID	Tous les éléments	Cette propriété ne peut pas être retirée des propriétés utilisées pour faire les clés de synchronisation
address	hôte	