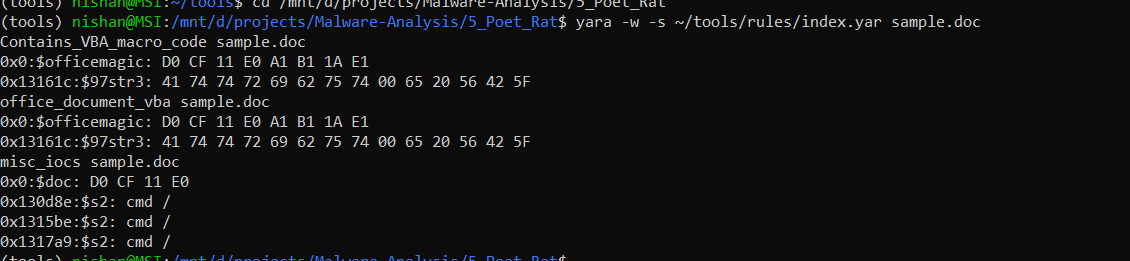
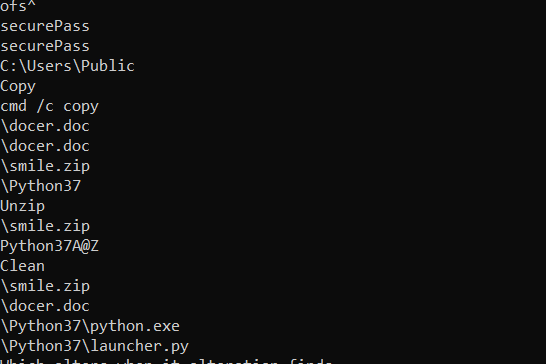
Sample: <https://app.any.run/tasks/5f8ffd19-61cf-4091-99c6-a909d3279745/>

