ObfusQrypt

This tool is used to upgrade your existing python malware code, making them more resilient to forensics attempts at them. There are multiple features in this tool and they can be used together or separately depending on your preference.

1. Code Embedder

The **Code Embedder** tab offers 3 modes to inject functionalities into Python scripts.

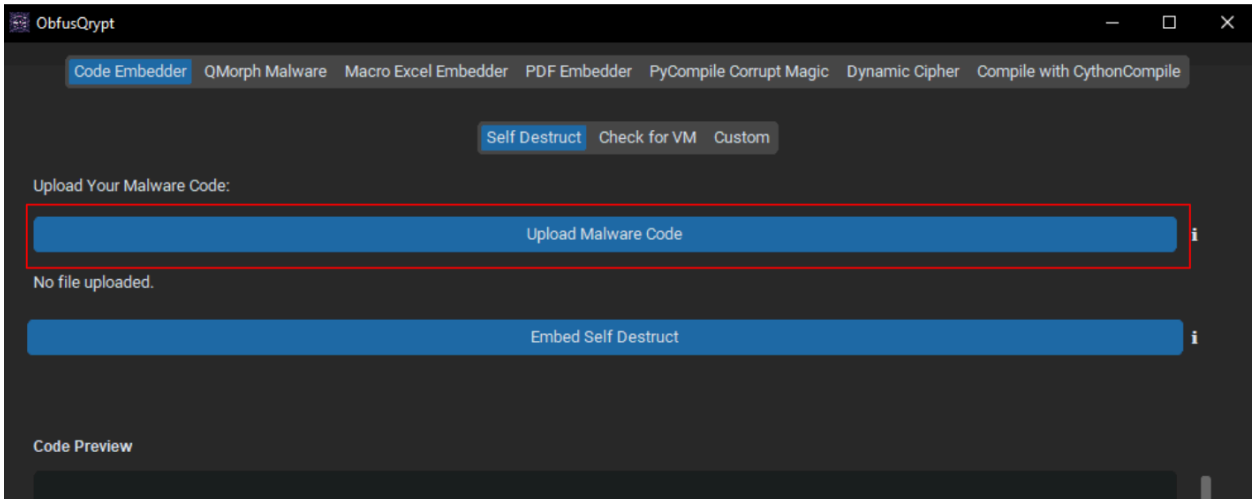
1. Self-Destruct Mode
2. VM Evasion Mode
3. Custom Mode

Self-Destruct

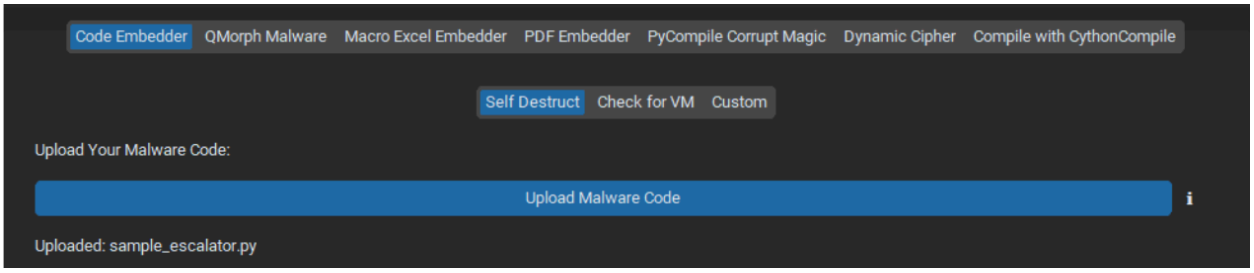
The Self-Destruct sub-tab embeds a function to the end of the main function to delete the script once the script is terminated.

Usage

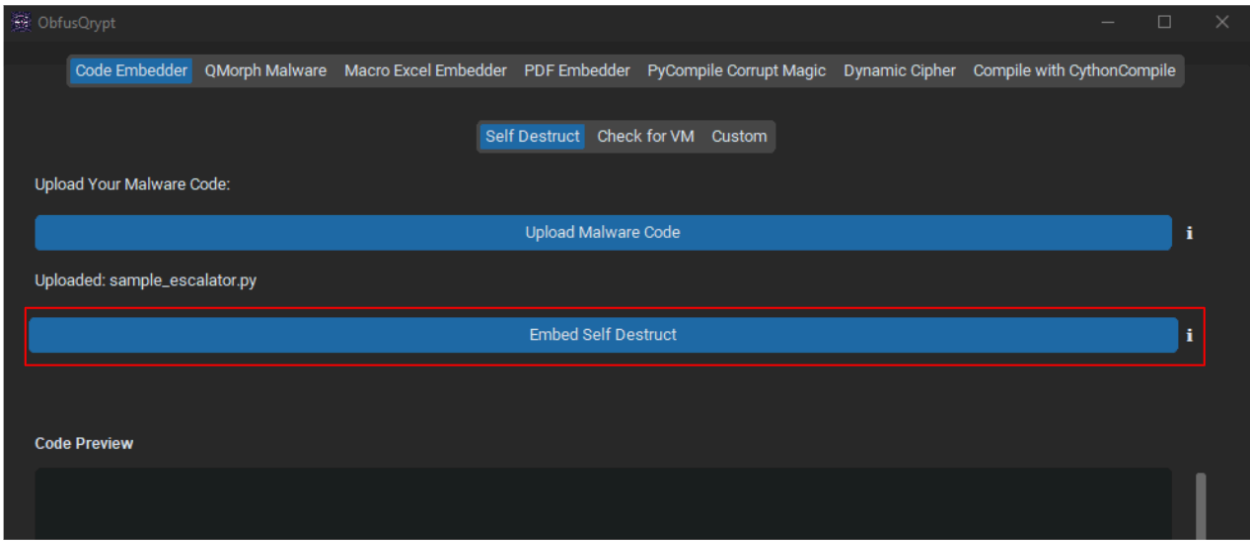
1. **Upload Code**: Click Upload Malware Code.



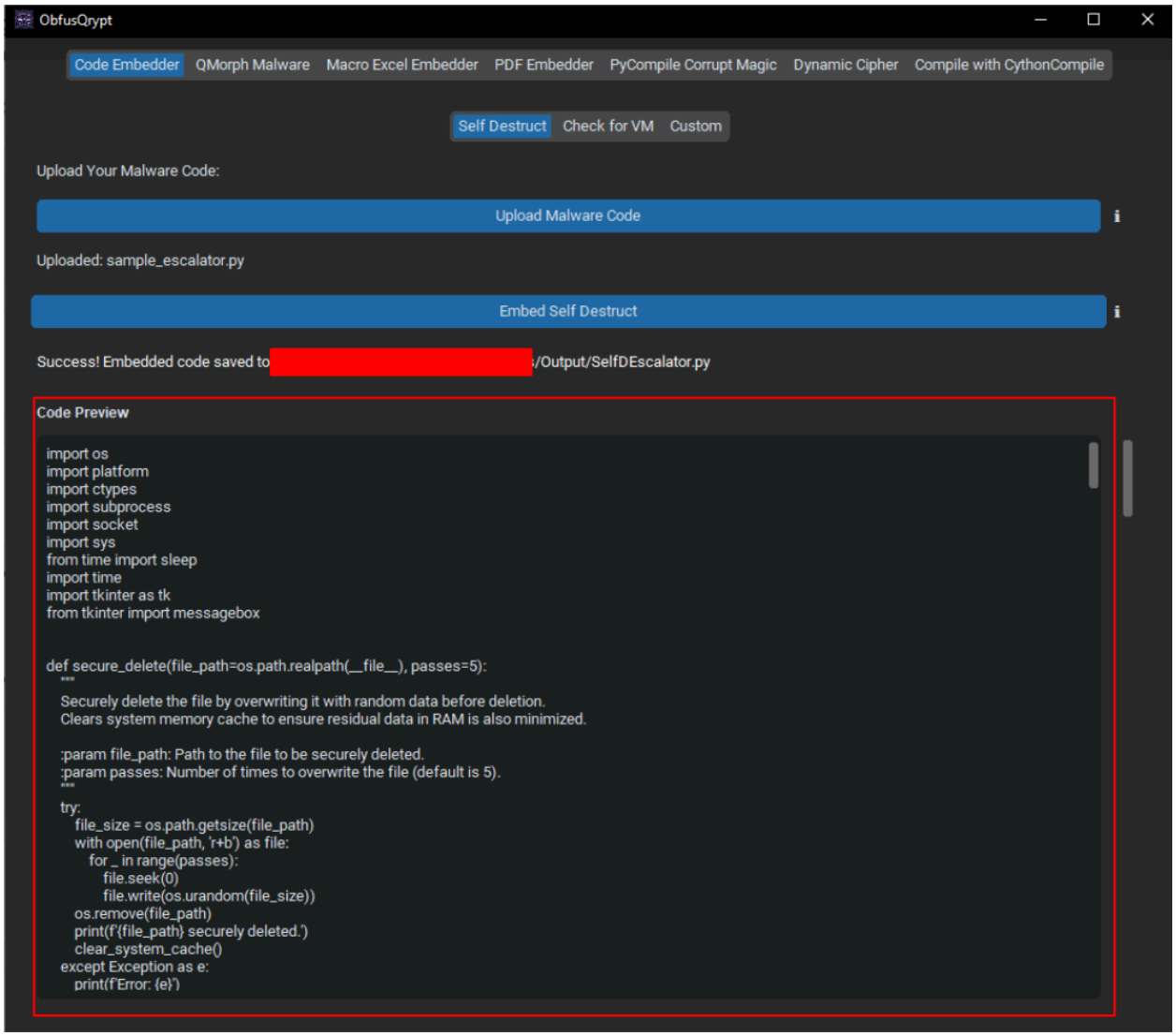
Once Uploaded the name of the code will be shown at the bottom:



2. **Embed Self Destruct**: Click **Embed Self Destruct** and choose a filename for your result code and .



3. **Successful**: The preview box at the bottom will display the modified code, while the success message will show right above it.

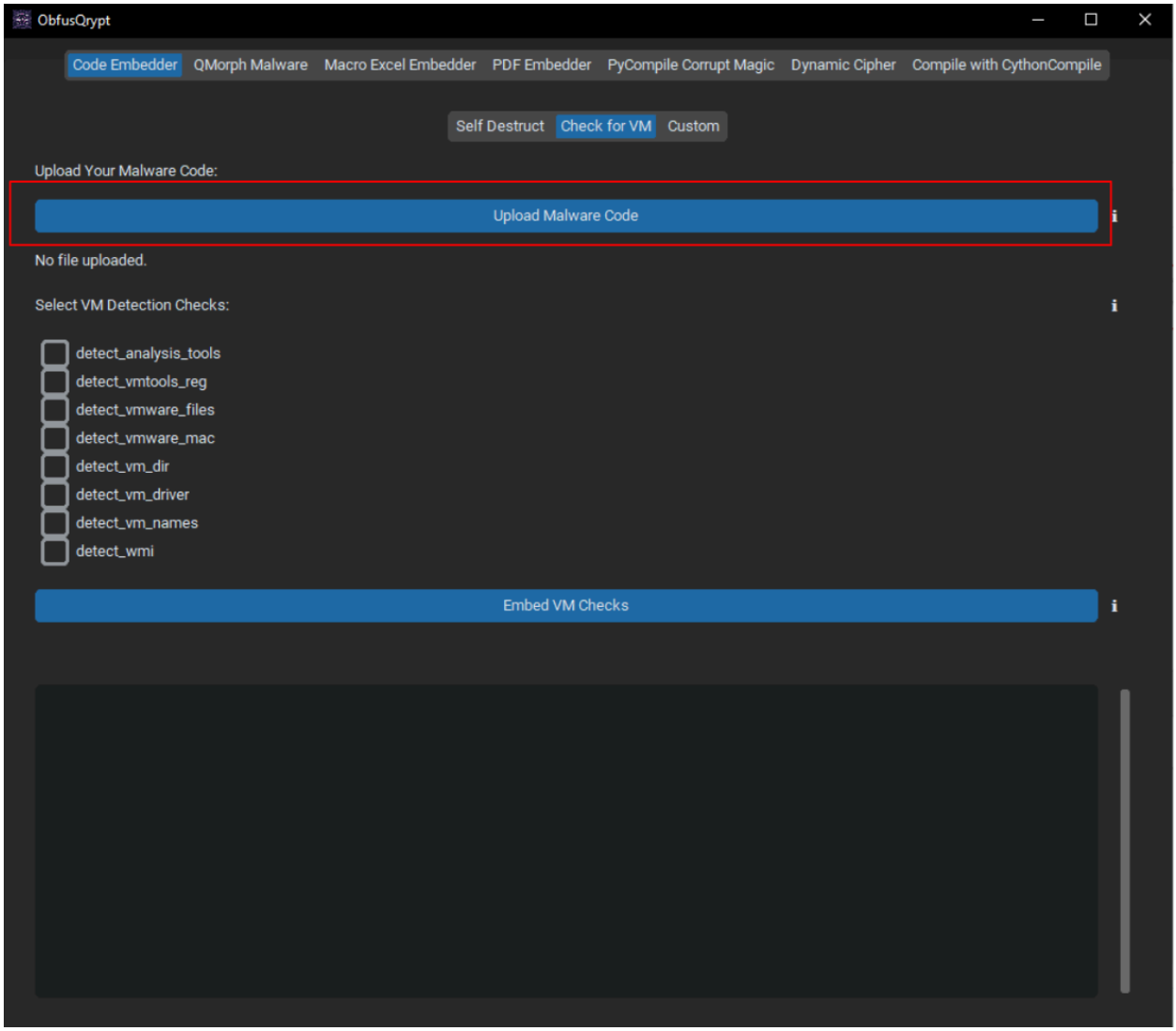


VM Evasion

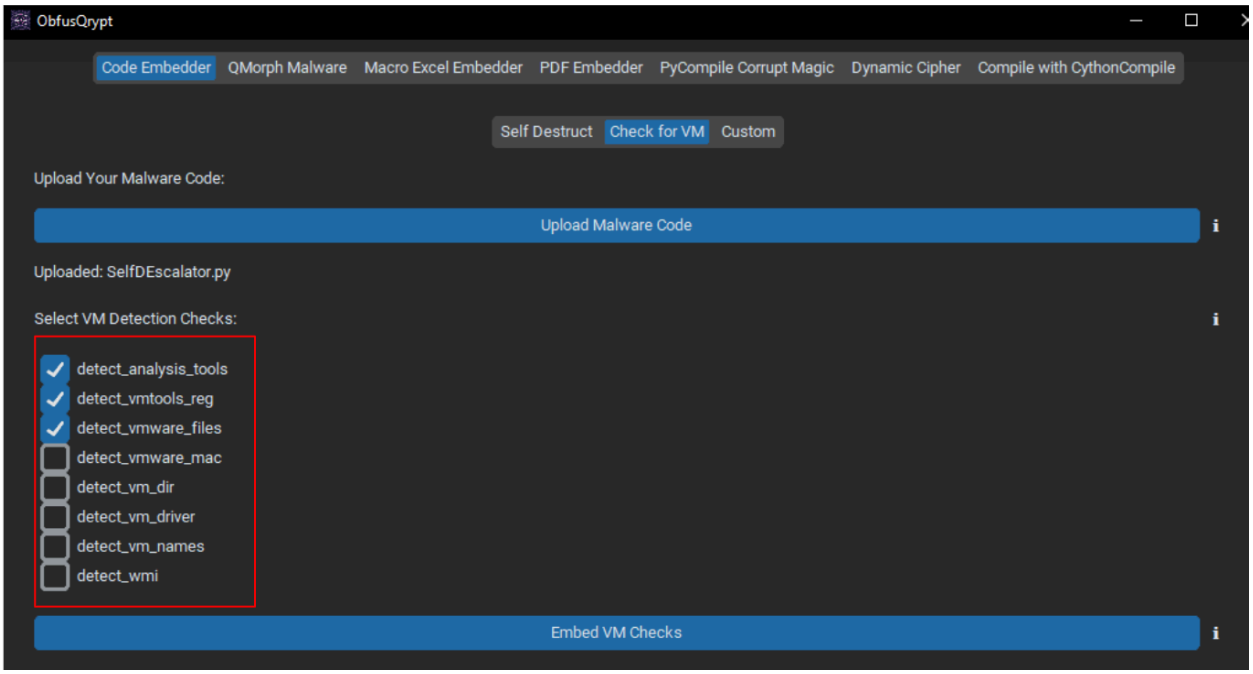
The **Check for VM** sub-tab allows you to add Virtual Machine detection code into your python code, and trigger an exit
if the code is running within a virtual environment, which can trigger an exit or alternate behavior.

