

FISE 2

Documentation utilisateur

Kim-Celine Franot
2021 - 2022

Table des matières

Introduction.....	2
Téléchargement du projet.....	2
Mise en place du fichier de configuration.....	2
Installation des dépendances et lancement de l'utilitaire	6
Annexes	7
Installation des dépendances manuellement	7
Création d'un fichier CRON manuellement.....	7

Introduction

Nous présenterons ici la documentation utilisateur d'un utilitaire d'archivage. Cet utilitaire permet de récupérer un fichier ZIP à partir d'une URL particulière, d'extraire le fichier SQL contenu dedans et de le compresser au sein d'une archive TAR.

Par la suite, cette archive est transférée vers un serveur SMB (Samba) distant. L'utilitaire peut également vérifier la validité des anciennes archives situées sur le serveur distant.

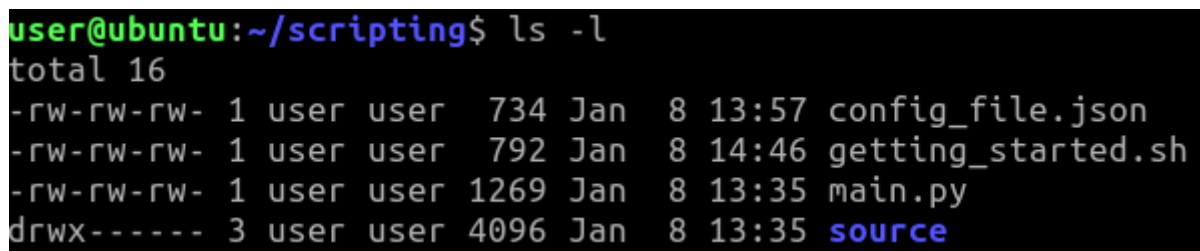
En fin d'exécution, l'utilitaire a la possibilité d'envoyer des mails et des notifications sur un serveur Mattermost.

Téléchargement du projet

Après avoir téléchargé l'archive contenant le projet, il faut le décompresser dans un répertoire de votre choix sur la machine locale. Dans le reste de la documentation, le répertoire **scripting** est pris comme exemple.

Le répertoire (ici **scripting**) doit ainsi contenir les fichiers et dossier suivants :

- **getting_started.sh** : un script bash permettant de télécharger l'ensemble des dépendances nécessaires, de créer un fichier CRON afin d'automatiser l'utilitaire et de le lancer.
- **main.py** : le programme principal en python qui fait appel à l'ensemble des instructions pour exécuter l'utilitaire.
- **source** : dossier contenant l'ensemble des fichiers pour les classes en Python utilisé par le programme principal.
- **config_file.json** : un fichier de configuration à modifier avec les données de l'utilisateur.



```
user@ubuntu:~/scripting$ ls -l
total 16
-rw-rw-rw- 1 user user 734 Jan  8 13:57 config_file.json
-rw-rw-rw- 1 user user 792 Jan  8 14:46 getting_started.sh
-rw-rw-rw- 1 user user 1269 Jan  8 13:35 main.py
drwx----- 3 user user 4096 Jan  8 13:35 source
```

Figure 1 : Contenu du répertoire de travail

Mise en place du fichier de configuration

Il faut tout d'abord procéder à la modification du fichier `config_file.json` pour faire fonctionner l'utilitaire correctement. Pour cela, il faut ouvrir le fichier dans un éditeur de texte. Le contenu devrait être similaire à l'exemple ci-dessous (cf. figure 2) :

```
{
  "url-zip": "https://opensource.telecomste.fr/inforx/test_export.sql.zip",
  "file": "test_export.sql",
  "save_days": "7",

  "smb": {
    "ip": "192.168.1.79",
    "hostname": "debian",
    "user": "user",
    "password": "user",
    "sharename": "samba-share",
    "path": "/"
  },
  "email": {
    "auth": {
      "email": "toto@domain.com",
      "password": "motdepasse"
    },
    "server": {
      "host": "smtp.gmail.com",
      "port": "465"
    },
    "send-mail": "yes",
    "include_log": "yes",
    "title": "Daily summary",
    "dest": [
      "monmail1@domain.com",
      "monmail2@gmail.com"
    ]
  },
  "notification": "always",
  "mattermost_webhook": "https://chat.telecomste.fr/hooks/13ta3nw3e787tnxxqkyxfh5xnr"
}
```

Figure 2 : Contenu du fichier de configuration

Chaque champ est constitué d'un mot-clé (**Key**) et d'une valeur (**Value**). Il faut donc remplacer chaque valeur actuelle par celle qui correspond aux paramètres de l'utilisateur. Attention à ne pas oublier les guillemets pour chaque valeur renseignée.