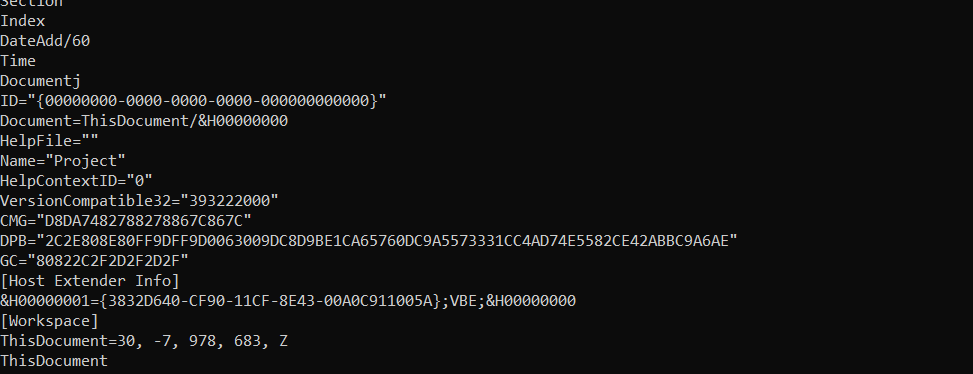
PoetRat Analysis



Contains macro code

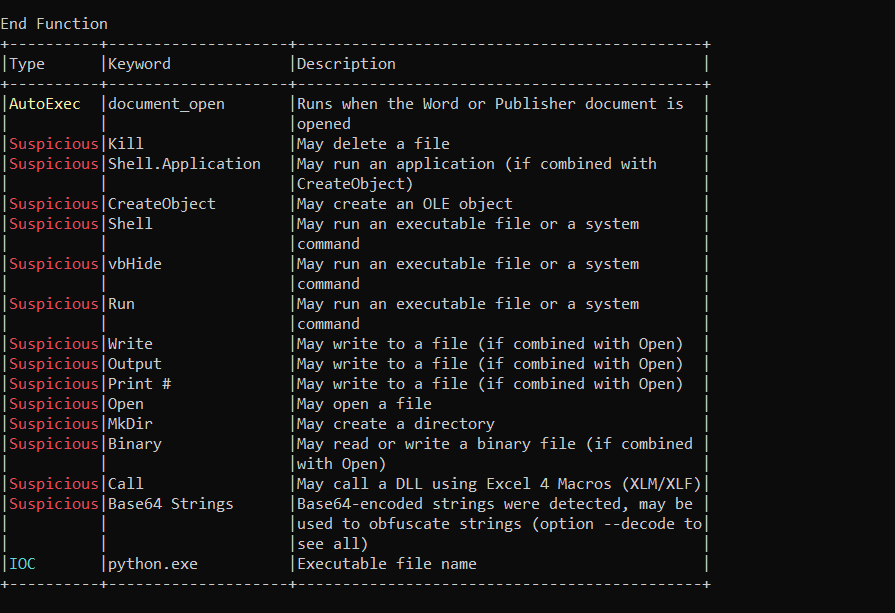
Cmd/ strings







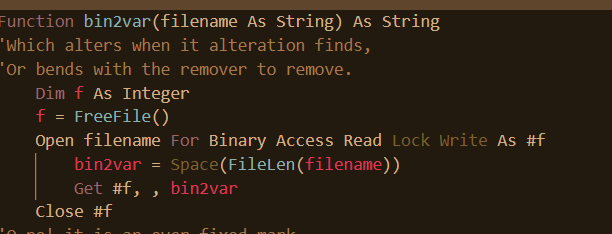
Olevba

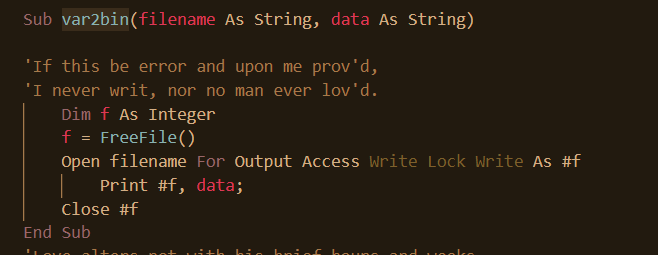


No obfuscation attempt in the code at all



Copies the document to user public user directory, bin2var() function reads the data and gets data , Right() get data from the right side, and save as smile.zip via the var2bin function.