Usage

1. **Upload Code**: Select your code file by clicking **Upload Malware Code**.

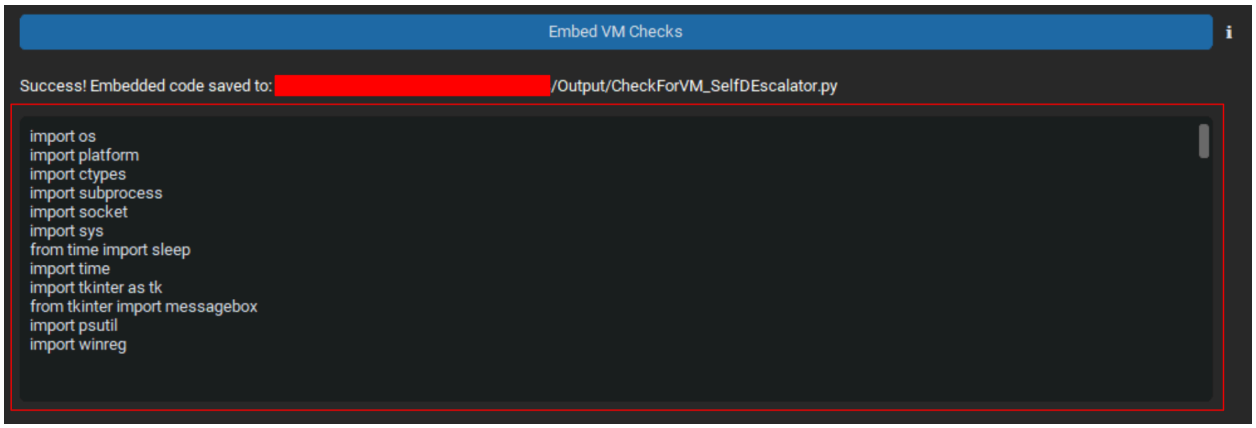


2. **Select VM Detections to check**: Choose desired VM detection options by clicking on the check boxes.



3. **Embed VM Checks**: Click **Embed VM Checks** and give it a new name.

4. **Display output**: Once you click save, the Success message displays the location of your output code, and the Result code.

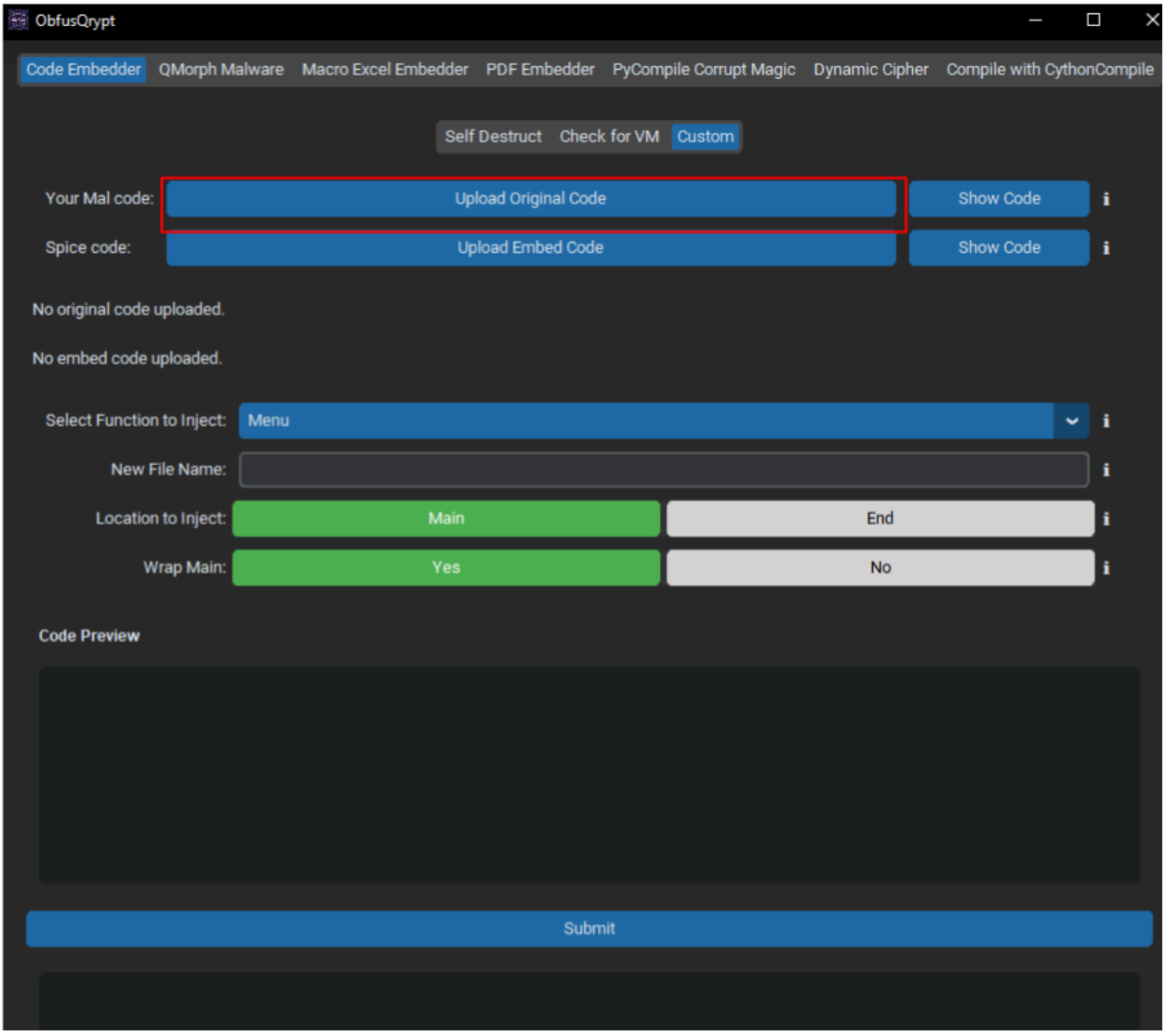


Custom Feature

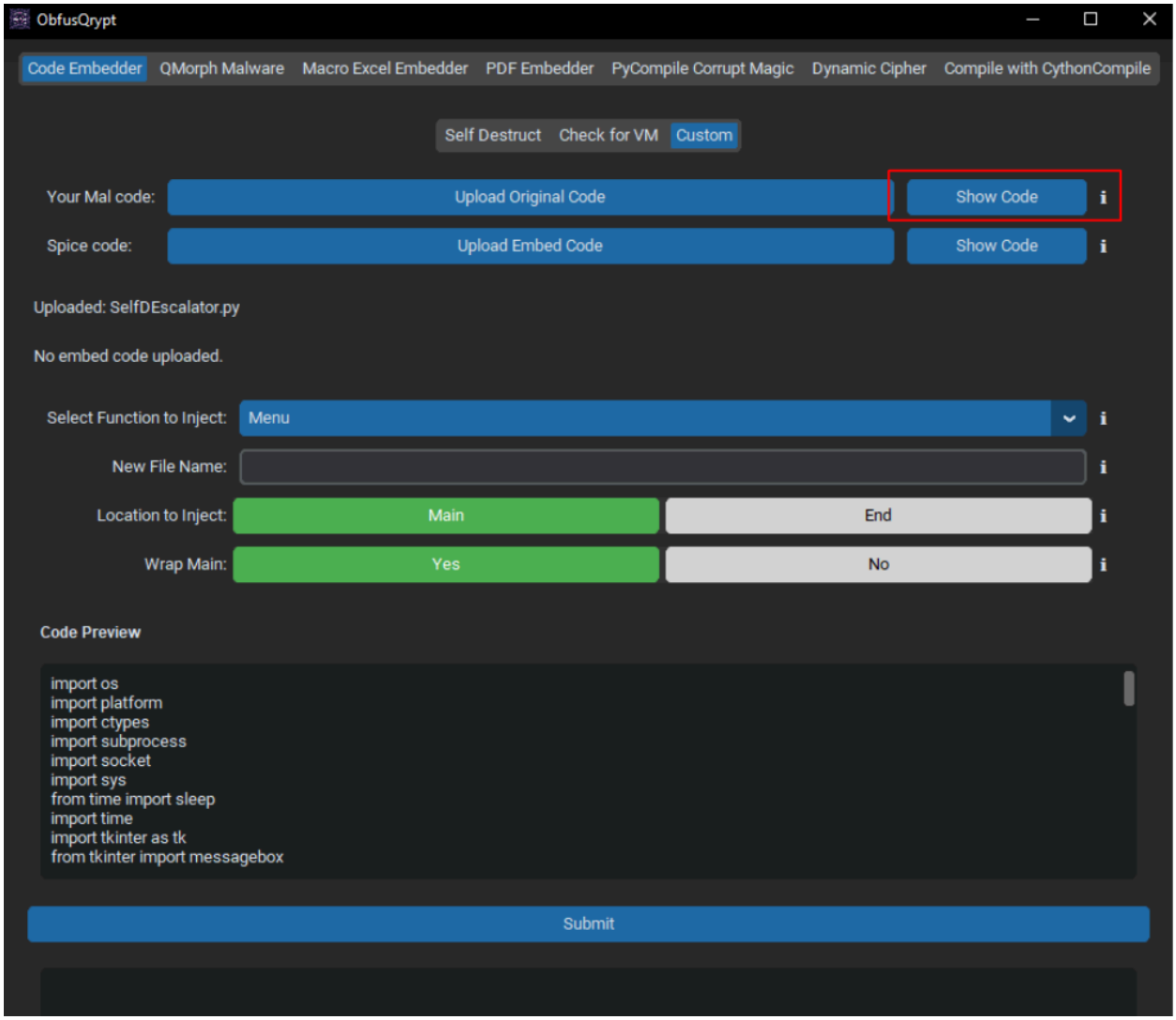
The **Custom** sub-tab allows you to embed a custom function from your own python file into any of your main malicious Python script.

Usage

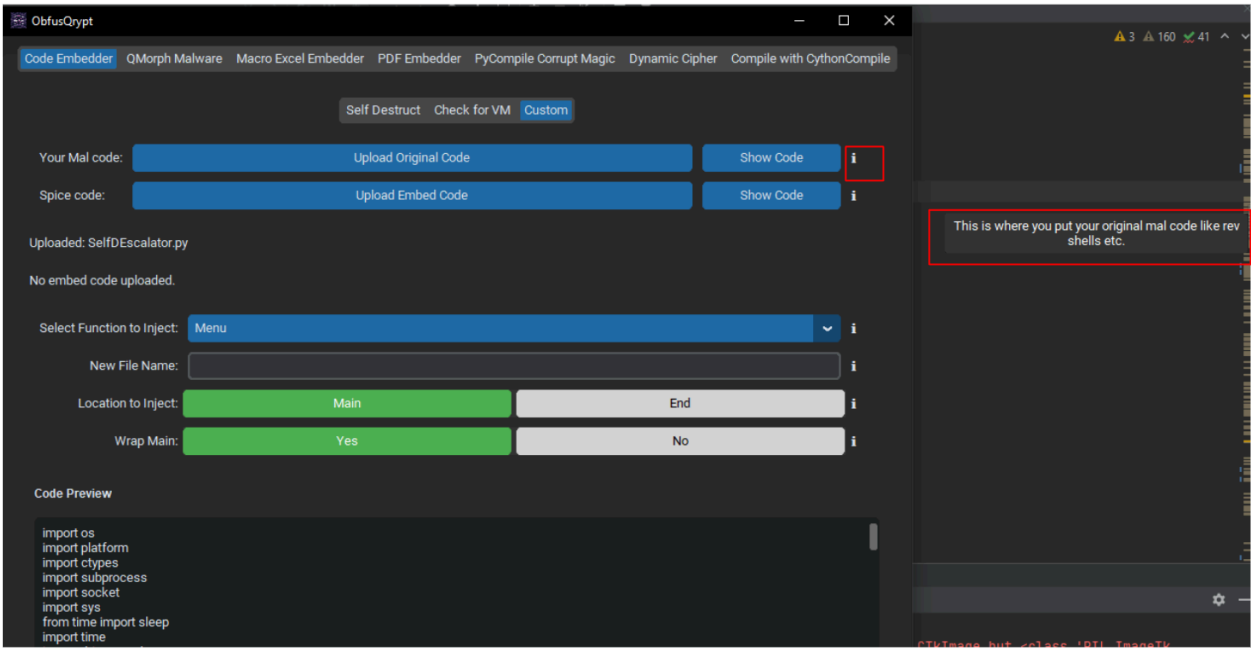
1. **Upload Files**: Use **Upload Original Code** and **Upload Embed Code**.



