



ARCHIVING TOOL: USER GUIDE

LEANG Denis



2021-2022

FISE 2

Telecom Saint-Etienne

Table des matières

I – About the project	2
II - Getting started	3
• Prerequisites	3
• Installation	3
III - Usage	4
• Json configuration file	4
1. IP adress	6
2. Gmail adress	6
• SMB share server	7
1. Network activation	7
2. Folder creation.....	8
• Task scheduler	10
1. Batch file	10
2. Windows Task Scheduler.....	11
IV – Conclusion	12

I – About the project

This is a friendly user guide for my Python archiving tool. This script allows a user to do these successive actions:

1. Download a zip file from a web server (HTTP)
2. Extract that zip file
3. Check whether its content is up to date and valid
4. Compress it after validation into a tgz file
5. Send the tgz file to a server (SMB/CIFS)
6. (Optional) Send an email stating if operation succeeded or not

A logfile stores useful information throughout the whole operations to check if everything works properly or not. It can be attached to the email if the user chooses to do so.

Here is a simple drawing I have made at the beginning of the project to summarize the operations:

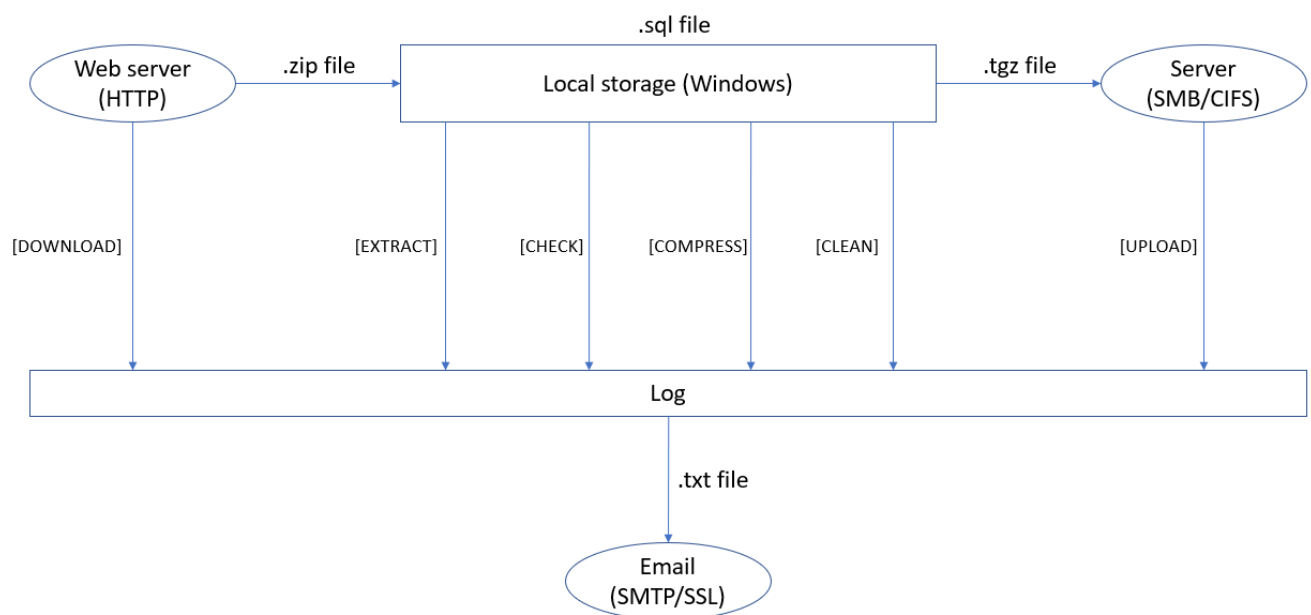


Figure 1: Simple representation of the operations

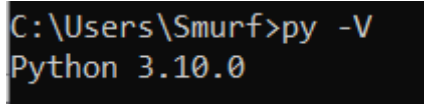
This script is aimed at Windows workstations for I have developed my script on Windows 10 using Visual Studio Code. Therefore, it may not work for MacOS or Linux users.