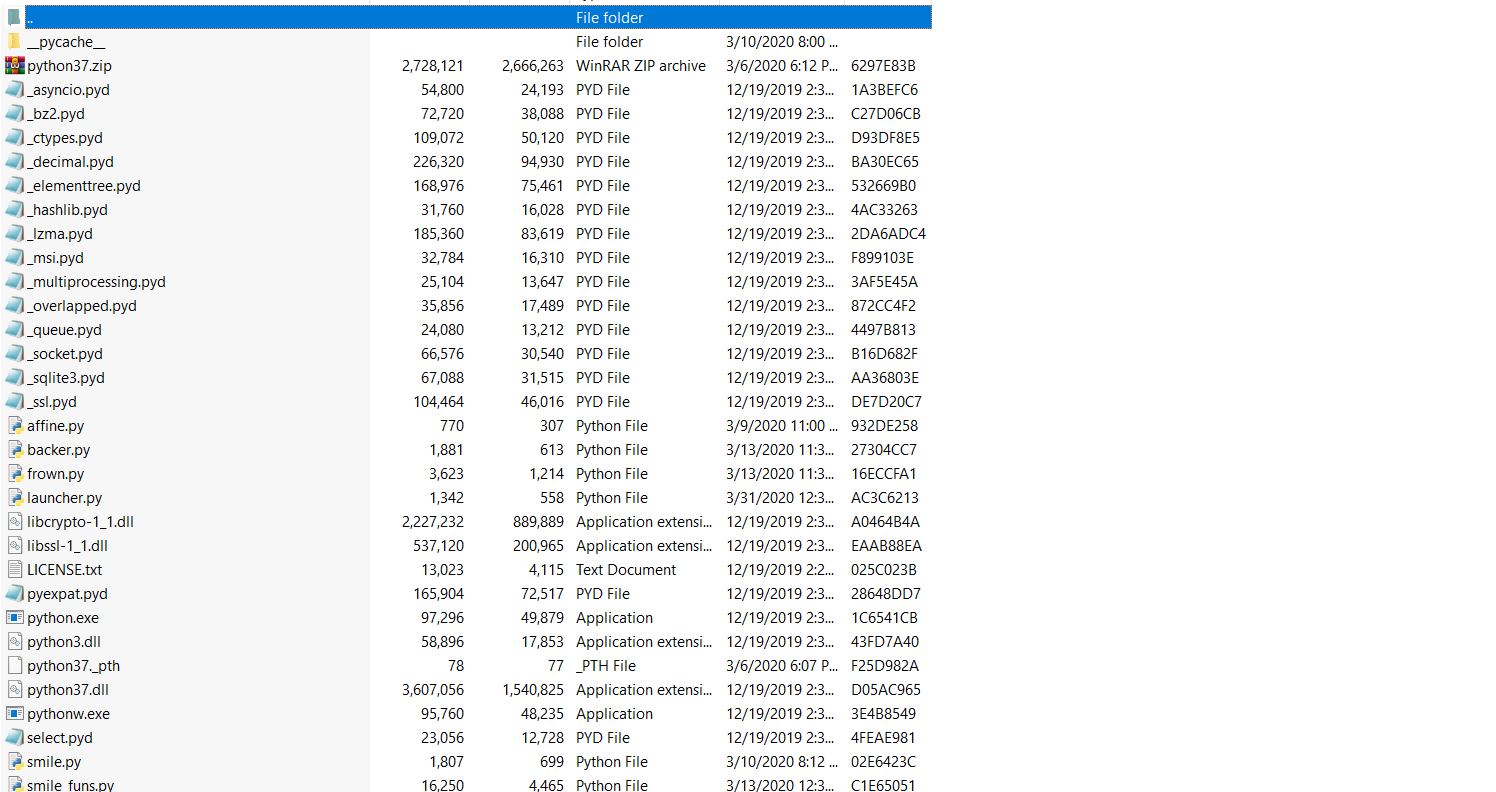




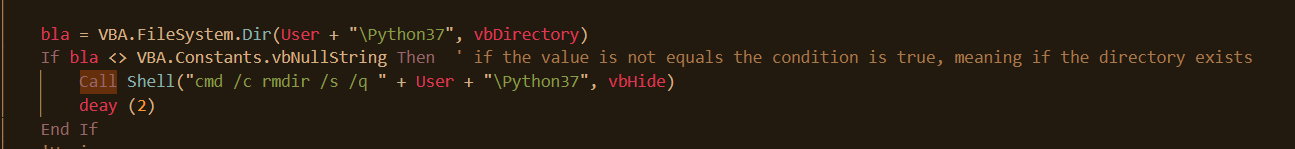
Python code



The zip file contains python itself



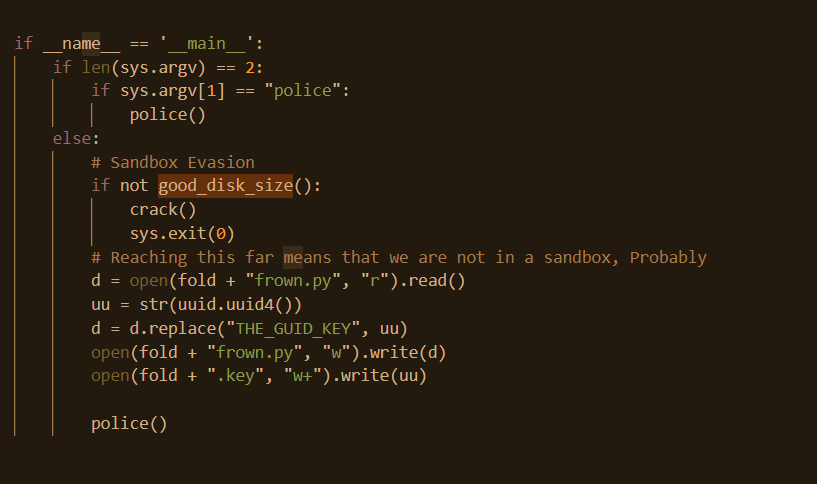
Then checks if python directory exists, if so it deletes



