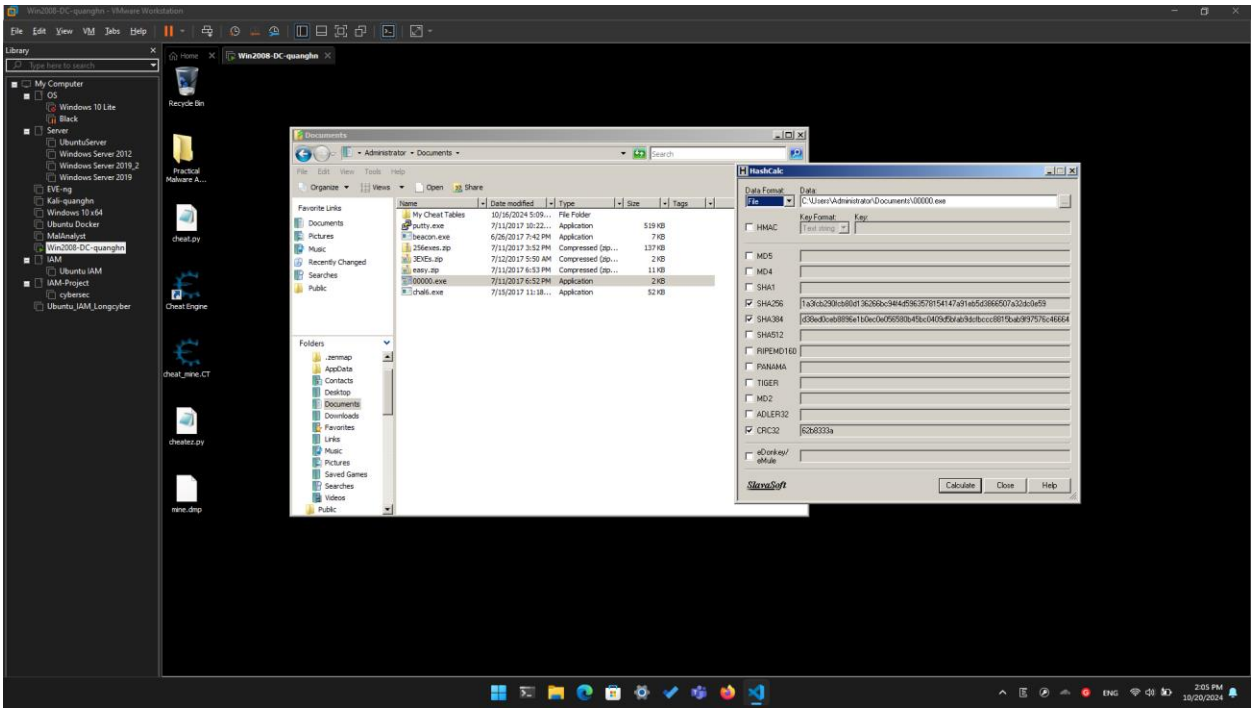
Lab 18.2: Patching EXEs with Ollydbg

Huynh Ngoc Quang – SE181838