II - Getting started

- Prerequisites

Firstly, you will need to have Python 3 installed in your computer. If you wish to check if you already have a version of Python on your device, type in your terminal:

➤ **py -V**



```
C:\Users\Smurf>py -V
Python 3.10.0
```

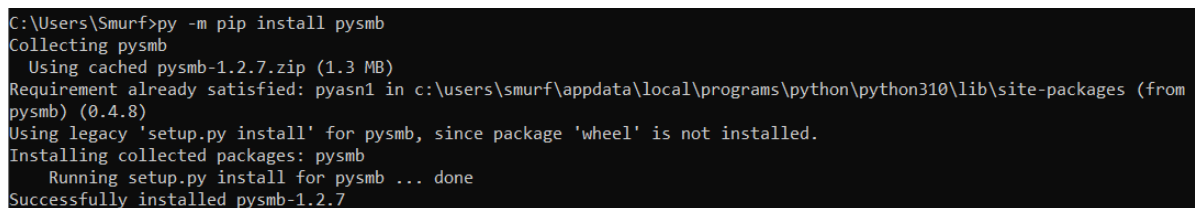
Figure 2: Check Python's version

If you do not have any, just install python [here](#).

- Installation

I only used two external libraries which are *requests* and *pysmb*. To install them, just run these commands in your terminal:

➤ **py -m pip install pysmb**
➤ **py -m pip install requests**



```
C:\Users\Smurf>py -m pip install pysmb
Collecting pysmb
  Using cached pysmb-1.2.7.zip (1.3 MB)
Requirement already satisfied: pyasn1 in c:\users\smurf\appdata\local\programs\python\python310\lib\site-packages (from
pysmb) (0.4.8)
Using legacy 'setup.py install' for pysmb, since package 'wheel' is not installed.
Installing collected packages: pysmb
  Running setup.py install for pysmb ... done
Successfully installed pysmb-1.2.7
```

Figure 3: pysmb library installation

III - Usage

- Json configuration file

The first thing that you will have to setup is the *config.json* file.

```
1  {
2      "archive": {
3          "url": "http://localhost:8000/Server/",
4          "zip_file": "data_zip.zip",
5          "file": "data.sql",
6          "log_file": "log.txt",
7          "expiring_time": 86400
8      },
9
10     "smb_client": {
11         "username": "username",
12         "password": "password",
13         "machine_name": "name",
14         "server_name": "pc-name",
15         "ip_address": "ip_address",
16         "smb_share": "smb_server_name",
17         "save_duration": 86400
18     },
19
20     "mail": {
21         "sender_adress": "sender@gmail.com",
22         "password": "password",
23         "port": "465",
24         "smtp_server": "smtp.gmail.com",
25         "receiver_adress": "receiverg@gmail.com",
26         "send_mode": "on",
27         "attach_mode": "on"
28     }
29 }
```

Figure 4: json.config file

The names of the json objects are conveniently named after the classes in the code. But all you must worry about as a user are the following elements. Changing the values in which “NO CHANGE” is specified is not advised:

➤ **archive:**

- “url”: NO CHANGE
- “zip_file”: NO CHANGE
- “file”: NO CHANGE
- “log_file”: Name of the generated log file
- “expiring_time”: Duration (in seconds) after which a file is considered outdated

➤ **smb_client:**

- “username”: Current windows session username
- “password”: Current windows session password
- “machine_name”: Whatever name you want to enter
- “server_name”: Name of the current PC
- “ip_address”: IP address
- “smb_share”: Name of the shared samba server (see after)
- “save_duration”: Duration (in seconds) after which a file will be deleted in the smb server depending on its last modification date

