# Module architecture-export

**Sommaire**


- Introduction
- Configuration du module
  - Détails des sections composant le fichier de configuration
    - Identification du module
    - Chemin d'accès à NagVis
    - Options de fonctionnement du module
    - Communication du module
      - Configuration des connexions entrantes
        - Communication avec le listener-shinken
    - Paramètres de création de cartes
    - Paramètre de communication avec le Broker et ses modules
- Envoi de la description de l'architecture


## Introduction

Le module architecture-export permet

- d'envoyer une description de l'architecture d'installation Shinken
- de recevoir cette configuration ( ou celle d'une autre installation ) et d'en générer :
  - des cartes NagVis
  - des hôtes Shinken qui seront envoyés à leur tour au Synchronizer pour les superviser

Une description détaillée de ce module, son utilité et son utilisation sont présentes dans les pages de documentations associées : Visualiser l'architecture de son installation Shinken

Ci-dessous sont présentées de manière synthétique les différentes options de configuration de ce module, leur rôle ainsi que leurs valeurs par défaut.

## Configuration du module

Le module architecture-export est configurable via le fichier de configuration suivant: /etc/shinken/modules/architecture-export.cfg

Les paramètres modifiables dans le fichier sont les suivants:

**/etc/shinken/modules/architecture-export.cfg**

```
#===============================================================================
# Architecture-Export
#===============================================================================
# Module that export daemon map from arbiter:
#===============================================================================

define module {

    #======== Module identity =========
    # Module name. Must be unique
    module_name             architecture-export

    # Module type (to load module code). Do not edit.
    module_type             architecture_export



    #======== File path =========
    # path: Path of NagVis installation to save the maps configuration files in. Default value is defined by
Shinken installation.
    path                    /etc/shinken/external/nagvis

    #======== Architecture description communication =========
    # This module opens a listening socket on which other shinken installations will send their architecture
```

```
description.
    # When an architecture description is received by the module, it creates corresponding hosts and NagVis
maps.

    # host: interface used for the listening socket ( 0.0.0.0 = all interfaces )
    host                    0.0.0.0

    # Port to use for the listening socket
    port                    7780

    # 0 = Use HTTP, 1 = Use HTTPS
    use_ssl                 0
    ssl_cert                /etc/shinken/certs/server.cert
    ssl_key                 /etc/shinken/certs/server.key

    # Name with which this Shinken installation will be identified in the NagVis maps
    # The following characters are forbidden in the architecture name: ~!$%^&*"'|<>?,()=/+
    architecture_name       Shinken

    # Base of URL used to display links in the Visualization UI
    # Defaults to Arbiter URL if empty
    # map_base_url  http://example.com/

    # Architecture description recipients
    #   When the architecture of this Shinken installation changes ( realms and daemons configuration ),
    #   and the arbiter is restarted, the architecture description will be sent to the following hosts.
    send_my_architecture_to_recipients   http://127.0.0.1:7780
    # Connection to the shinken-listener
    #   if the connection parameter to the shinken-listener has been changed, set the new value here.
    #listener_use_ssl                    0
    #listener_login                      login
    #listener_password                   pass

    # ==================================    Broker connection parameters
============================================= #

#
#
    # --- These parameters are used to allow nagvis to communicate with the Broker and modules you
want                  ---

    # --- Name of the Broker holding the modules you want nagvis to communicate
with                                    ---
    # >>> DEFAULT : broker-
master                                                                                          ---
    # architecture_export__broker_connection__broker_name                                         broker-
master

    # --- Name of the Livestatus module you want nagvis to communicate with to retrieve objects
information               ---
    # >>> DEFAULT :
Livestatus                                                                                              ---
    # architecture_export__broker_connection__broker_livestatus Livestatus

    # --- Type of the target WebUI you want to communicate
with                                                        ---
    # --- This allows redirection when clicking on objects on the
maps                                                    ---
    # >>> module : Use a WebUI module configuration to get it's address ( Default
)                                     ---
    # --- url : Use a URL ( the WebUI address is behind a reverse proxy or use an DNS address
)                           ---
    # architecture_export__broker_connection__broker_webui_communication_type module

    # --- Targetted WebUI to communicate
with                                                                        ---
    # --- If previous parameter is set to "module", this must be a WebUI
name                                            ---
    # --- If previous parameter is set to "url", this must be a
URL                                                 ---
    # >>> Default :
```

```
WebUI                                                                                    ---
    # architecture_export__broker_connection__broker_webui_target                      WebUI
}
```

## Détails des sections composant le fichier de configuration

### Identification du module

Il est possible de définir l'identité du module de type **architecture-export** .