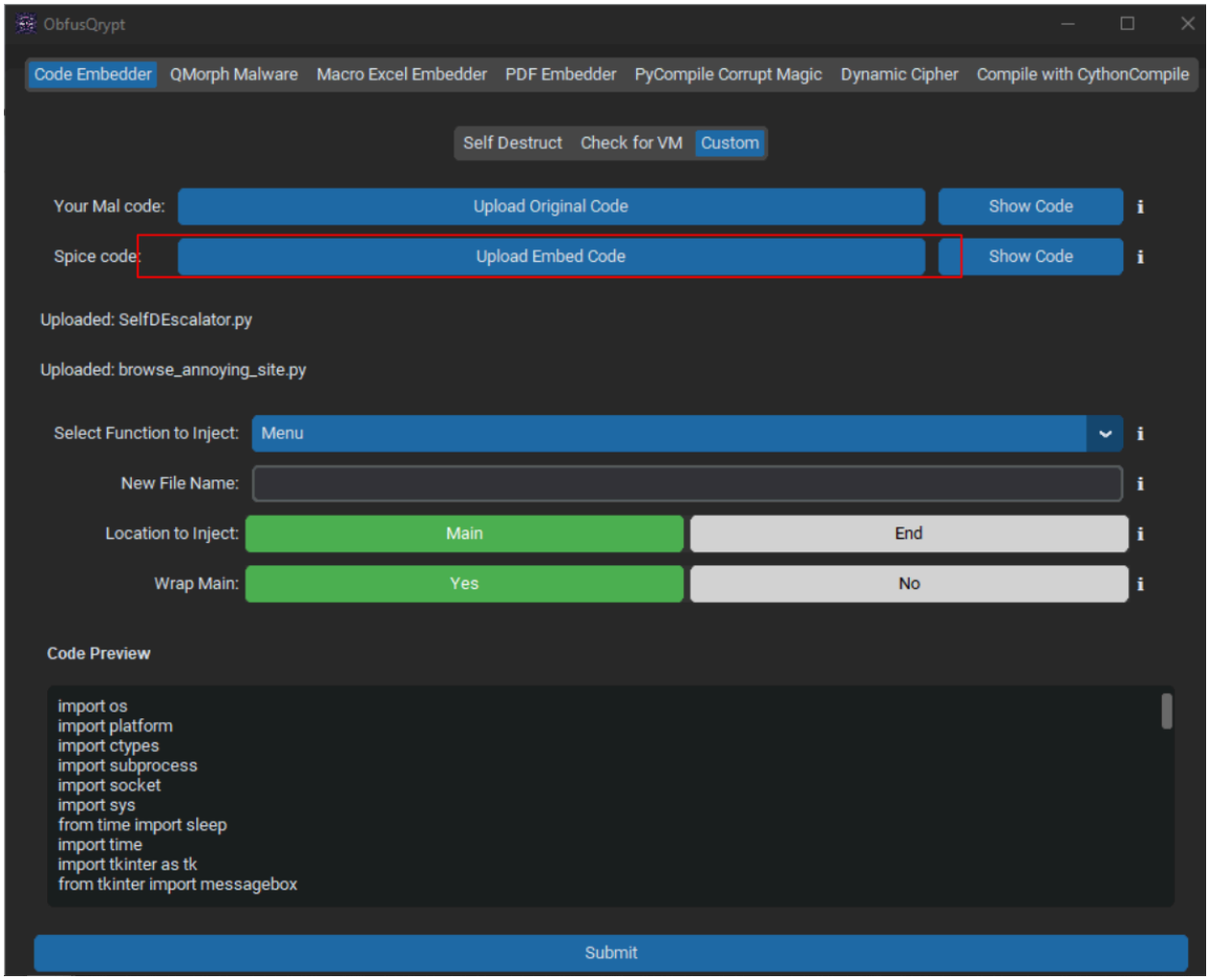
2. **View Code Preview:** Click the **Show Code** button next to the upload button to view code in the code preview pane


ObfusQrypt
Code Embedder
QMorph Malware
Macro Excel Embedder
PDF Embedder
PyCompile Corrupt Magic
Dynamic Cipher
Compile with CythonCompile
Self Destruct
Check for VM
Custom
Your Mal code:
Upload Original Code
Show Code
Spice code:
Upload Embed Code
Show Code
Uploaded: SelfDEscalator.py
No embed code uploaded.
Select Function to Inject:
Menu
New File Name:
Location to Inject:
Main
End
Wrap Main:
Yes
No
Code Preview
import os
import platform
import ctypes
import subprocess
import socket
import sys
from time import sleep
import time
import tkinter as tk
from tkinter import messagebox
Submit

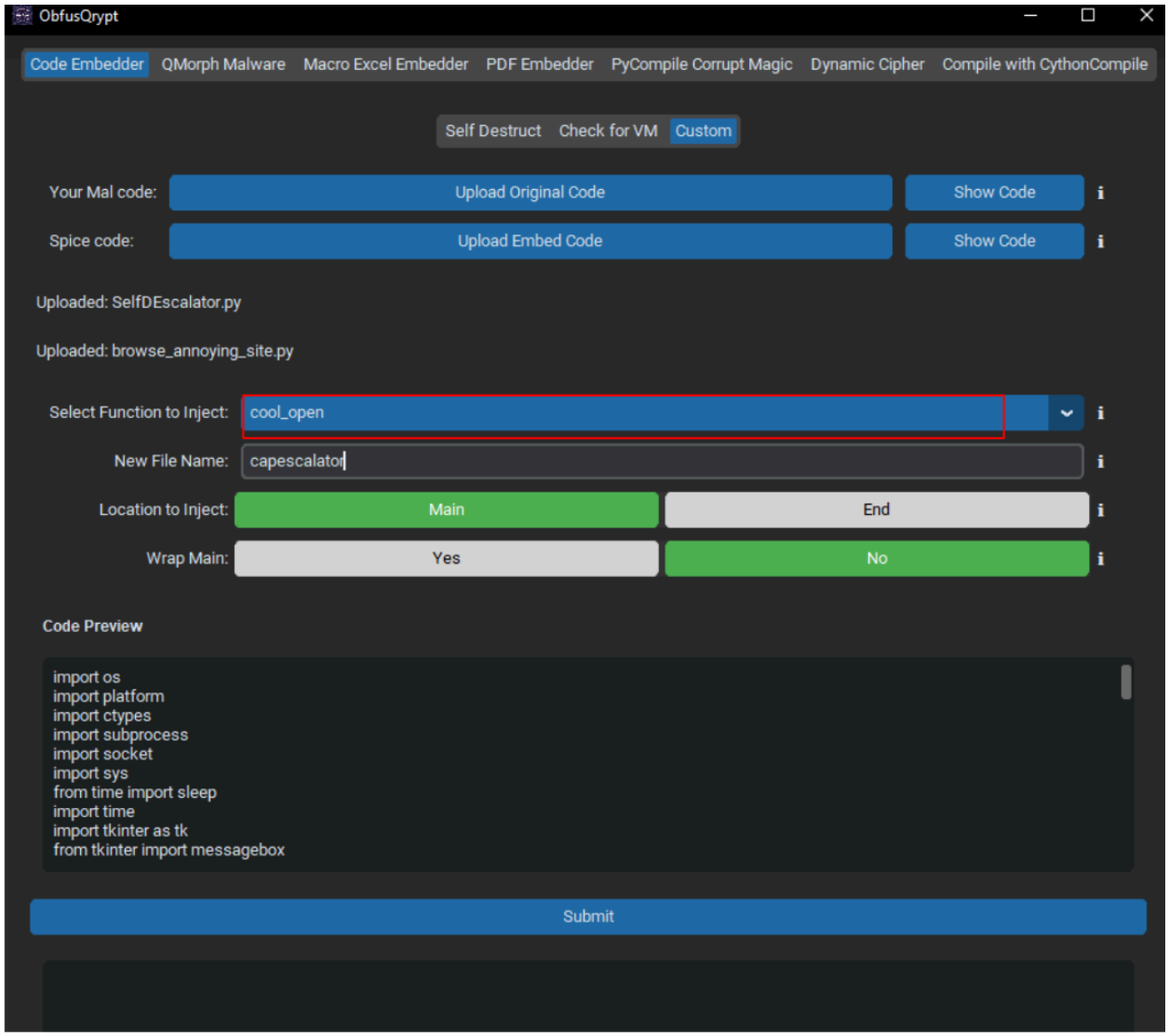
3. **Infokit**: For more information about what the functions do there is an infokit icon at the side which will explain when hovered over.



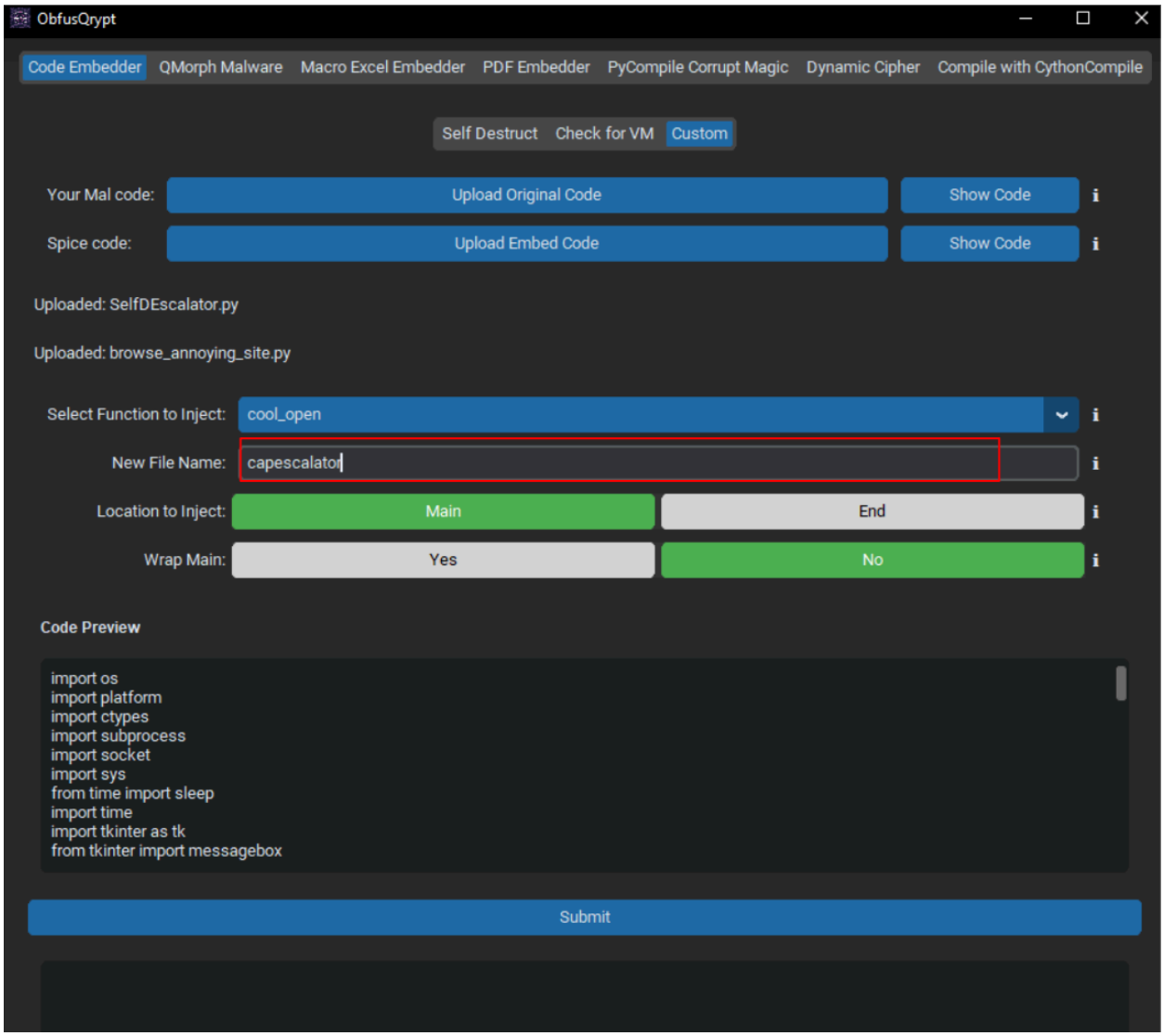
4. **Upload Spice Code:** Select the code to be embedded into your original code.


ObfusQrypt
Code Embedder
QMorph Malware
Macro Excel Embedder
PDF Embedder
PyCompile Corrupt Magic
Dynamic Cipher
Compile with CythonCompile
Self Destruct
Check for VM
Custom
Your Mal code:
Upload Original Code
Show Code
Spice code:
Upload Embed Code
Show Code
Uploaded: SelfDEscalator.py
Uploaded: browse_annoying_site.py
Select Function to Inject:
Menu
New File Name:
Location to Inject:
Main
End
Wrap Main:
Yes
No
Code Preview
import os
import platform
import ctypes
import subprocess
import socket
import sys
from time import sleep
import time
import tkinter as tk
from tkinter import messagebox
Submit

5. **Select Function**: Choose a function from the dropdown after uploading the spice code, this function will be put into main to be triggered as a call function.

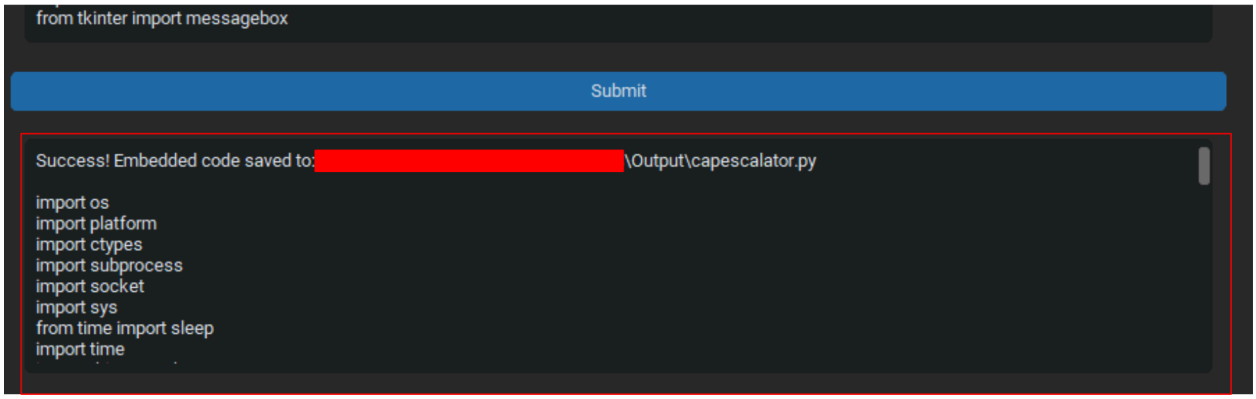


6. **Name the file**: Choose a name for your new file, the file will later be saved to the **Output** folder.



7. **Location to Inject**: Choose location of your **original** to inject the **spice code**. Whether to inject the function at the **before main()** or at the **end** of your code (This is for functions you want to take effect after you are done with your original such as cleanup etc.).
8. **Wrap main**: Choose whether to wrap the function, if yes, the function you are adding will wrap the original main with an if else statement being a condition to.

9. **Embed**: Click **Submit** to finalize.



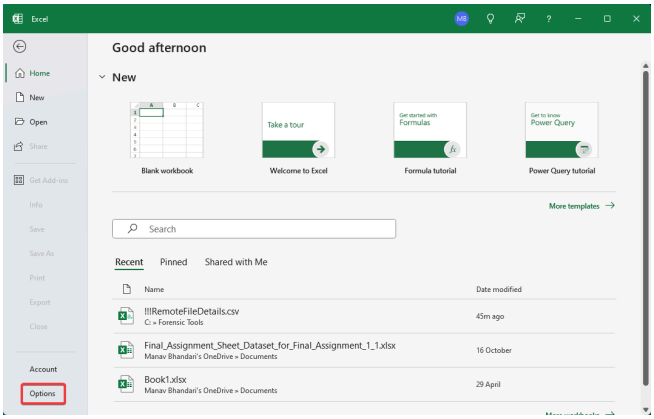
Upon submission it should show the success message, the location it was saved to, and the new code preview.