Les différents champs sont répertoriés par ordre chronologique d'apparition dans le tableau suivant avec leur signification.

Description des différents champs :

Key	Value	Exemple
url-zip	URL du fichier ZIP à récupérer	"https://opensource.telecomste.fr/inforx/test_export.sql.zip"
file	Nom du fichier SQL contenu dans l'archive ZIP	Pour un fichier nommé test_export.sql, renseigner : "test_export.sql"
save-days	Durée de conservation des archives sur le serveur distant en jours Valeur par défaut : 7	Pour conserver les archives 5 jours, renseigner "5".

Configuration du serveur smb (**Key** : smb)

Key	Value	Exemple
ip	Adresse IP du serveur distant où seront stockés les archives	Si le serveur a pour adresse 192.168.1.79 , renseigner "192.168.1.79"
hostname	Nom de la machine du serveur distant Valeur par défaut : "debian"	Pour trouver le nom, saisir la commande cat /etc/hostname
user	Nom de l'utilisateur Samba	
password	Mot de passe de l'utilisateur Samba	
sharename	Nom du partage Samba	Si le partage est nommé samba-share dans le fichier de configuration smb.conf, renseigner "samba-share"
path	Chemin vers le répertoire (créé au préalable) où seront stockés les fichiers envoyés sur le serveur distant Valeur par défaut (à la racine du partage) : "/"	Si on souhaite stocker les fichiers dans le répertoire archives situé à la racine du partage, renseigner "/archives/"

Configuration de l'envoi d'email (**Key** : email)

Key	Value	Exemple
auth	email	Adresse mail utilisée pour envoyer le mail Pour une adresse mail toto@domain.com , renseigner "toto@domain.com"
	password	Mot de passe associé à l'adresse mail saisie précédemment Pour le compte toto@domain.com , le mot de passe est motdepasse , alors renseigner "motdepasse"

server	host	Adresse du serveur SMTP de la messagerie liée à l'adresse mail Valeur par défaut (pour une adresse Gmail) : "smtp.gmail.com"	Pour une adresse Outlook : "smtp-mail.outlook.com" Pour une adresse Yahoo : "smtp.mail.yahoo.com"
	port	Port SMTP lié au serveur Valeur par défaut (pour Gmail) : "465"	Pour une adresse Outlook : "587" Pour une adresse Yahoo : "465"
send-mail		2 valeurs possibles : "yes"/"y" : permet d'envoyer un mail aux destinataires saisis récapitulant le déroulement du processus. "no"/"n" : aucun mail ne sera envoyé à la fin du processus Valeur par défaut : "yes"	
include_log		2 valeurs possibles : "yes"/"y" : permet d'inclure le fichier de log en pièce jointe au mail. "no"/"n" : le fichier de log ne sera pas inclus en pièce jointe au mail. Valeur par défaut : "yes"	
title		Objet du mail à envoyer Valeur par défaut : "Daily summary"	
dest		Adresse(s) mail(s) à qui envoyer un mail à la fin du processus	Si un seul destinataire, ajouter entre les crochets son adresse mail entre guillemets : ["toto@outlook.com"] Si plusieurs destinataires, ajouter entre les crochets chaque adresse entre guillemets séparé d'une virgule : ["toto@outlook.com", "toto@gmail.com"]

Key	Value
notification	<p>3 valeurs possibles :</p> <p>"always" : envoie une notification sur le serveur Mattermost à chaque fois que l'utilitaire a été exécuté</p> <p>"error" : envoie une notification sur le serveur Mattermost lorsqu'une erreur a été rencontrée lors de l'exécution de l'utilitaire</p> <p>"never" : n'envoie pas de notification sur le serveur Mattermost à la fin de l'exécution</p> <p>Valeur par défaut : "always"</p>
mattermost_webhook	<p>URL du webhook vers le serveur Mattermost sur lequel envoyer la notification.</p> <p>Valeur par défaut (webhook vers le canal INFORX-SCRIPT du serveur Mattermost pédagogique de TSE) :</p> <p>"https://chat.telecomste.fr/hooks/13ta3nw3e787tnxxqkyxfh5xnr"</p>

Installation des dépendances et lancement de l'utilitaire

Une fois le fichier de configuration rempli, il faut rendre exécutable le script bash **getting_started.sh**.