| Nom | Type | Unité | Défaut | Commentaire |
|---|---|---|---|---|
| `module_name` | Texte | --- | **architecture-export** | **Nous vous conseillons de choisir un nom en fonction de l'utilisation du module pour que votre configuration soit simple à maintenir.**<br><br>Doit être unique. |
| `module_type` | Texte | --- | **architecture-export** | Ne peut être modifié. |

### Chemin d'accès à NagVis

```
       ...

         #======== File path =========
    # path: Path of NagVis installation to save the maps configuration files in. Default value is defined by
Shinken installation.
    path                    /etc/shinken/external/nagvis

       ...
```

Afin de modifier les paramètres de NagVis et de sauvegarder les fichiers de cartes générées, le module doit connaître le chemin d'accès vers le NagVis utilisé

| Nom | Type | Unité | Défaut | Commentaire |
|---|---|---|---|---|
| `path` | Texte | --- | **/etc/shinken/external/nagvis** | Chemin d'accès vers l'installation NagVis utilisée par le module. |

### Options de fonctionnement du module

```
    ...

    # Base of URL used to display links in the Visualization UI
# Defaults to Arbiter URL if empty
# map_base_url  http://example.com/

# Architecture description recipients
#   When the architecture of this Shinken installation changes ( realms and daemons configuration ),
#   and the arbiter is restarted, the architecture description will be sent to the following hosts.
send_my_architecture_to_recipients   http://127.0.0.1:7780

    ...
```

Le module génère des cartes et envois son architecture à d'autres modules du même type. Il est possible de spécifier l'URL d'accès aux cartes, mais aussi la liste des modules à qui envoyer sa propre architecture

| Nom | Type | Unité | Défaut | Commentaire |
|---|---|---|---|---|
| `map_base_url` | Texte | --- | **Url de l'Arbiter** | URL d'accès aux cartes générées par le module. L'URL sera utilisée pour les liens dans l'interface de Visualisation. |
| `send_my_architecture_to_recipients` | Texte | --- | **http://127.0.0.1:7780** | Liste des adresses des modules architecture-export vers lesquels envoyer son architecture. Les adresses doivent être séparées par des virgules. |

### Communication du module

Le module doit écouter vers l'extérieur pour recevoir des architectures, mais il doit aussi communiquer avec le listener-shinken du Synchronizer pour lui envoyer les hôtes générés pour ses cartes. Ces paramètres de communications sont modifiables pour correspondre à chaque architecture.

#### Configuration des connexions entrantes

```
    ...

     # host: interface used for the listening socket ( 0.0.0.0 = all interfaces )
host                    0.0.0.0

# Port to use for the listening socket
port                    7780

# 0 = Use HTTP, 1 = Use HTTPS
use_ssl                 0
ssl_cert                /etc/shinken/certs/server.cert
ssl_key                 /etc/shinken/certs/server.key

    ...
```

| Nom | Type | Unité | Défaut | Commentaire |
|---|---|---|---|---|
| `host` | Texte | --- | **0.0.0.0** | Interface réseau sur lequel le module écoutera.<br>**Remarque** : 0.0.0.0 correspond à toutes les interfaces. |
| `port` | Texte | --- | **7780** | Port d'écoute du module. |

| | | | | |
|---|---|---|---|---|
| `use_ssl` | Booléen | --- | **0** | Paramètre activant le mode SSL ( *HTTPS* ).<br>Valeurs possibles :<br>• **0**<br>• **1** |
| `ssl_cert` | Texte | --- | **/etc/shinken/certs/server.cert** | Chemin d'accès vers le certificat SSL à utiliser ( *si use_ssl est à 1* ). |
| `ssl_key` | Texte | --- | **/etc/shinken/certs/server.key** | Chemin d'accès vers la clé SSL à utiliser ( *si use_ssl est à 1* ). |

### Communication avec le listener-shinken

```
    ...

      # Connection to the shinken-listener
#   if the connection parameter to the shinken-listener has been changed, set the new value here.
#listener_use_ssl                    0
#listener_login                      login
#listener_password                   pass

    ...
```

| Nom | Type | Unité | Défaut | Commentaire |
|---|---|---|---|---|
| `listener_use_ssl` | Texte | --- | **0** | Paramètre activant le mode SSL pour la communication avec le listener-shinken ( *HTTPS* ).<br>Valeurs possibles :<br>• **0**<br>• **1** |
| `listener_login` | Texte | --- | **Shinken** | Nom d'utilisateur à utiliser pour communiquer avec le listener-shinken. |
| `listener_password` | Texte | --- | **mot de passe généré à l'installation** | Mot de passe à utiliser pour communiquer avec le listener-shinken. |