2. Macro Excel Embedder

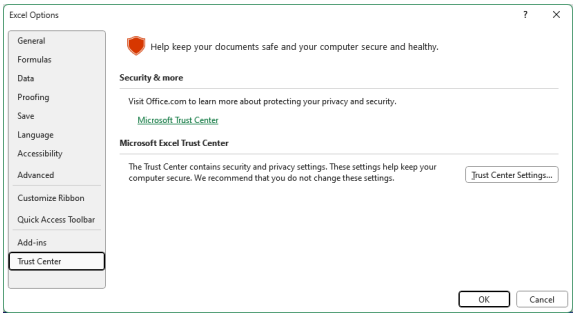
The **Macro Excel Embedder** integrates Python scripts and VBA macros into Excel files.

Usage

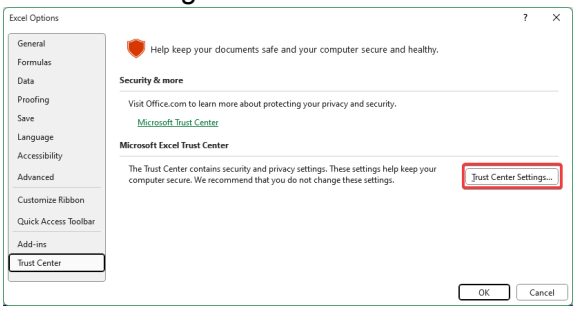
1. **Setup Microsoft Excel Macro settings**:
    a. Open Microsoft Excel and Click on the Options menu



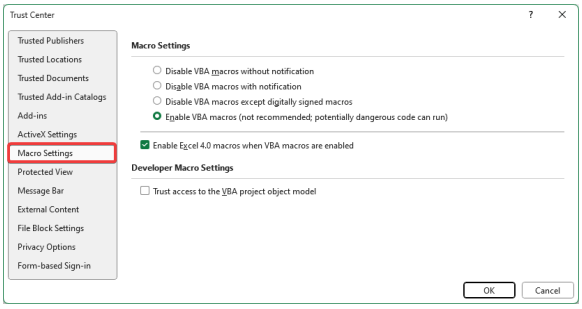
    b. Inside Options, open the Trust Center tab



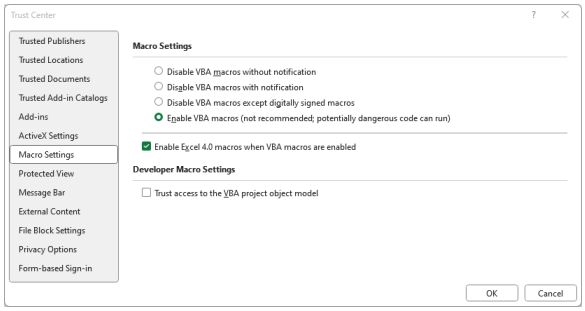
    c. Open Trust Center Settings



    d. Open Macro Settings

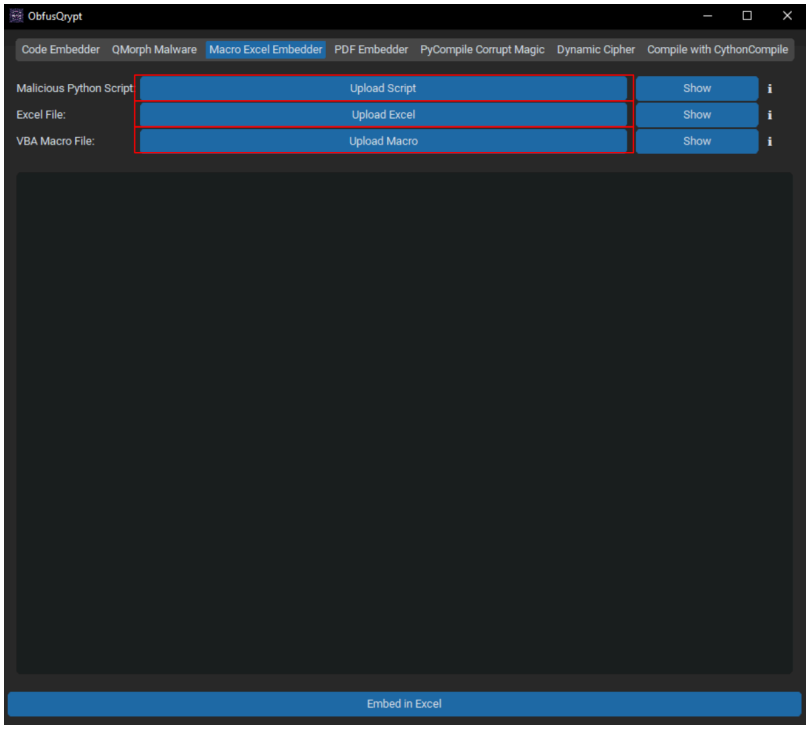


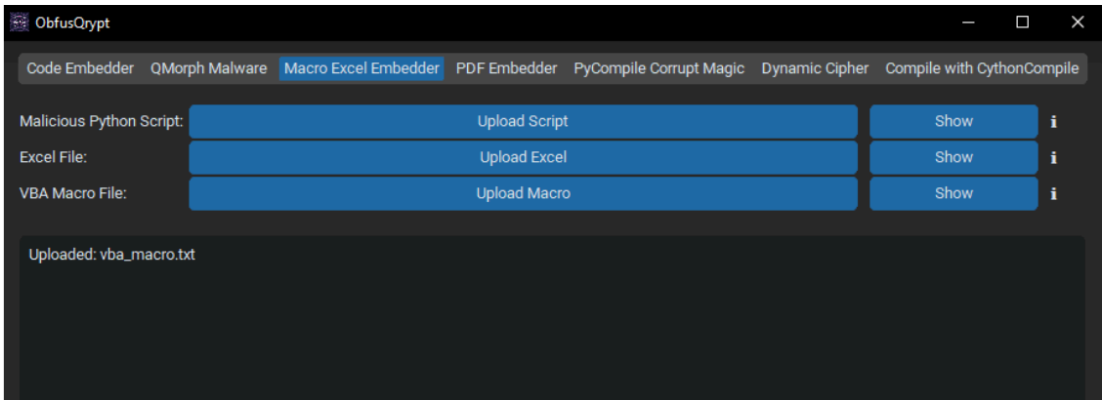
e. Enable VBA macros. Excel 4.0 macros, and Trust access to the VBA project model



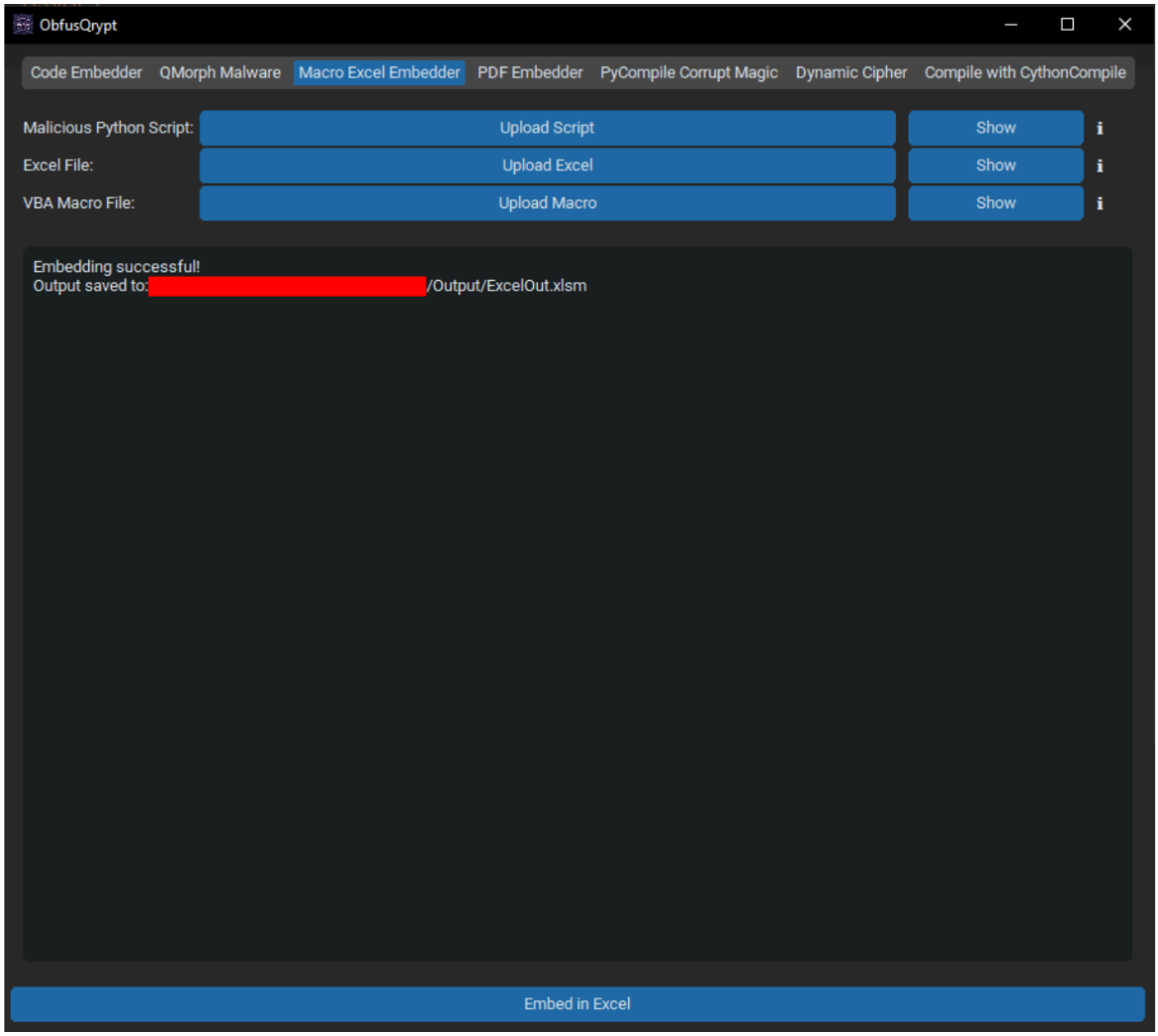
f. Click OK to save all settings.

2. **Upload Files**: Select the Python script, Excel file, and VBA macro.





3. **Embed in Excel**: Press **Embed in Excel** to complete the embedding process, you will be prompted to where you wish to output your file, once you have selected, click save and the file will be outputted to the location specified.



The new file location will be displayed in the preview box, you can navigate to the directory to retrieve it.

3. PDF Embedder

The **PDF Embedder** tab embeds Python malware code into multiple PDF files using RSA encryption. It consists of mainly 2 modes of embedding:

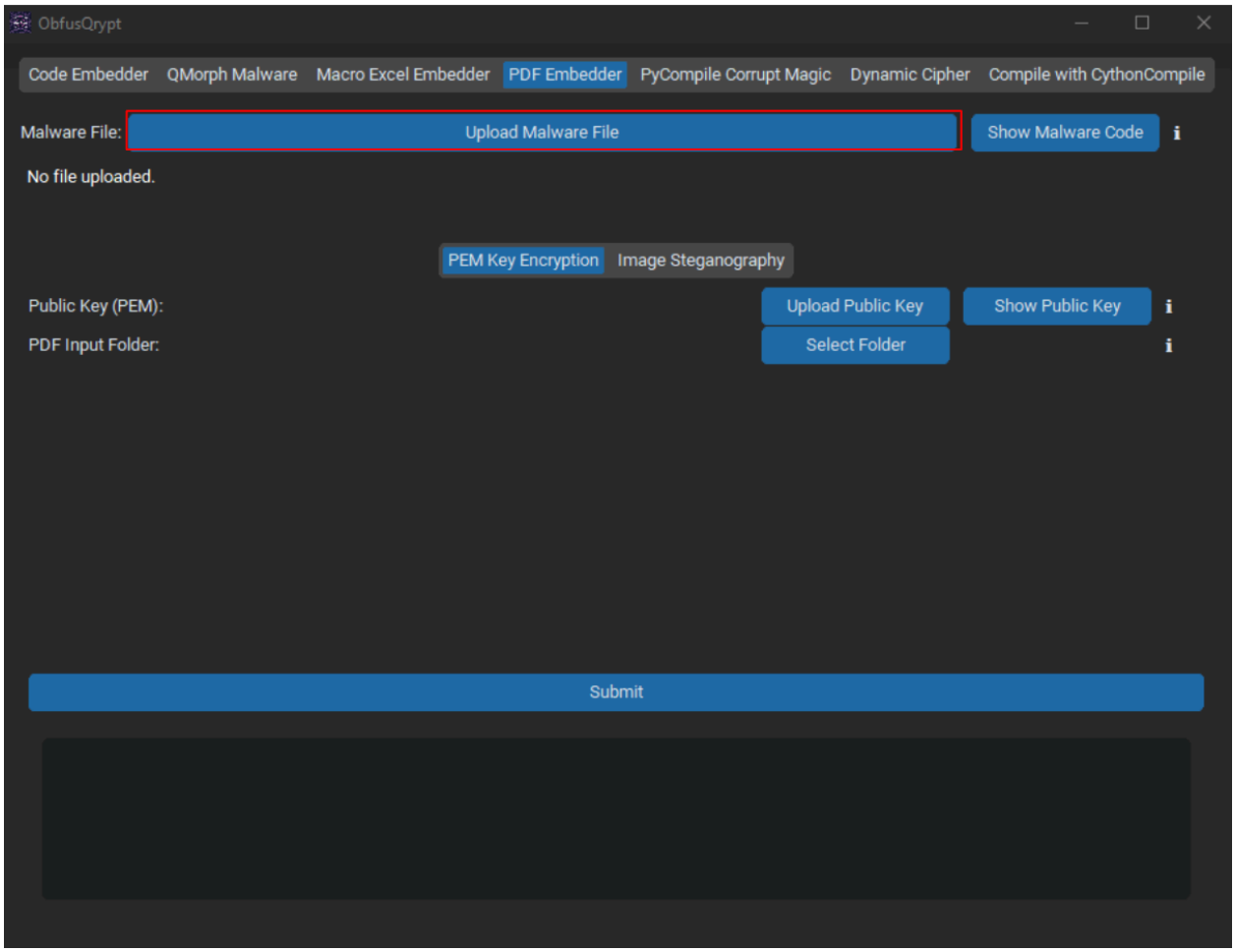
- PEM Key Encryption
- Image Steganography