Pour cela, il faut saisir la commande **chmod +x getting_started.sh** dans un terminal de commande (bien vérifier qu'on est dans le répertoire de travail défini précédemment).

Pour vérifier que le fichier est exécutable, saisir la commande **ls -l** (cf. figure 3).

```
user@ubuntu:~/scripting$ chmod +x getting_started.sh
user@ubuntu:~/scripting$ ls -l
total 16
-rw-rw-rw- 1 user user 734 Jan 8 13:57 config_file.json
-rwxrwxrwx 1 user user 792 Jan 8 14:46 getting_started.sh
-rw-rw-rw- 1 user user 1269 Jan 8 13:35 main.py
drwx----- 3 user user 4096 Jan 8 13:35 source
```

Figure 3 : Après avoir rendu le fichier exécutable

On peut alors lancer le script qui va permettre de télécharger l'ensemble des dépendances manquantes pour faire fonctionner l'utilitaire. Ce script va également permettre la création d'un fichier CRON pour automatiser l'exécution de l'utilitaire quotidiennement à 12h00.

Pour lancer le script, il vous suffit de saisir la commande suivante : **./getting_started.sh**

```

user@ubuntu:~/scripting$ ./getting_started.sh
Python 3.8.10
[sudo] password for user:
Reading package lists... Done
Building dependency tree
Reading state information... Done
python3 is already the newest version (3.8.2-0ubuntu2).
The following packages were automatically installed and are no longer required:
  linux-image-5.4.0-54-generic linux-modules-5.4.0-54-generic linux-modules-extra-5.4.0-54-generic
Use 'sudo apt autoremove' to remove them.
0 to upgrade, 0 to newly install, 0 to remove and 76 not to upgrade.
Reading package lists... Done
Building dependency tree
Reading state information... Done
python3-pip is already the newest version (20.0.2-5ubuntu1.6).
The following packages were automatically installed and are no longer required:
  linux-image-5.4.0-54-generic linux-modules-5.4.0-54-generic linux-modules-extra-5.4.0-54-generic
Use 'sudo apt autoremove' to remove them.
0 to upgrade, 0 to newly install, 0 to remove and 76 not to upgrade.
pip 20.0.2 from /usr/lib/python3/dist-packages/pip (python 3.8)
Collecting email-validator
  Using cached email_validator-1.1.3-py2.py3-none-any.whl (18 kB)
Requirement already satisfied: dnspython>=1.15.0 in /home/user/.local/lib/python3.8/site-packages (from email-validator) (2.1.0)
Requirement already satisfied: idna>=2.0.0 in /usr/lib/python3/dist-packages (from email-validator) (2.8)
Installing collected packages: email-validator
  WARNING: The script email_validator is installed in '/home/user/.local/bin' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed email-validator-1.1.3
Processing /home/user/.cache/pip/wheels/39/28/cc/8c5b4f43ef23d12ff4b08f68c1f7c9cf6e4c3f8de5eac1fa9e/pysmb-1.2.7-py3-none-any.whl
Requirement already satisfied: pyasn1 in /home/user/.local/lib/python3.8/site-packages (from pysmb) (0.4.8)
Installing collected packages: pysmb
Successfully installed pysmb-1.2.7
Crontab created in /var/spool/cron/crontabs

```

Figure 4 : Exécution du script bash

Il est possible de visualiser le fichier CRON en saisissant la commande **crontab -l**.

```

user@ubuntu:~/scripting$ crontab -l
00 12 * * * cd /home/user/scripting; python3 main.py

```

Figure 5 : Vérification du fichier CRON

Annexes

Installation des dépendances manuellement

- Pour obtenir la dernière version de python :
sudo apt-get install --upgrade python3
- Pour installer pip :
sudo apt-get install python3-pip
- Pour installer les dépendances pour l'utilitaire :
python3 -m pip install pysmb
python3 -m pip install email-validator

Création d'un fichier CRON manuellement

- Saisir la commande **crontab -e** pour ouvrir un fichier CRON
- Editer le contenu en rajoutant la ligne suivante (pour une automatisation quotidienne à 12h00) :
00 12 * * * cd <chemin vers répertoire de travail> ; python3 main.py
Il est possible de s'aider du site <https://crontab.guru> pour configurer l'automatisation du CRON
- Vérifier que le fichier CRON a été sauvegardé avec la commande **crontab -l**