## Paramètres de création de cartes

```
    ...

       # Name with which this Shinken installation will be identified in the NagVis maps
# The following characters are forbidden in the architecture name: ~!$%^&*"'|<>?,()=/+
architecture_name       Shinken

    ...
```

Il est possible de paramétrer certains aspects de la création des cartes NagVis

| Nom | Type | Unité | Défaut | Commentaire |
|---|---|---|---|---|
| `architecture_name` | Texte | --- | **Shinken** | Nom de l'architecture de l'installation Shinken où se situe le module.<br>Ce nom sera affiché sur les cartes NagVis et les hôtes générés |

### Paramètre de communication avec le Broker et ses modules

```
        ...

    # ==================================    Broker connection parameters
============================================== #

#
#
    # --- These parameters are used to allow nagvis to communicate with the Broker and modules you
want                   ---

    # --- Name of the Broker holding the modules you want nagvis to communicate
with                                      ---
    # >>> DEFAULT : broker-
master                                                                                  ---
    # architecture_export__broker_connection__broker_name                                 broker-
master

    # --- Name of the Livestatus module you want nagvis to communicate with to retrieve objects
information               ---
    # >>> DEFAULT :
Livestatus                                                                                   ---
    # architecture_export__broker_connection__broker_livestatus                           Livestatus

    # --- Type of the target WebUI you want to communicate
with                                                          ---
    # --- This allows redirection when clicking on objects on the
maps                                                     ---
    # >>> module : Use a WebUI module configuration to get it's address ( Default
)                                        ---
    # --- url : Use a URL ( the WebUI address is behind a reverse proxy or use an DNS address
)                            ---
    # architecture_export__broker_connection__broker_webui_communication_type module

    # --- Targetted WebUI to communicate
with                                                                    ---
    # --- If previous parameter is set to "module", this must be a WebUI
name                                              ---
    # --- If previous parameter is set to "url", this must be a
URL                                                    ---
    # >>> Default :
WebUI                                                                                        ---
    # architecture_export__broker_connection__broker_webui_target                          WebUI

        ...
```

Afin d'avoir les statuts des hôtes sur les cartes NagVis, mais aussi d'être correctement redirigé vers une WebUI lors du clic sur l'un d'entre eux, il existe des paramètres pour rendre la communication entre NagVis et Shinken possible

| Nom | Type | Unité | Défaut | Commentaire |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| `architecture_export__broker_connection__broker_name` | Texte | --- | **broker-master** | Nom du Broker sur lequel sont les modules avec lesquels nous allons communiquer |
| `architecture_export__broker_connection__broker_livestatus` | Texte | --- | **Livestatus** | Nom du module Livestatus avec lequel NagVis va communiquer pour afficher les statuts des hôtes sur ses cartes |
| `architecture_export__broker_connection__broker_webui_communication_type` | Texte | --- | **module** | Type de communication avec la WebUI souhaitée. Ce paramètre est utilisé pour la redirection lorsqu'on clique sur les liens de la carte.<br><br>Valeurs possibles :<br><br>• **module** ( *si on souhaite spécifier le nom d'un module WebUI pour obtenir son adresse* )<br>• **URL** ( *si on souhaite renseigner une URL : l'adresse de la WebUI est derrière un reverse proxy ou utilise une adresse DNS* ) |
| `architecture_export__broker_connection__broker_webui_target` | Texte | --- | **WebUI** | WebUI avec laquelle communiquer.<br><br>• Si le paramètre précédent est à **module,** celui-ci doit être le nom d'un module WebUI<br>• Sinon, ce doit être une URL redirigeant vers un module WebUI<br><br>⚠ La valeur de ce paramètre viendra écraser la configuration des paramètres "**hosturl**" et "**servicegroupurl**" du fichier "**/opt/nagvis/etc/nagvis.ini.php**" au redémarrage de l'Arbiter. |

## Envoi de la description de l'architecture

L'envoi de la description de l'architecture aux destinataires choisis ( *paramètre* ***send_my_architecture_to_recipients*** ) est déclenché au démarrage du module architecture-export, c'est-à-dire au démarrage du démon Arbiter.

Il est également possible de déclencher cet envoi manuellement, sans avoir à redémarrer le démon Arbiter, en envoyant une requête HTTP POST à l'URL suivante :

```
adresse_arbiter:7780/v1/architecture/send
```

Par exemple avec cURL :

```
curl -v -X POST http://localhost:7780/v1/architecture/send
```