PEM Key Encryption

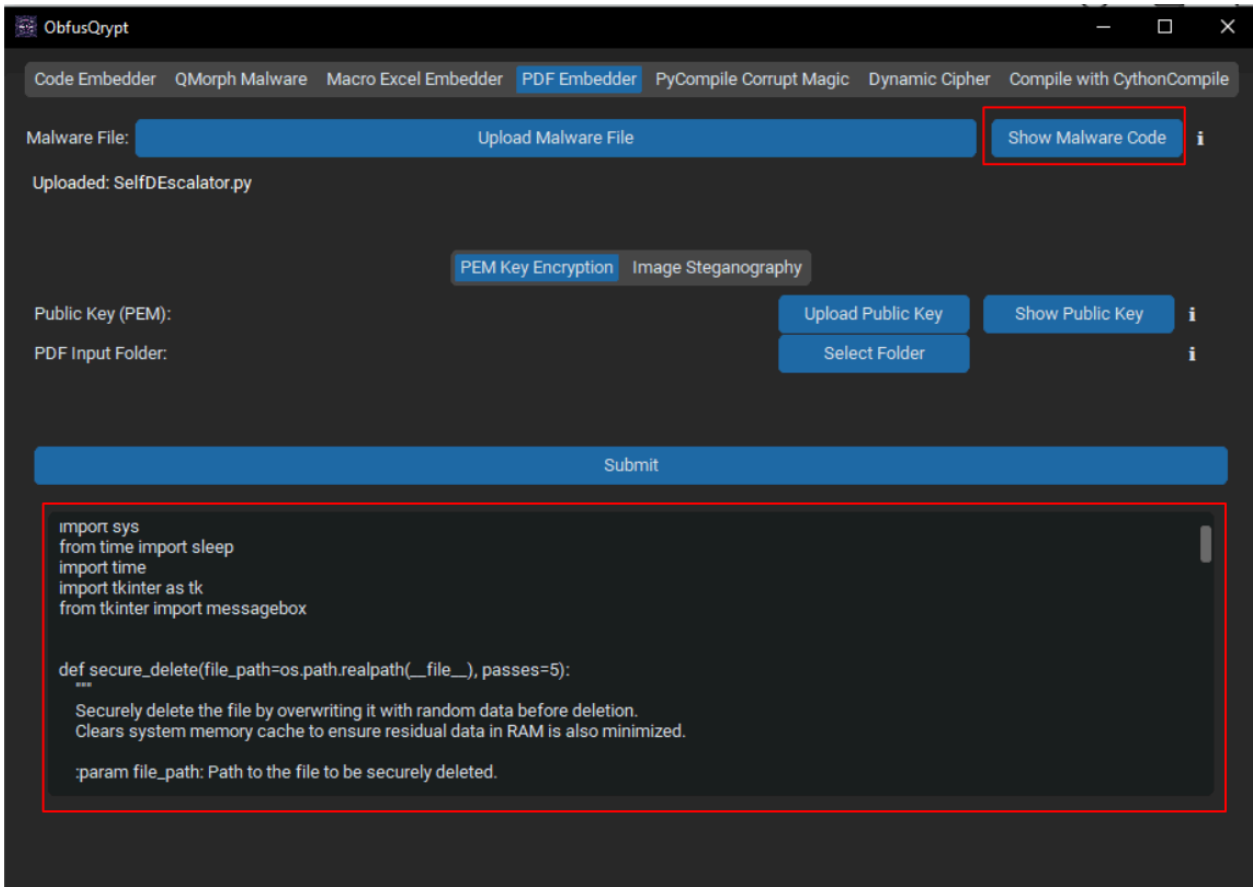
The **PEM Key Encryption** sub-tab allows you to select a PEM format Public key in order to encrypt your Malware File Contents into any of your main malicious Python script.

Usage

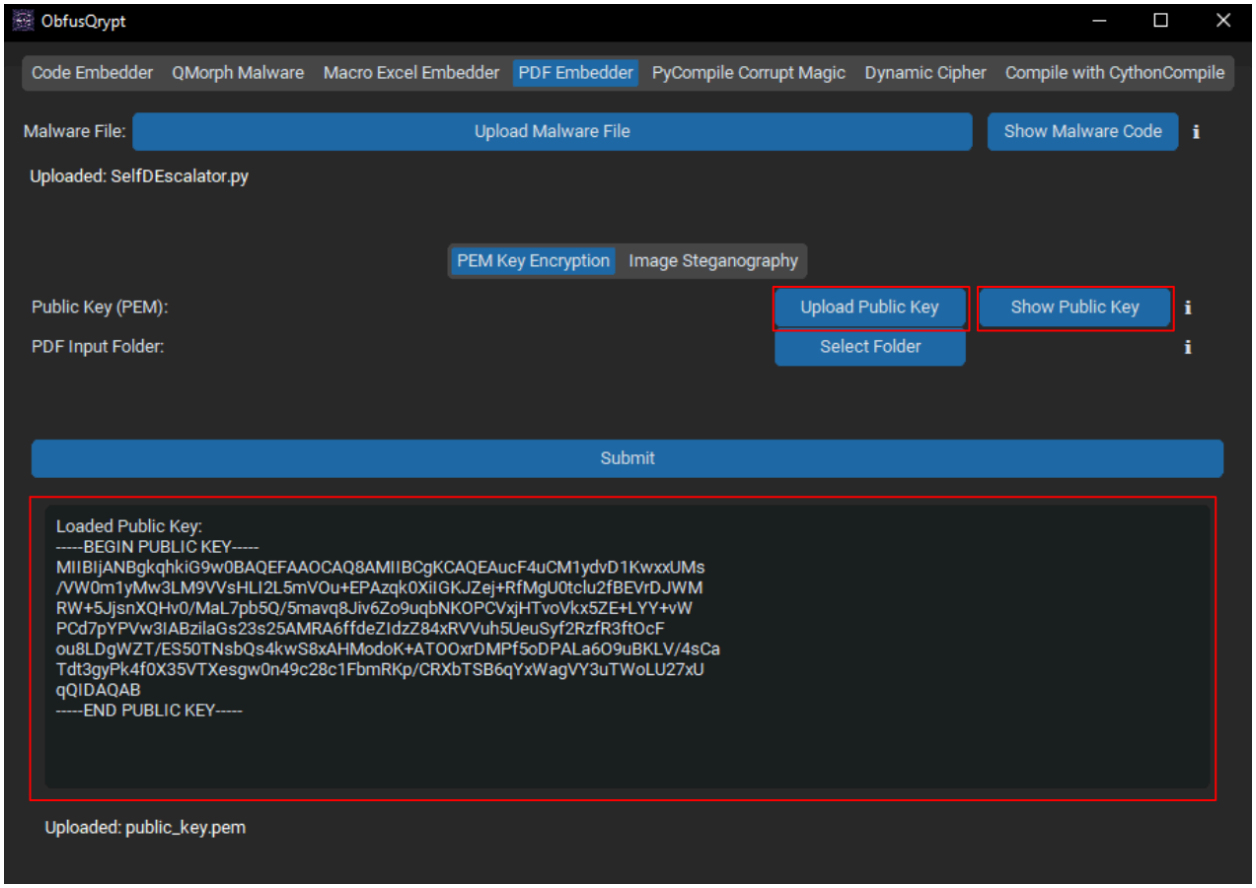
1. **Upload Files**: Upload the malware file, RSA public key, and select PDFs from a folder.



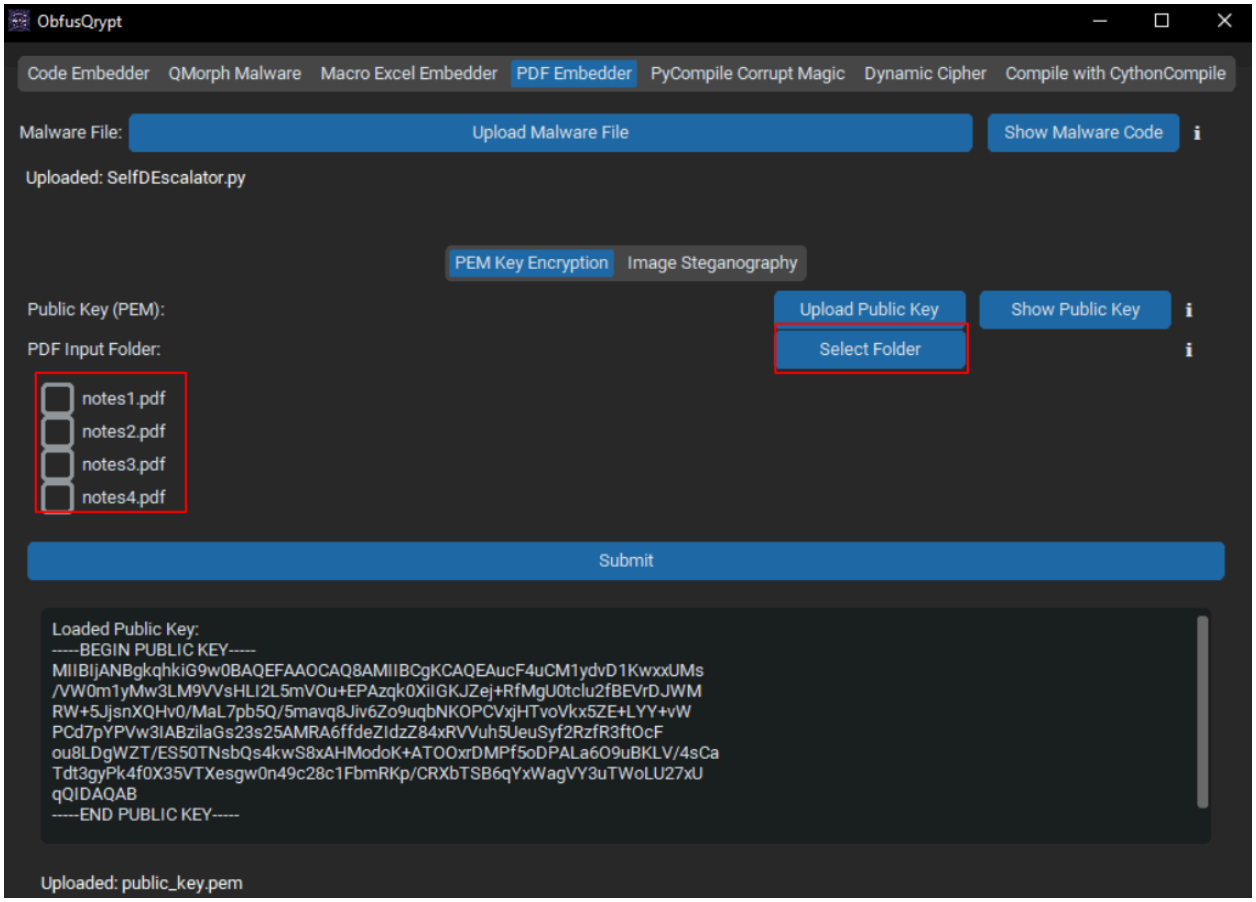
2. **Preview**: Upon upload the code will be displayed below the **submit** button, if another message replaces the preview message, you can click the **Show Malware Code** to re-show the code.



3. **Public key Upload:** PEM format Public key can be uploaded by clicking the **Upload Public Key** button, upon upload, the public key is displayed, if another message replaces the preview message, you can click the **Show Public Key** to re-show the Public key.



4. **Select PDFs**: Click **Select Folder** and navigate to a folder that has PDF files.



If the folder has pdf files, it will be listed in checkboxes for selection

5. **Submit**: Select the PDFs you want to embed your code into and click **Submit** to embed the malware into each selected PDF file. The Preview at the bottom will show the result of the embedding and the location of the output file(s).

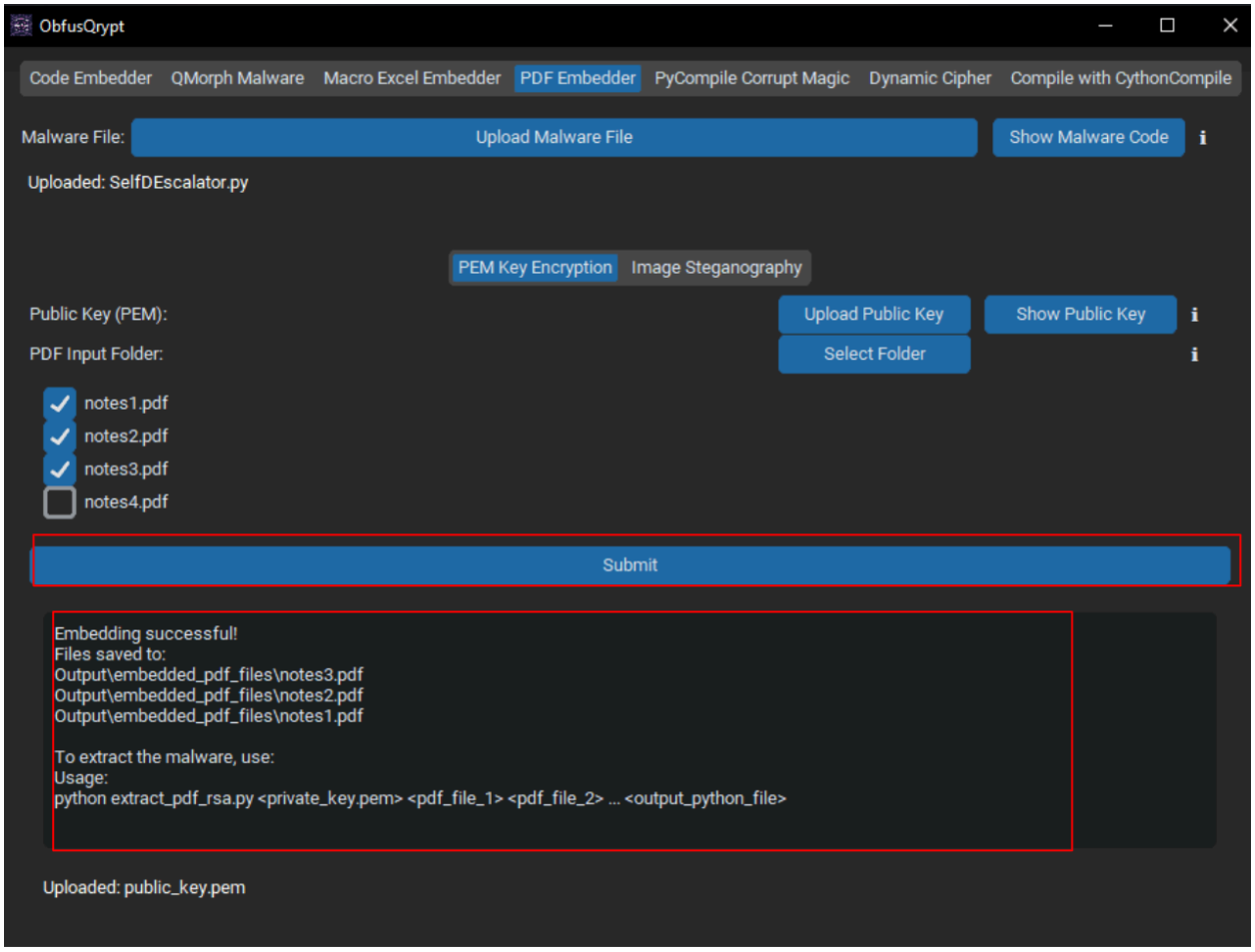
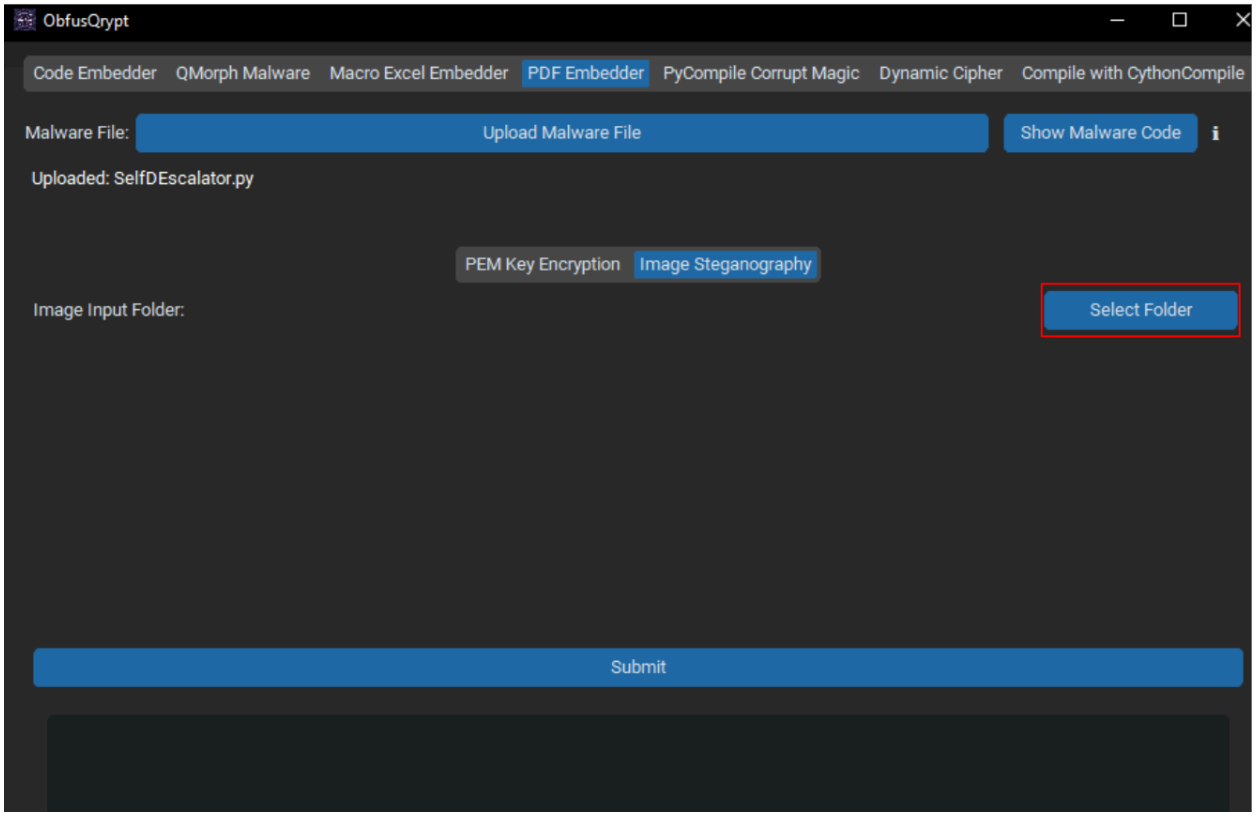