Unzips the new python, and starts the launcher.py

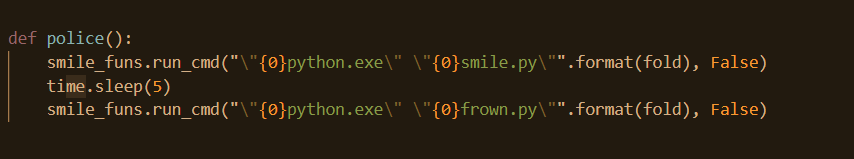


Now lets analyze the python code

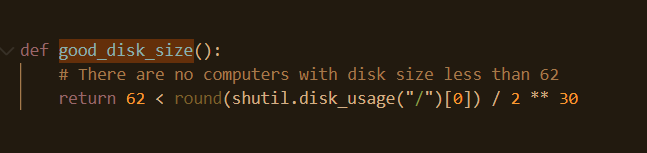


If argument is given which is “police” police function is called

Police() function calls method form another file, and possibly tries to launch simle.py and frown.py

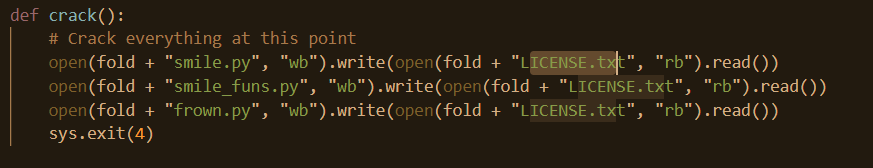


If police argument is not provided, then good\_disk\_usage() function is called, which checks if the root path “C drive in windows” has total size less then 62 GB



For some reason it does not check free size, rather total size

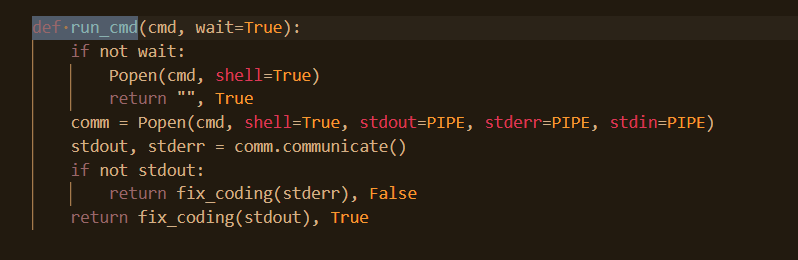
If the size is small, then crack() function is called,



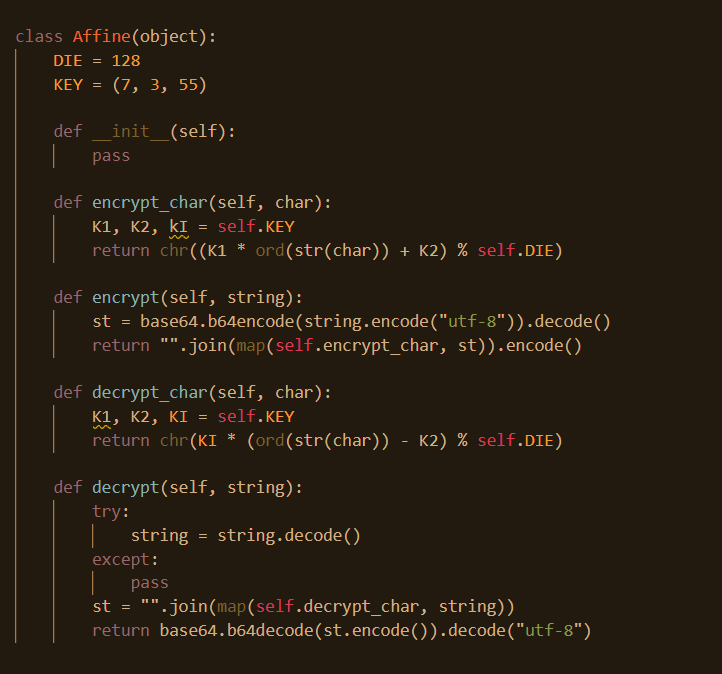
in which case smile.py, smile\_funs.py and frown.py files are open, and its contents are replace by Licesnse.txt contents

If the space is enough however, the script generates a random, uuid, then replaces the string “THE\_GUID\_KEY” with the random uuid in frown.py and also stores the UUID in “.key” file and finally calls police() function.