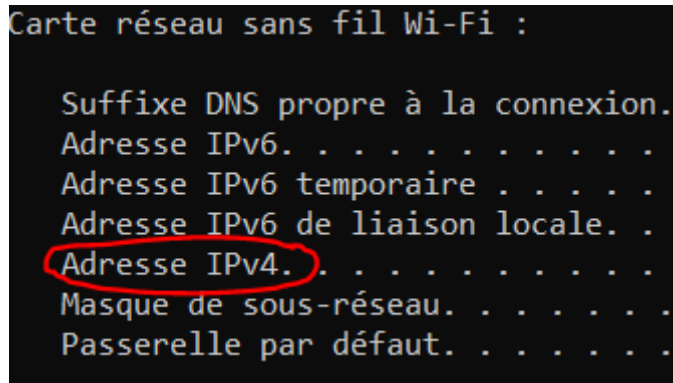
➤ **archive:**

- “sender_address”: An E-mail adress for the expeditor (Note: Gmail is advised)
- “password”: Password for the adress above
- “port”: NO CHANGE
- “smtp_server”: NO CHANGE
- “receiver_address”: An E-mail adress for the receiver (Note: Gmail is advised)
- “send_mode”: Put on or off depending on if you want to receive mails
- “attach_mode”: Put on or off depending on if you want to have logs attached in the mails

1. IP adress

If you have trouble finding your correct IP address, follow these steps:

1. Type *ipconfig* inside a terminal
2. Get the correct IP address in front of IPv4:



```
Carte réseau sans fil Wi-Fi :  
  
Suffixe DNS propre à la connexion.  
Adresse IPv6. . . . .  
Adresse IPv6 temporaire . . . . .  
Adresse IPv6 de liaison locale. .  
Adresse IPv4. . . . .  
Masque de sous-réseau. . . . .  
Passerelle par défaut. . . . .
```

Figure 5: ipconfig command to find IP adress

2. Gmail adress

Having a throwaway Gmail adress for the sender is advised for security reasons because you will have to enter login credentials inside the *config.json* to start the script. You must activate this mode: “*Allow less secure apps*” to on in Gmail. It is essential if you wish to receive the E-mails from the sender adress as a receiver. To find this mode, go to:

- Settings → Security → Access to less secure apps

- **SMB share server**

Now that the *config.json* is complete, you will now have to setup the SMB share server. You will have to follow these steps to ensure that everything will run smoothly later.

1. Network activation

Ensure that you have Network activated on your Windows workstation. It is normally activated by default. If not, follow these steps:

1. Go to control panel and click on *Network and Internet*

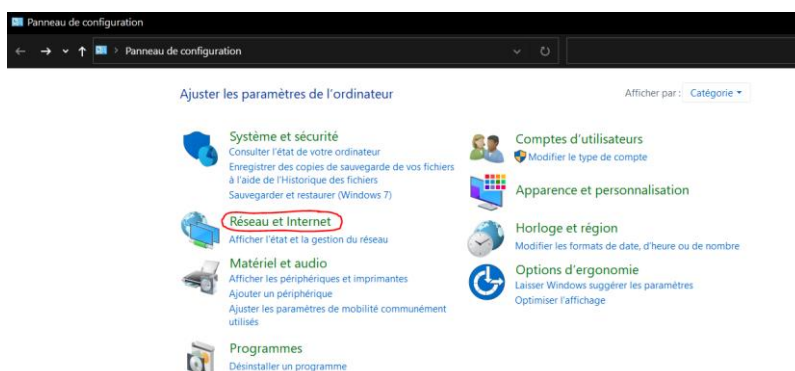


Figure 6: Network configuration

2. Go to *Network and Sharing Center*

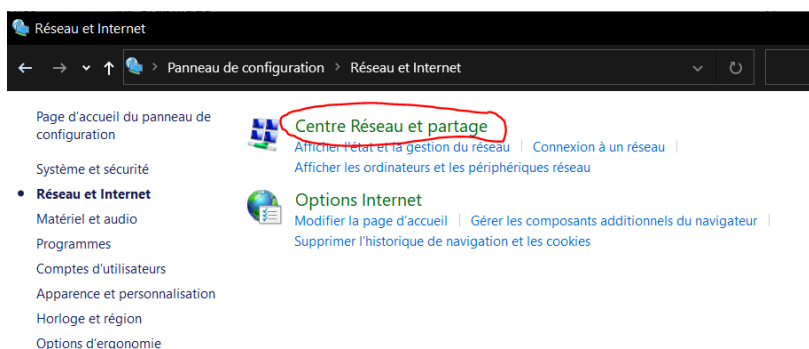


Figure 6: Network configuration

3. Finally in *Advanced sharing options*, check everything

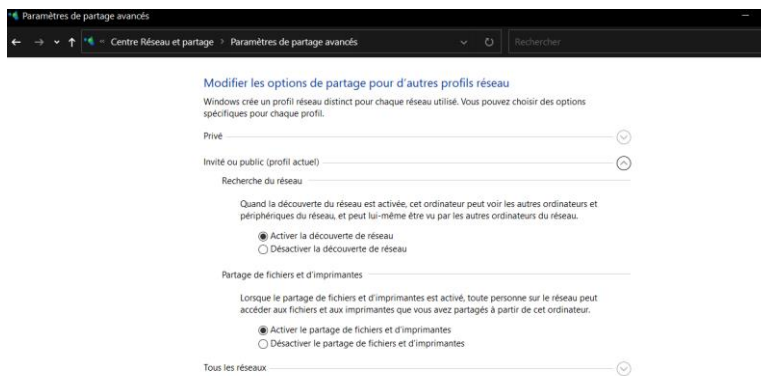


Figure 6: Network configuration

2. Folder creation

You will then have to create a folder in the root of the code's folder like this:

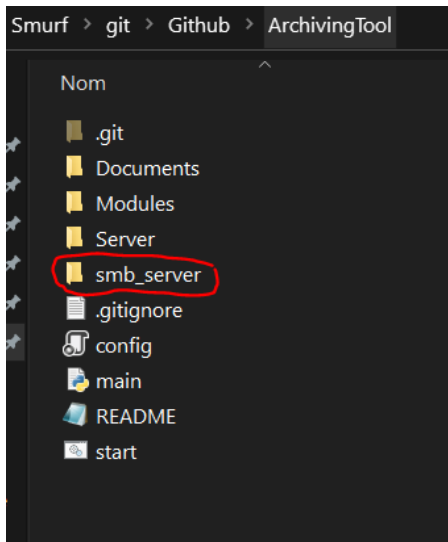


Figure 7: SMB folder creation

It is mandatory to create that folder here for the code to work properly so do not place it anywhere else.

Then, activate those settings by right clicking the folder to setup the SMB server:

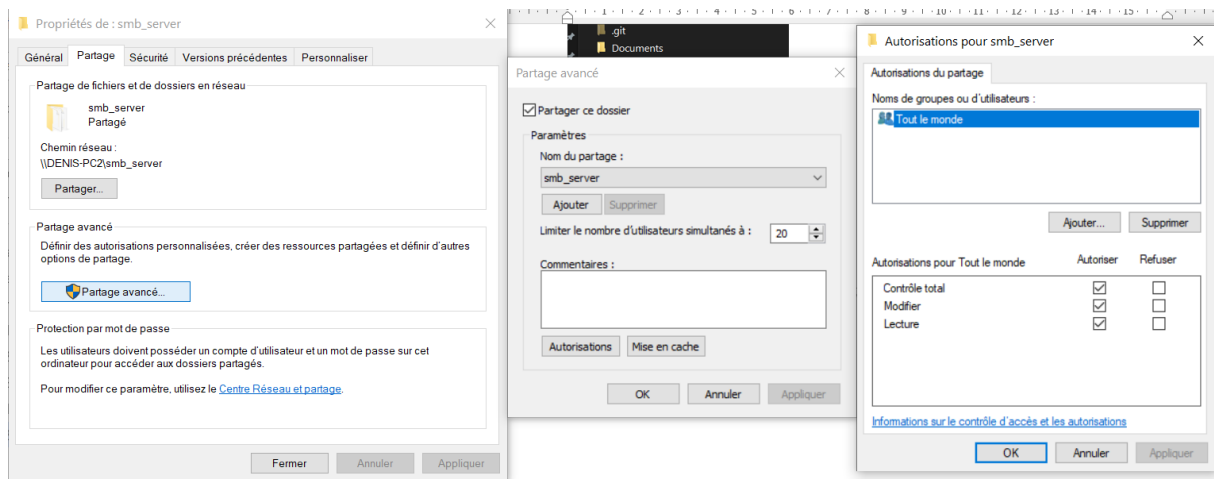


Figure 8: SMB folder creation

If everything went smoothly, you should now see this on your Windows explorer:

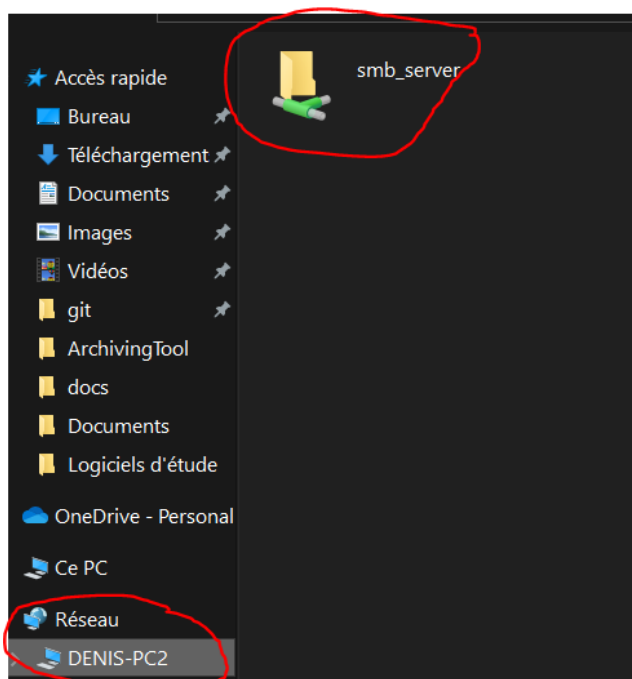


Figure 9: SMB folder in Network tab

The SMB share is now ready to use.

- **Task scheduler**

Everything should now be ready for the code to work. If you run *server.py* and *main.py* (in that order) on any IDE, the script will work. In our case, we want it to be automatic. To do so you need to follow these steps.

1. Batch file

You should see a start.bat file inside the folder. Inside it are these lines:

```
start py C:\Path\To\Script\Server\server.py  
schtasks /create /sc daily /st 00:00 /tn "Archival Tool" /tr C:\Path\To\Script\main.py
```

Figure 10: start.bat file content

- First line: Boots up the web server by launching *server.py*
- Second line: Create a new task on Windows Task Scheduler which will launch *main.py* daily at 00:00

You must change the path to match with your folder's path inside the batch file. It is also possible for you to change the frequency at which *main.py* is executed. For more information on how to do so, you can go [here](#).

When everything is changed, you can now run *start.bat* by double clicking it. One window will appear:

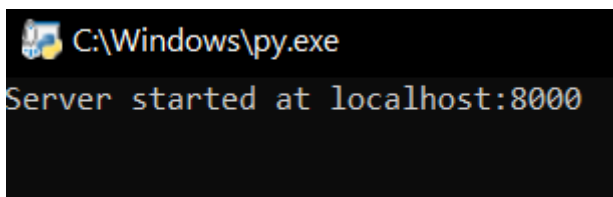


Figure 11: server.py launched

2. Windows Task Scheduler

You now must add path for the script to work in Task Scheduler.