ObfusQrypt
Code Embedder
QMorph Malware
Macro Excel Embedder
PDF Embedder
PyCompile Corrupt Magic
Dynamic Cipher
Compile with CythonCompile
Malware File:
Upload Malware File
Show Malware Code
Uploaded: SelfDEscalator.py
PEM Key Encryption
Image Steganography
Public Key (PEM):
Upload Public Key
Show Public Key
PDF Input Folder:
Select Folder
notes1.pdf
notes2.pdf
notes3.pdf
notes4.pdf
Submit
Embedding successful!
Files saved to:
Output\embedded_pdf_files\notes3.pdf
Output\embedded_pdf_files\notes2.pdf
Output\embedded_pdf_files\notes1.pdf
To extract the malware, use:
Usage:
python extract_pdf_rsa.py <private_key.pem> <pdf_file_1> <pdf_file_2> ... <output_python_file>
Uploaded: public_key.pem

Image Steganography

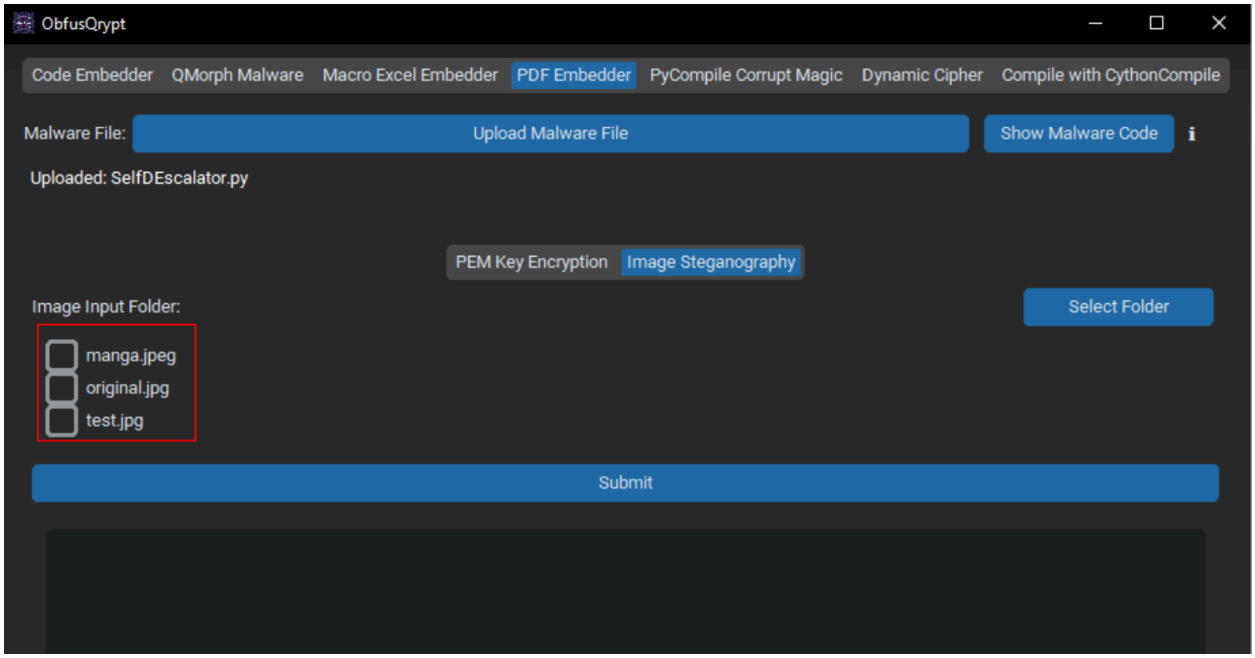
The **Image Steganography** sub-tab allows you to select **Images** in order to perform code embedding of your Malware into an image which would be stored into a pdf for later extraction and use.

Usage

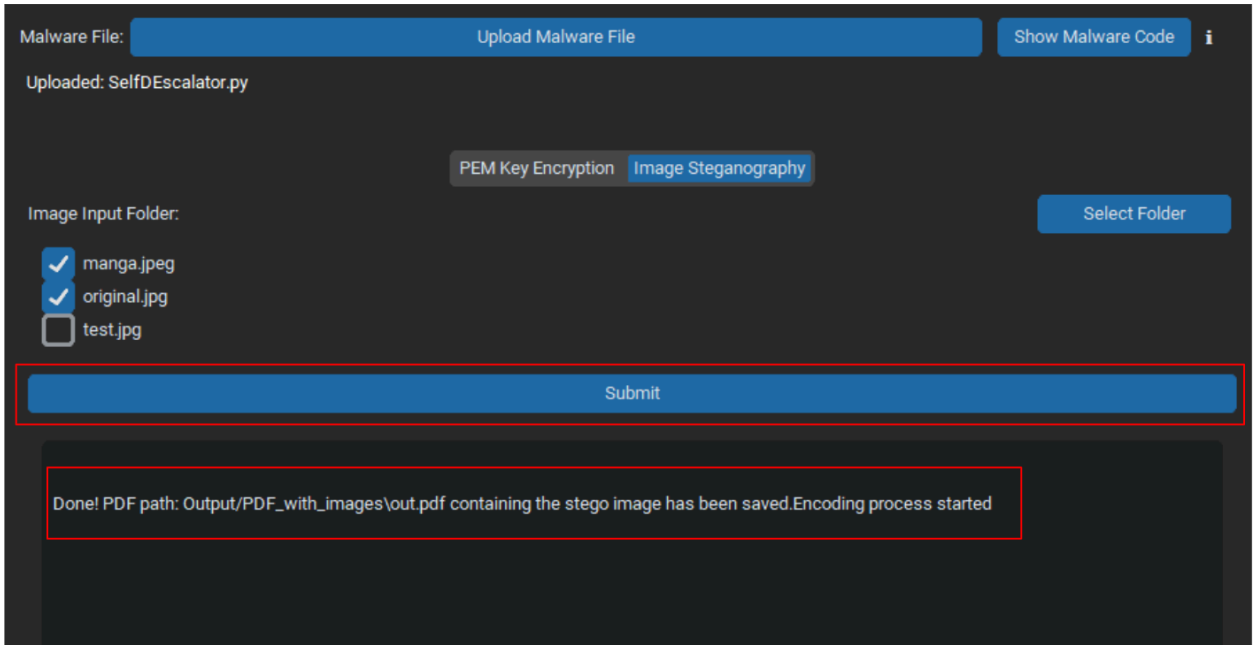
1. **Upload Files**: Upload the malware python code by clicking the Upload **Malware File** button and select your file.
2. **Images Selection:** Click the **Select Folder** button to choose the folder where images you wish to perform steganography embedding into.



If the image files exist in the folder, it will display as options to check.


ObfusQrypt
Code Embedder
QMorph Malware
Macro Excel Embedder
PDF Embedder
PyCompile Corrupt Magic
Dynamic Cipher
Compile with CythonCompile
Malware File:
Upload Malware File
Show Malware Code
Uploaded: SelfDEscalator.py
PEM Key Encryption
Image Steganography
Image Input Folder:
Select Folder
manga.jpeg
original.jpg
test.jpg
Submit

3. **Submit**: Select the Images you want to embed your code into via Steganography and click **Submit** to embed the malware into the images and store them as an output pdf. The Preview at the bottom will show the result of the embedding and the location of the output file.

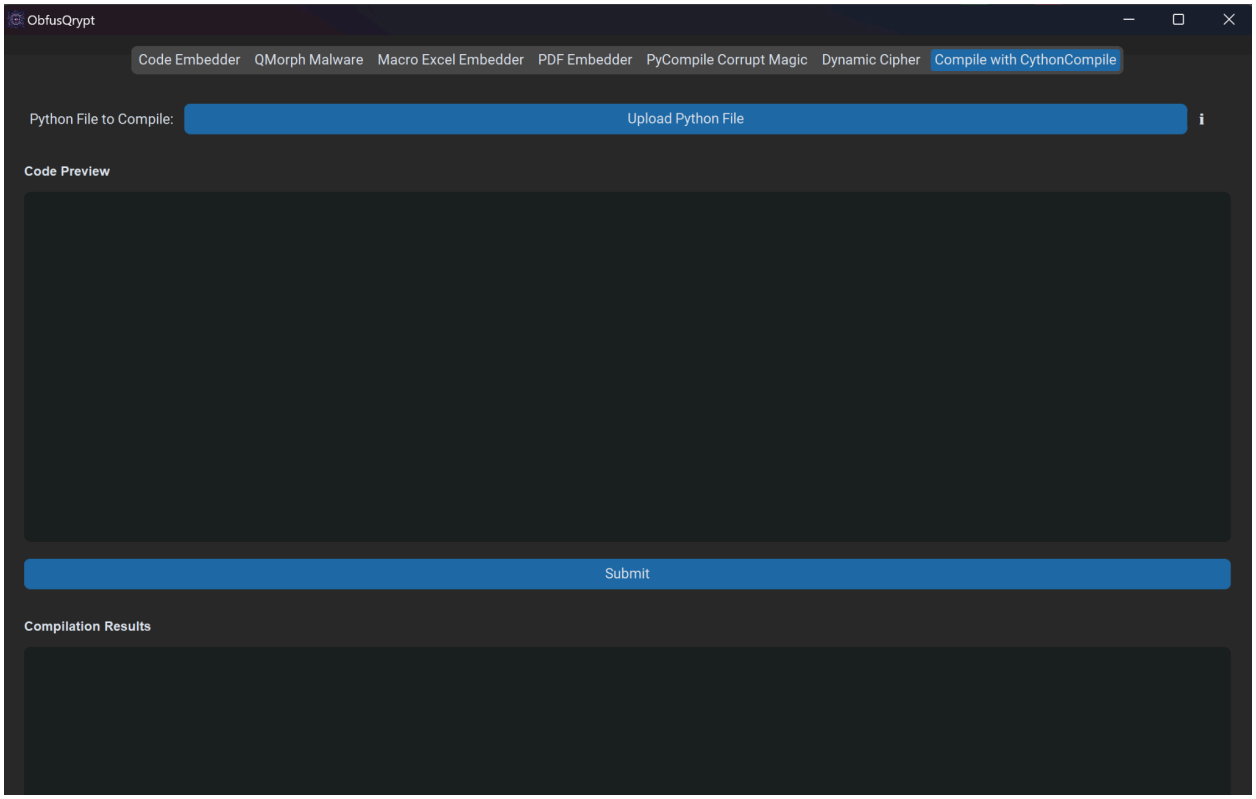


4. Compile with Cython

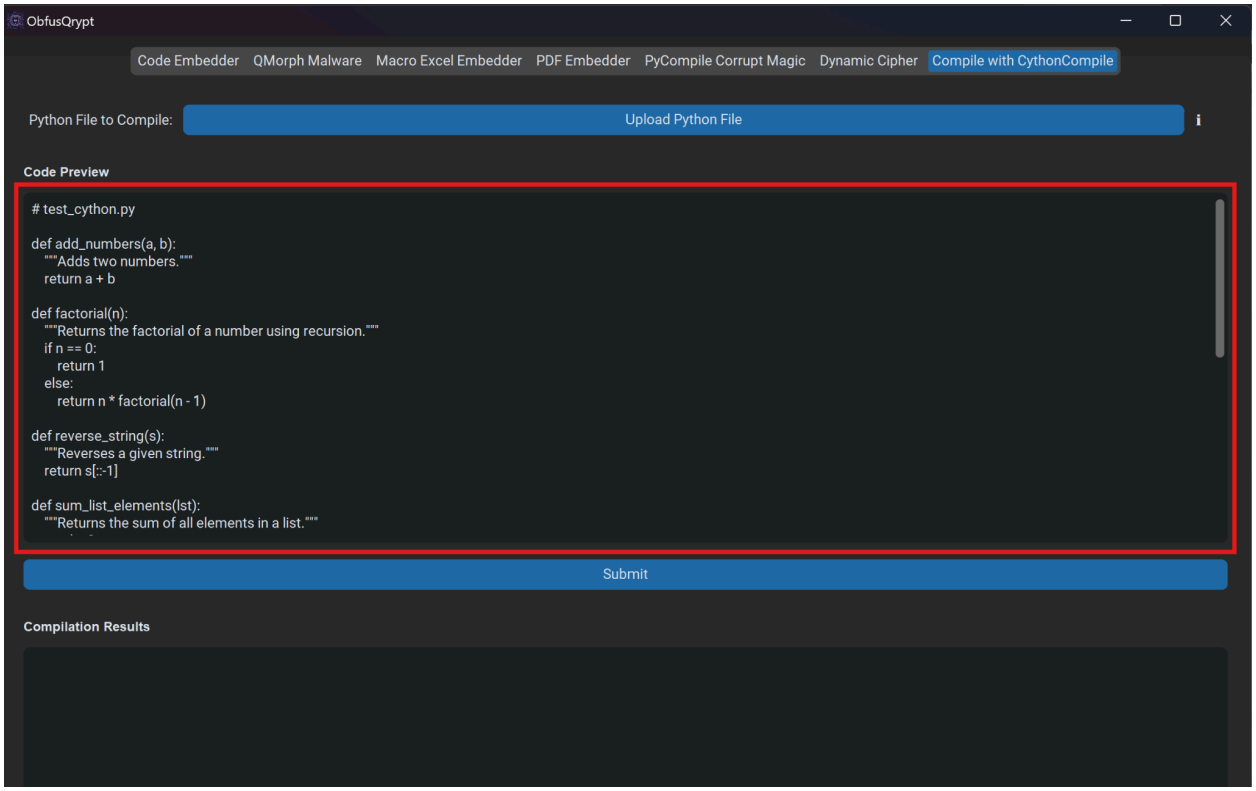
The **Compile with Cython** tab converts Python code into a ".exe" file using Cython for obfuscation.