The run\_cmd() function just runs the command in shell



The main function in smile calls communicate() function, it creates an Affine() object, the Affine class is user define and used for encryption and decryption



The encryption and decryption function is pretty simple

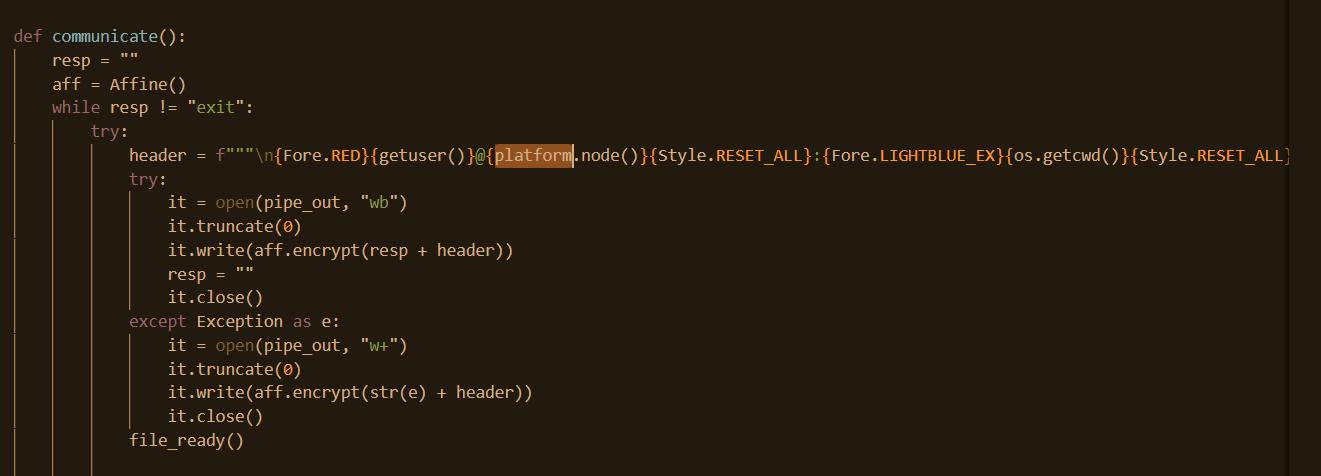
First get utf-8 encoded characters, then base64 encode them, then on each character

Get ascii value, multiply by 7, add 3 and the result of that is done modulo with 128 and finally the ascii character is returned.

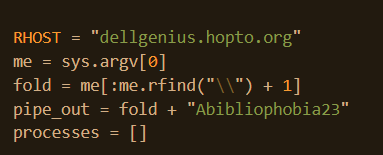
In decryption

On each string character get ascii value, multiply by 55, minus 3 and modulo wutg 128, and finaly decode base 64

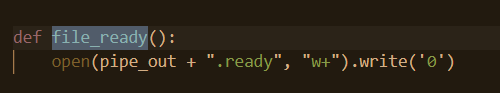
The main function of communicate() seems to cnc communication



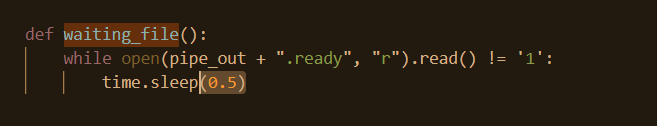
Gets some info about the system like current directory, username and so on, encrypts it and writes in pipe\_out file, which is declared in smile\_funs file i.e filename “Abibliophobia23”