1. Go to Task Scheduler, right click “Archiving Tool” and Properties:
2. Click on Actions and Modify

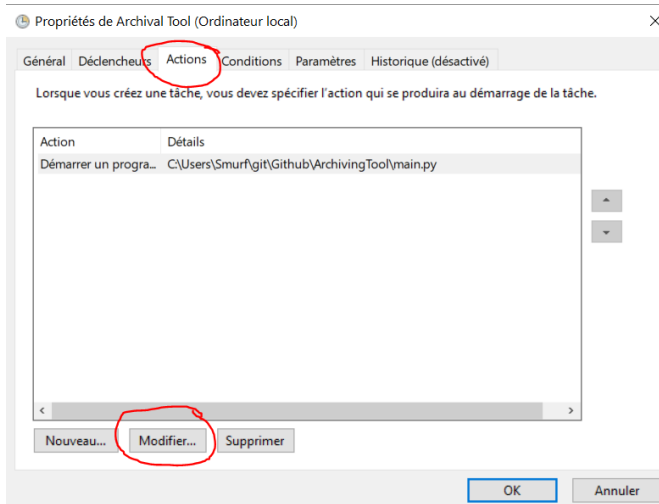


Figure 12: Path configuration

3. Add the path to your project’s root folder inside this:

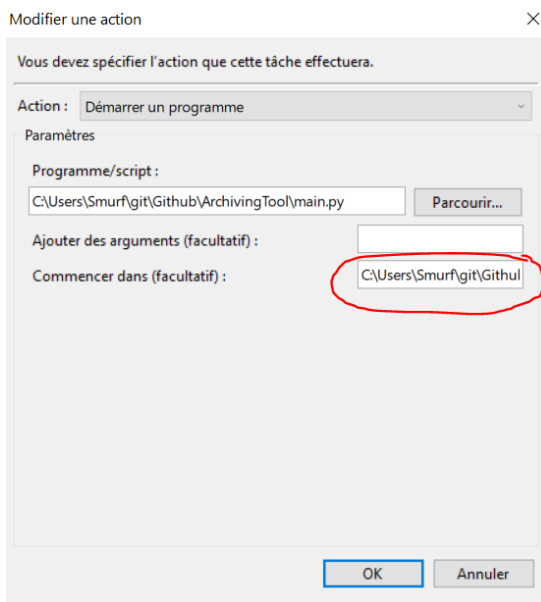


Figure 12: Path configuration

Note that **server.py** will run indefinitely until the workstation is shut down. So, beware to click again on **start.bat** if you close your computer at some point else the script will run but fail with the operations. You do not need to modify the task which runs **main.py** whatsoever so just type “N” if prompted to.

- Log Storage

In the current state of the code, the logfile is deleted at the end of the operations. This is to ensure that it does not pollute the current folder. However, if you wish to keep it, you will need to delete this line in *main.py* (L47): `archive.delete_log_file()`.

IV – Conclusion

Everything is now ready and set to work properly. You can thus enjoy storing files with this Archiving Tool.

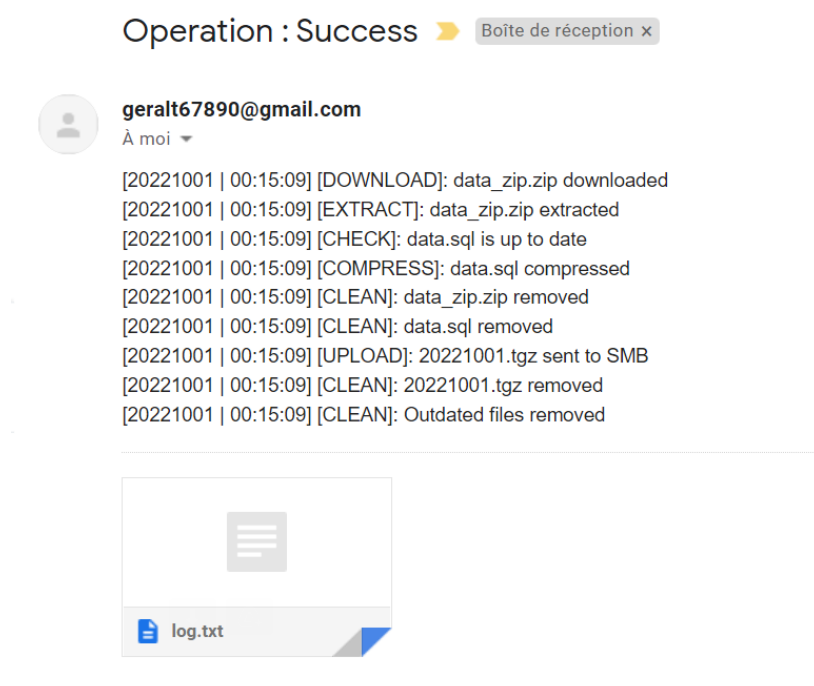


Figure 13: Successful transfer

Contact: denis.leang@gmail.com