Usage:

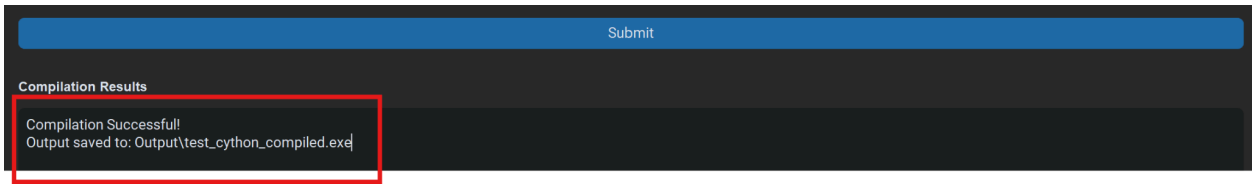
1. **Upload File**: Click **Upload Python File**.



Once uploaded the preview of the code is displayed



2. **Compile**: Click **Submit** to compile.



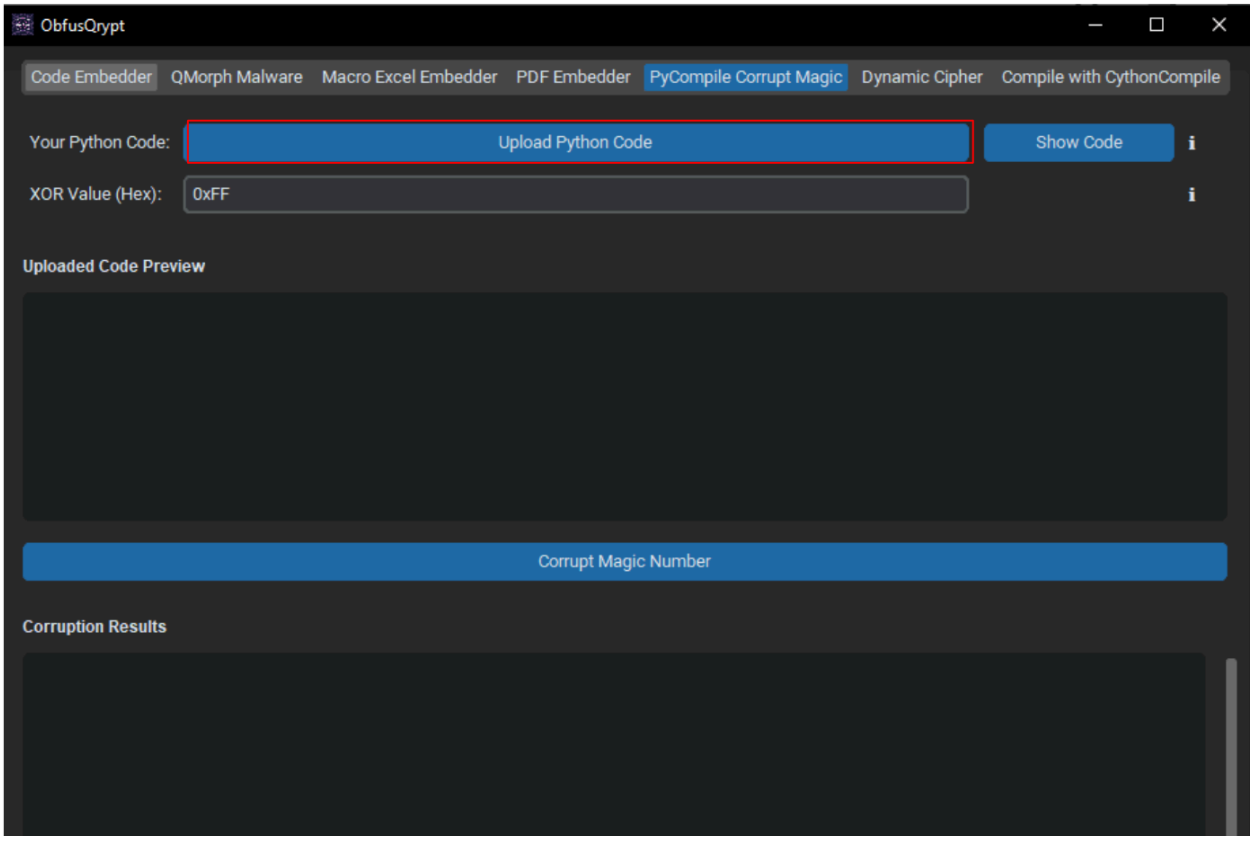
Upon successful submission the result file location is displayed.

5. PyCompile Corrupt

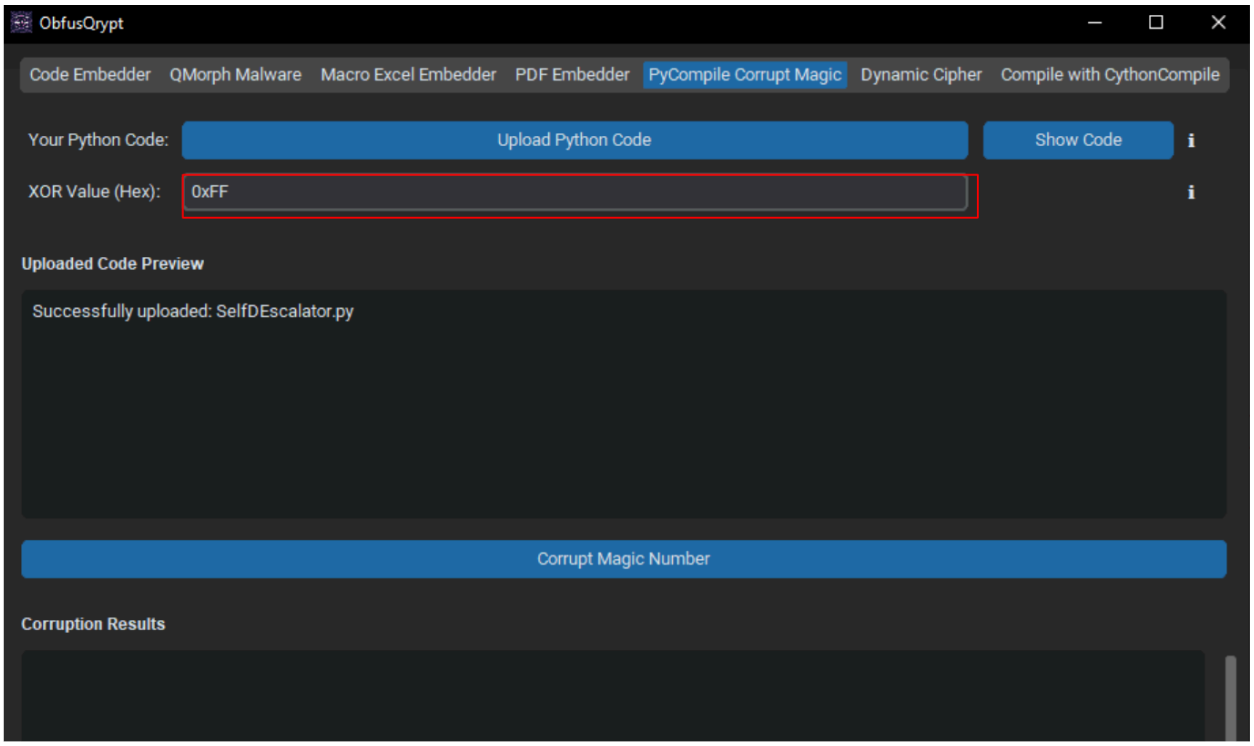
The **PyCompile Corrupt Magic** tab modifies the magic number in Python bytecode files to obfuscate it, rendering the file unreadable.The function applies an XOR operation on the magic number in the compiled Python file using the specified value.

Usage:

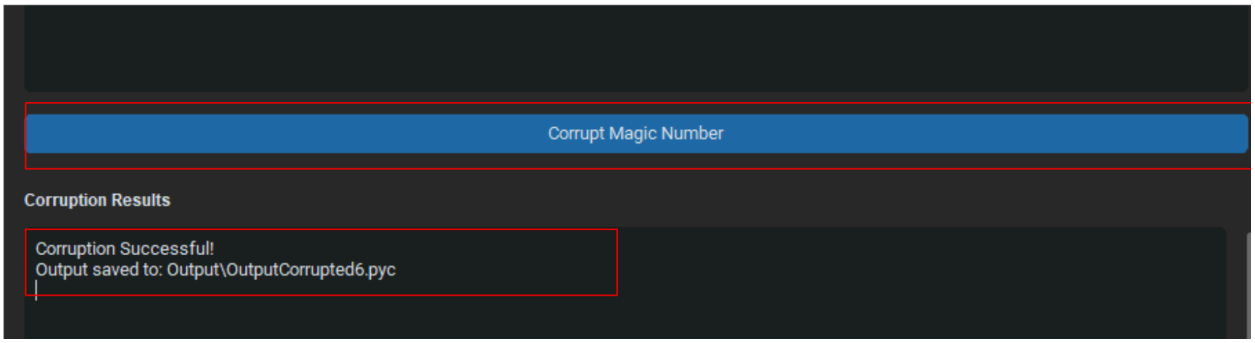
1. **Upload Code**: Click **Upload Python Code** and select the code you want to compile.



2. **Enter XOR Value**: Enter a hexadecimal XOR value in the **XOR Value (Hex)** field (default "0xFF"). The range is between 0x00-0xFF.



3. **Corrupt Magic Number**: Click **Corrupt Magic Number** to execute the corruption.



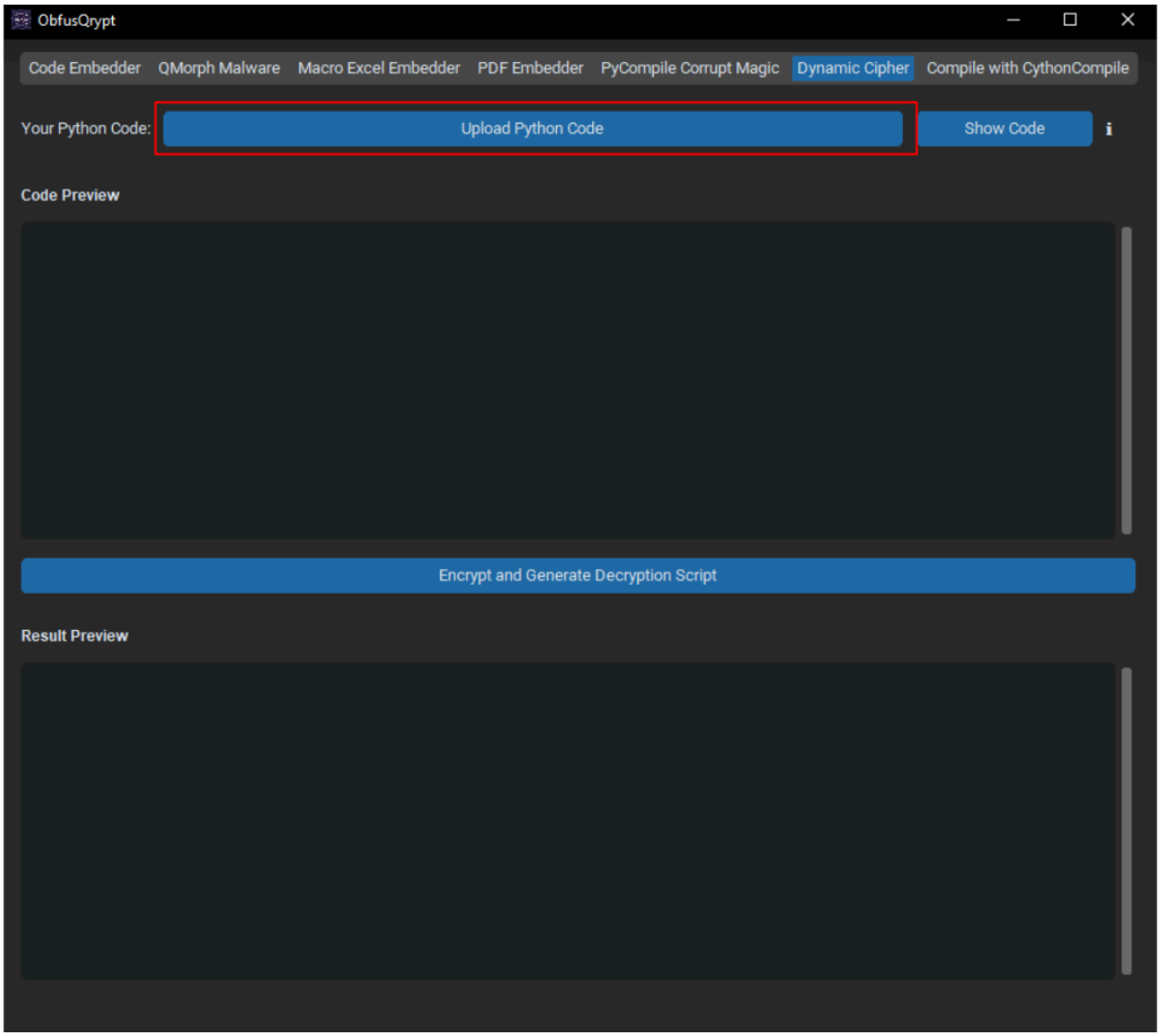
If successful, the output will be displayed relative to the Location of the UI python code, outputting the auto generated name of the compiled code.