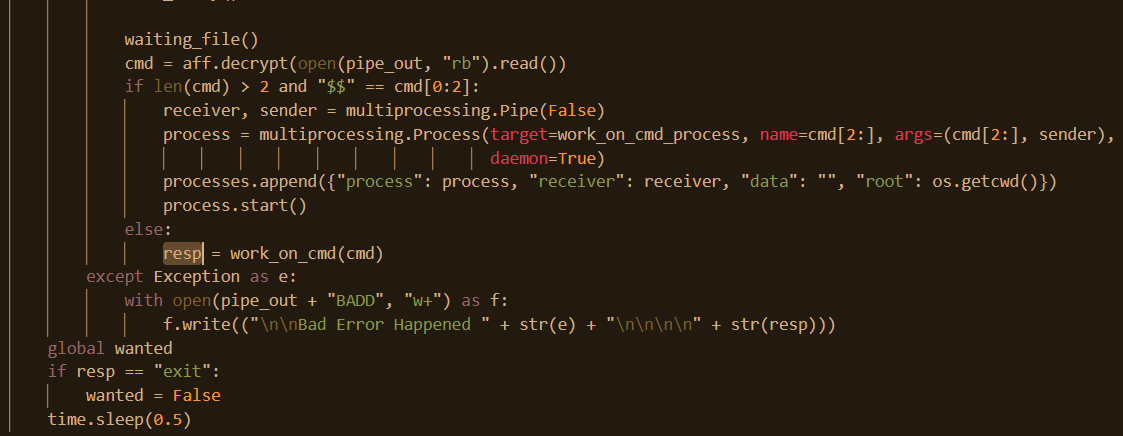
And calls file\_read() file



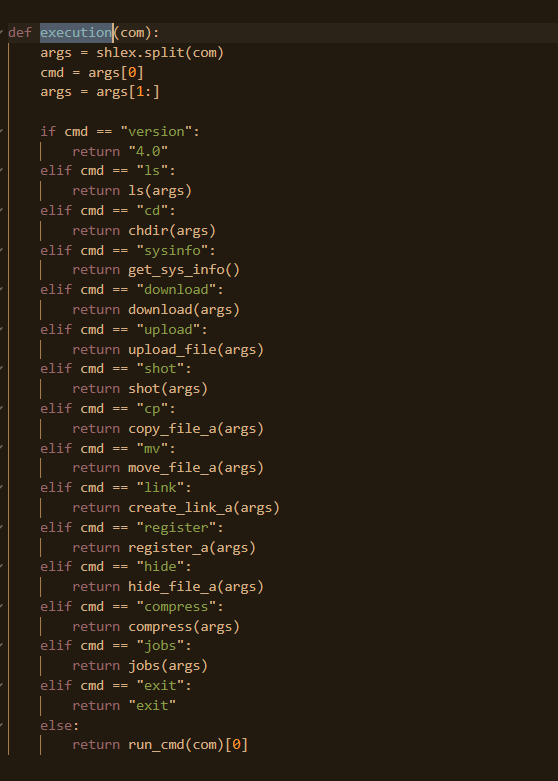
It just writes to file “Abibliophobia23.ready” ‘0’

And calls wainting\_file 

Until “Abibliophobia23.ready” file does no have ‘1’ sleep, keep looping.



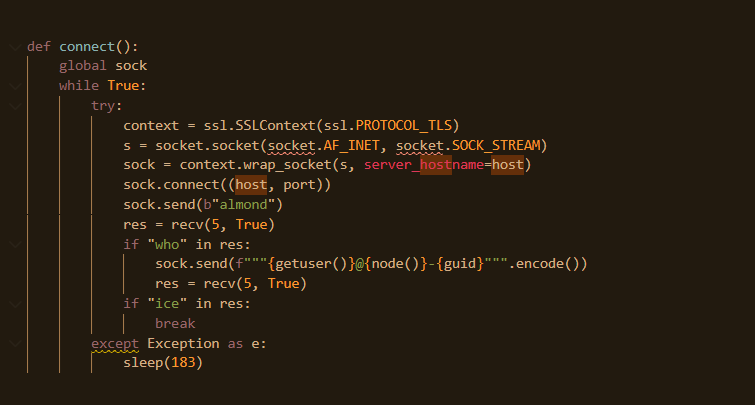
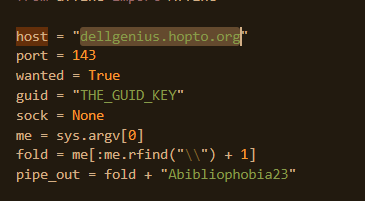
It then reads from the Abibliophobia23 file, decrypts it, and runs it via multiprocessing worn\_on\_cmd\_process or work\_cmd\_process. Both of them eventually calls the execution() function in smile\_funs.py



Seems to be a list of predefined commands with their respective functions

If the communicate() function received “exit”, we exit

Now where is the communication established? That is in frown.py. Function connect() is used to connect to cnc server “dellgenius.hopto.org” port 143

After that communicate function is called



It reads from the file Abibliophobia23, send and reicievs data form server, and encrypts and decrypts

IOC dellgenius.hopto.org