6. Dynamic Cipher

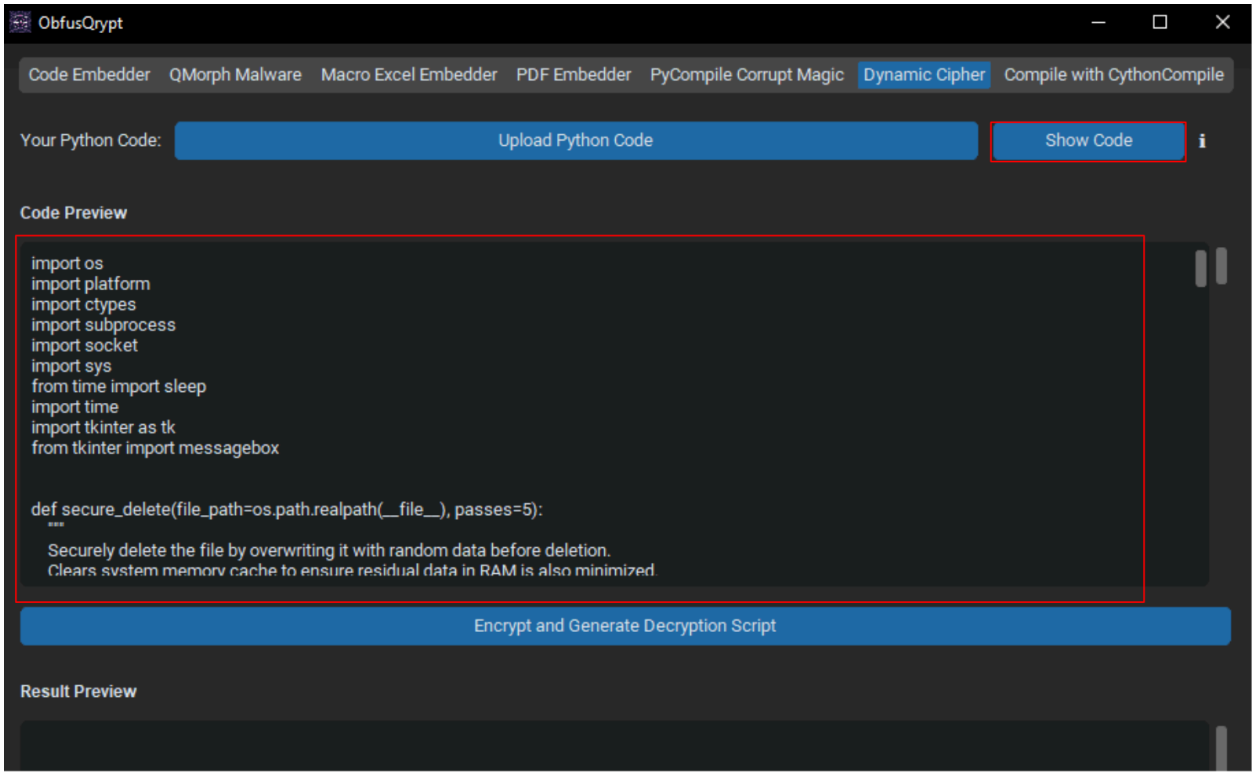
The **Dynamic Cipher** tab provides encryption for Python scripts, generating a decryption script alongside to use for runtime.

Usage:

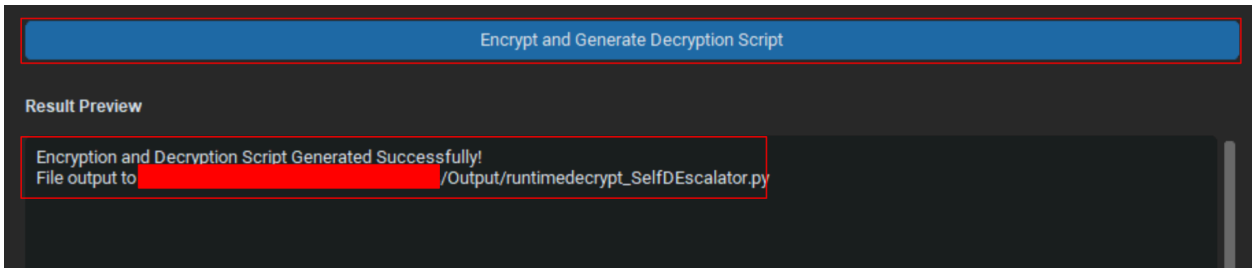
1. **Upload Code**: Click **Upload Python Code** and select a ".py" file to encrypt.



2. **Show Code**: Click **Show Code** to display the file content in the text box.



3. **Encrypt & Generate Script**: Press **Encrypt and Generate Decryption Script** to encrypt the code. The results display in the output box, including a success message and any output files.

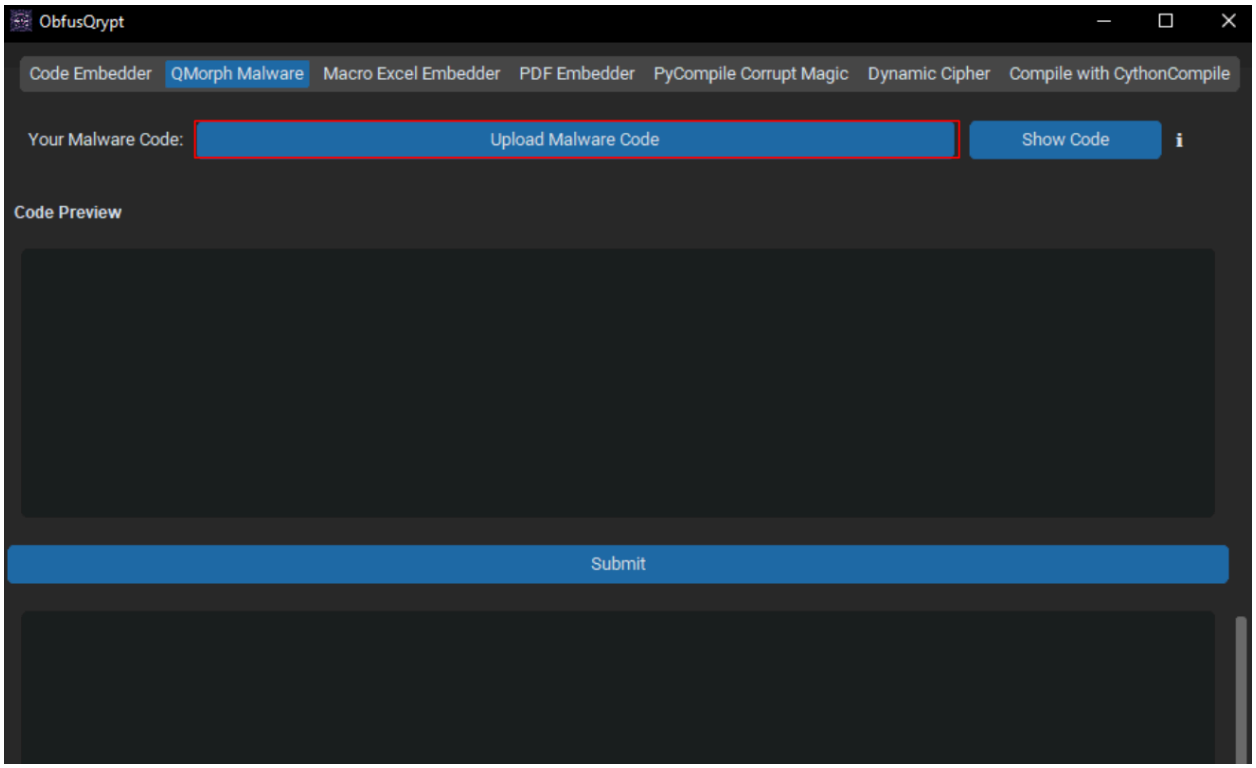


7. QMorph Malware

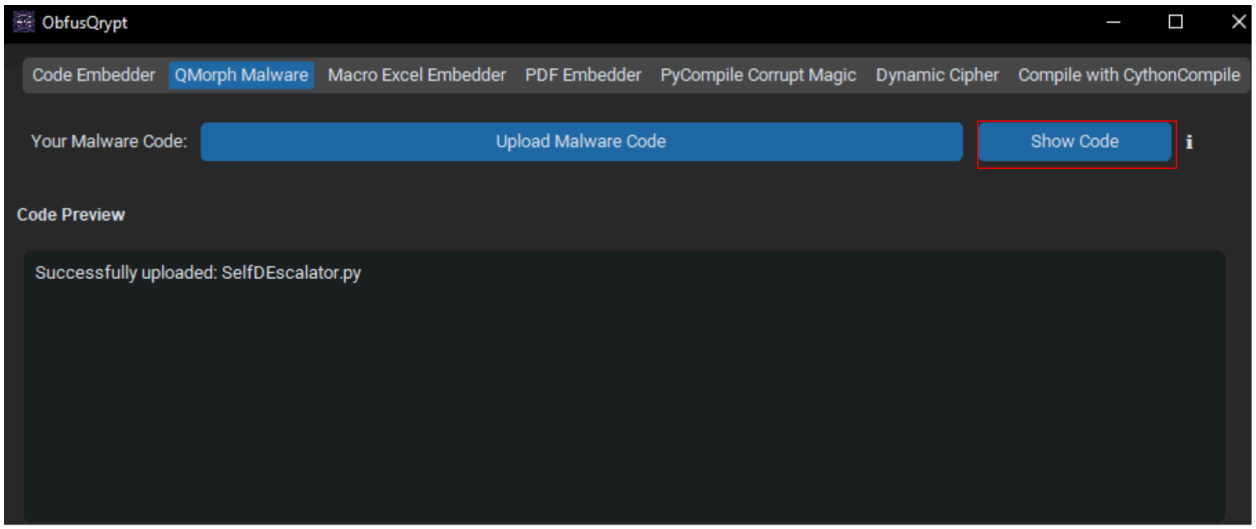
The **QMorph Malware** tab obfuscates malware code for enhanced evasion using quantum-inspired morphing.

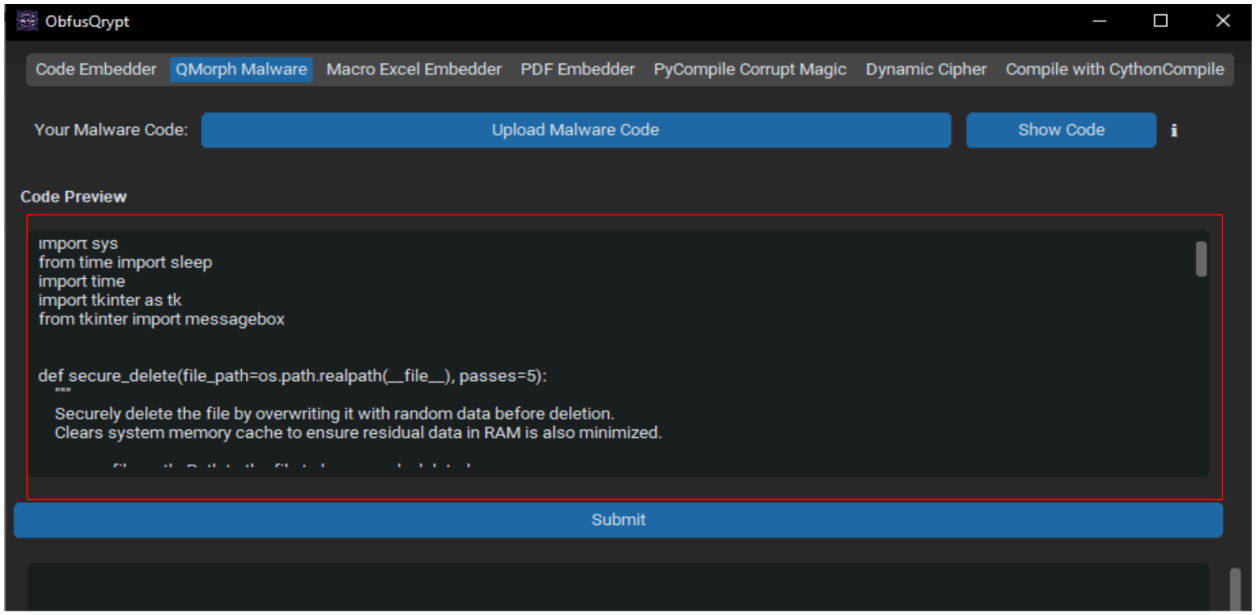
Usage:

1. **Upload Malware Code**: Click **Upload Malware Code** and upload the code you want to obfuscate.

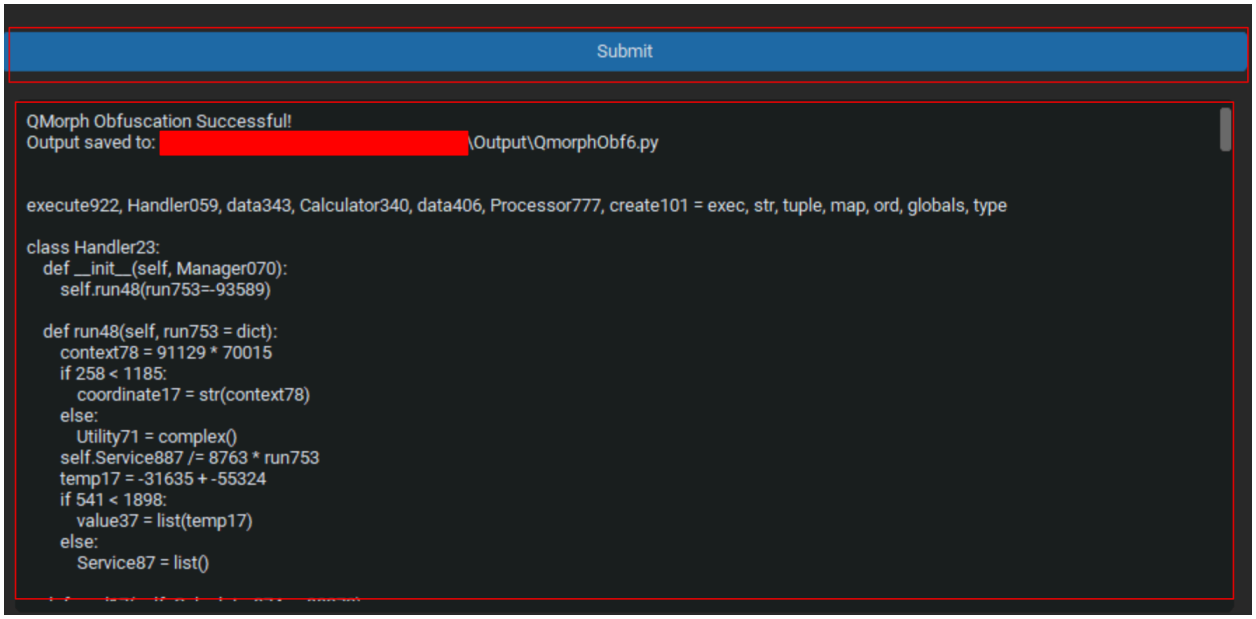


2. **View Code Preview:** Click the **Show Code** button next to the upload button to view code in the code preview pane.





3. **Obfuscate**: Press **Submit** to apply QMorph obfuscation.



The "submit_qmorph" function uses "QMorph" to obfuscate malware code, making it harder to reverse-engineer.