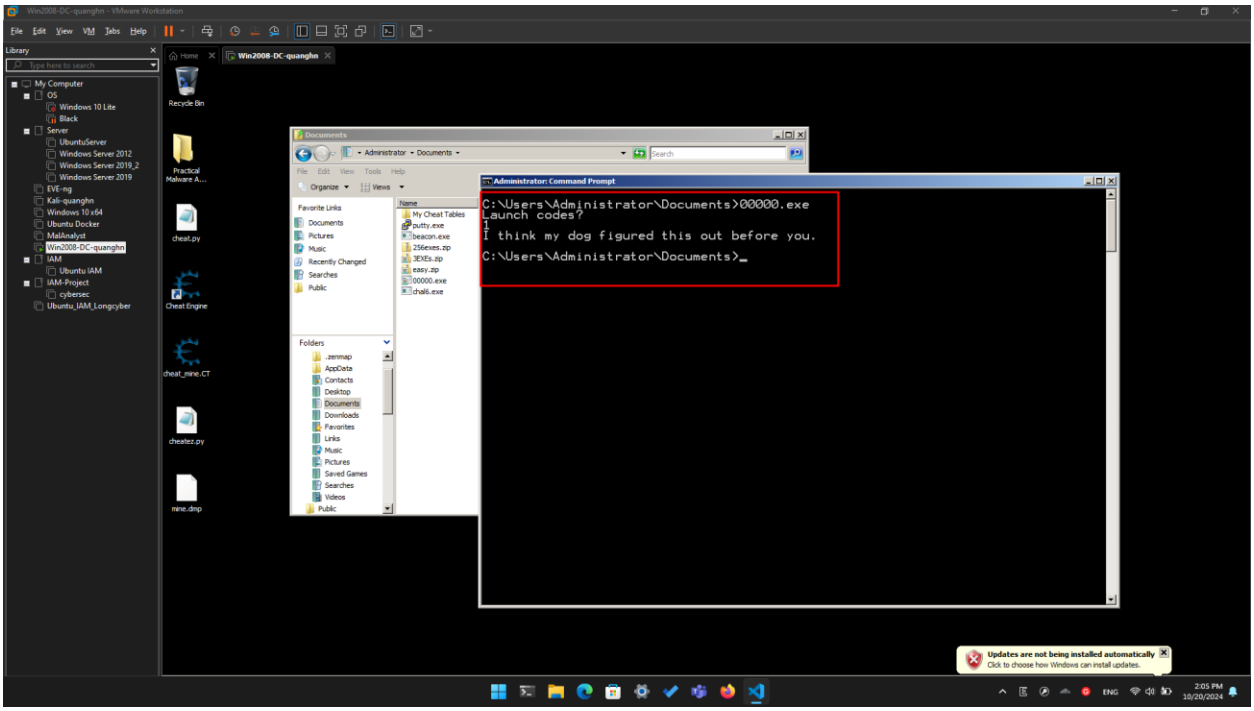
A. Patching an EXE

Checking the Hash:

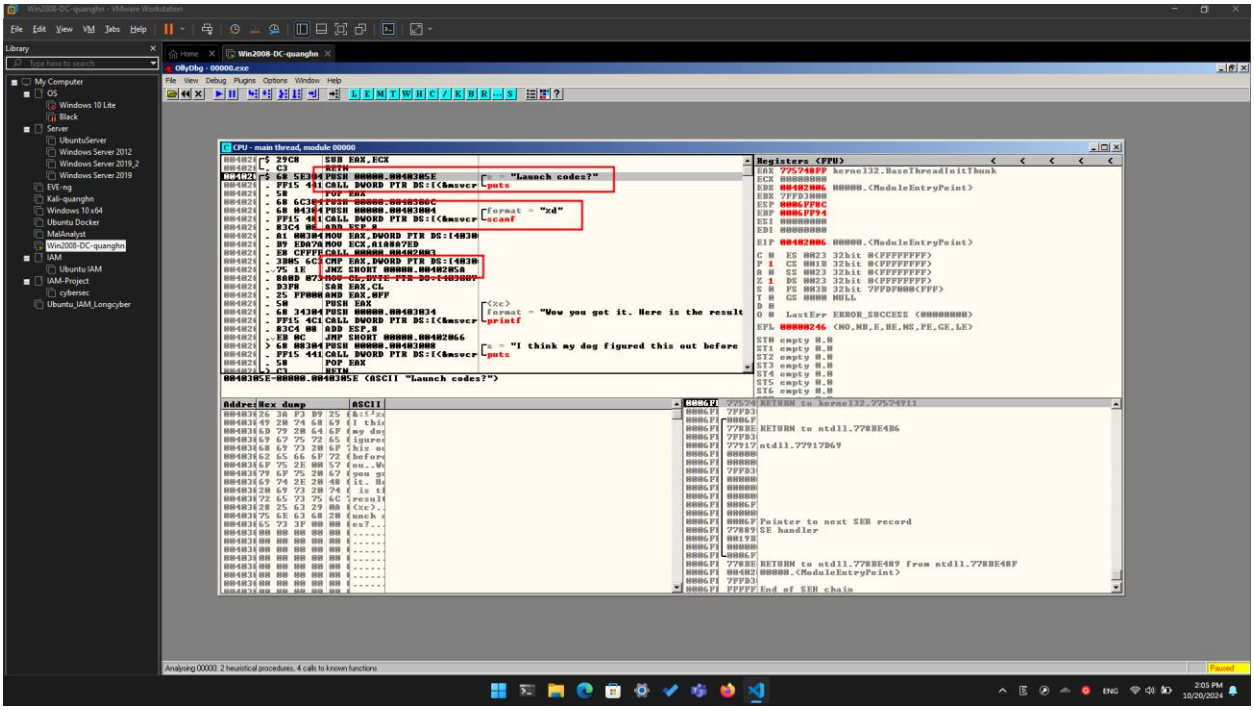


SHA256:
1a3fcb290fcb80d136266bc94f4d5963578154147a91eb5d3866507a32dc0e59

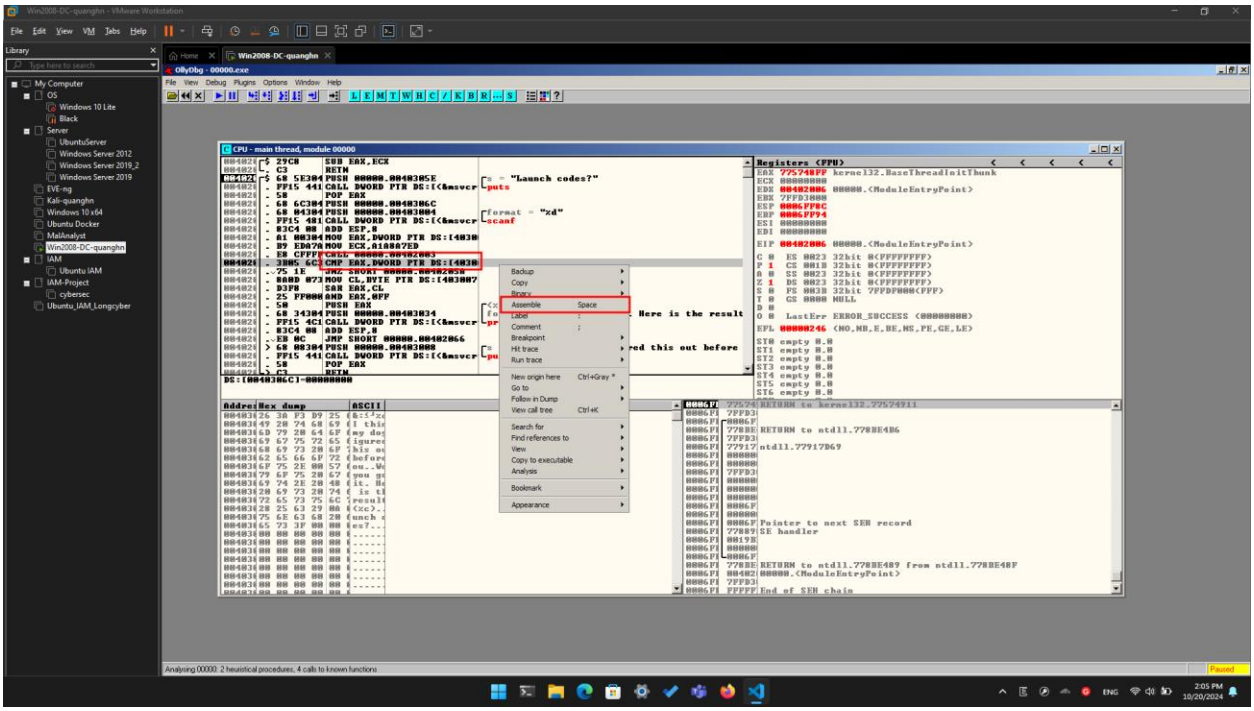
Running the EXE:

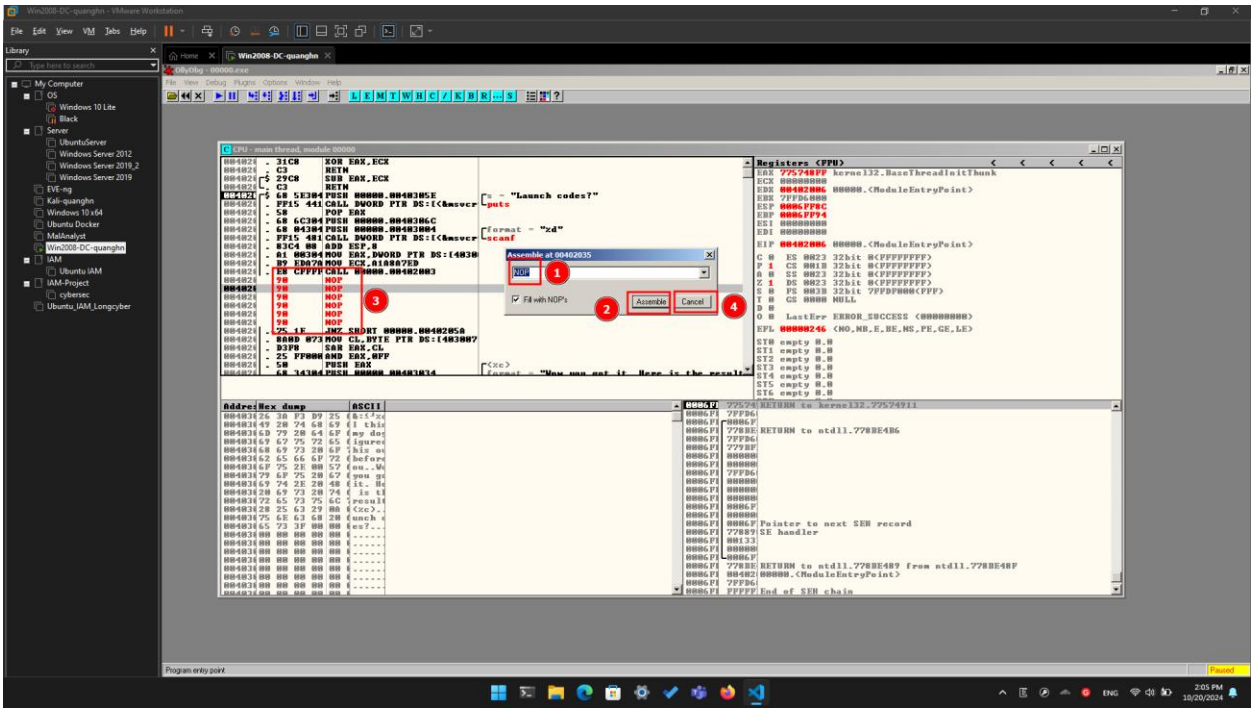


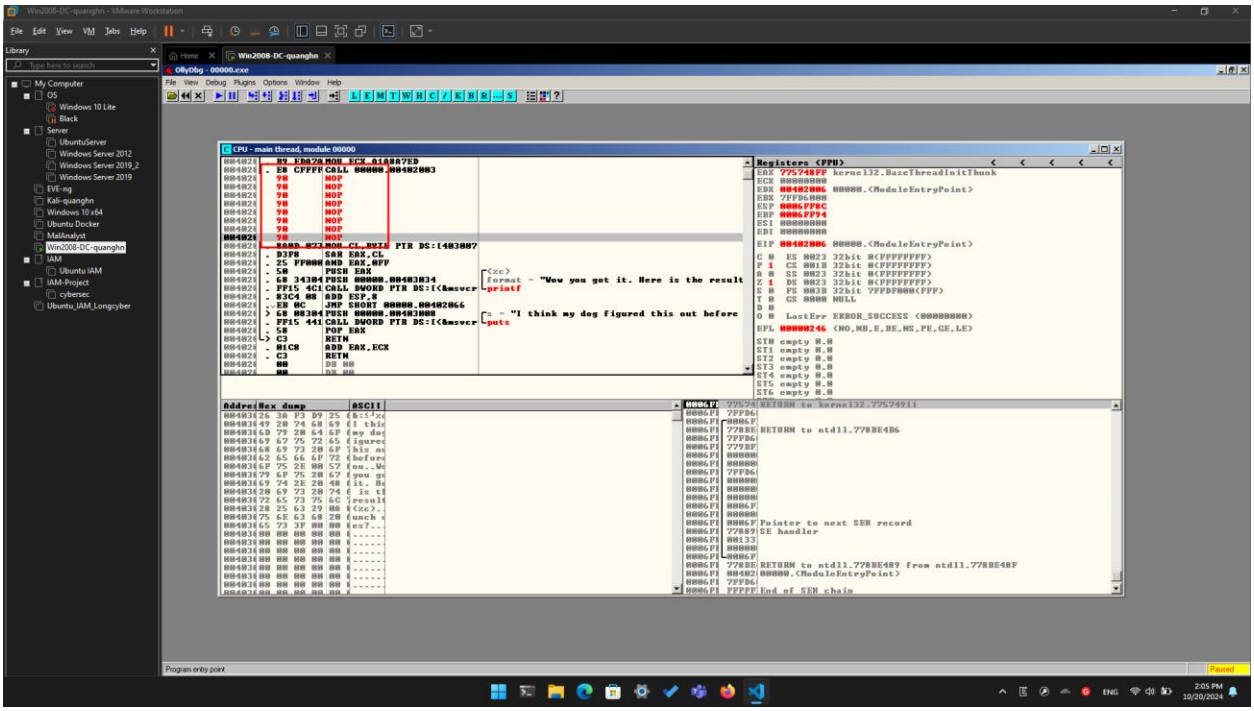
Examining the EXE with Ollydbg:



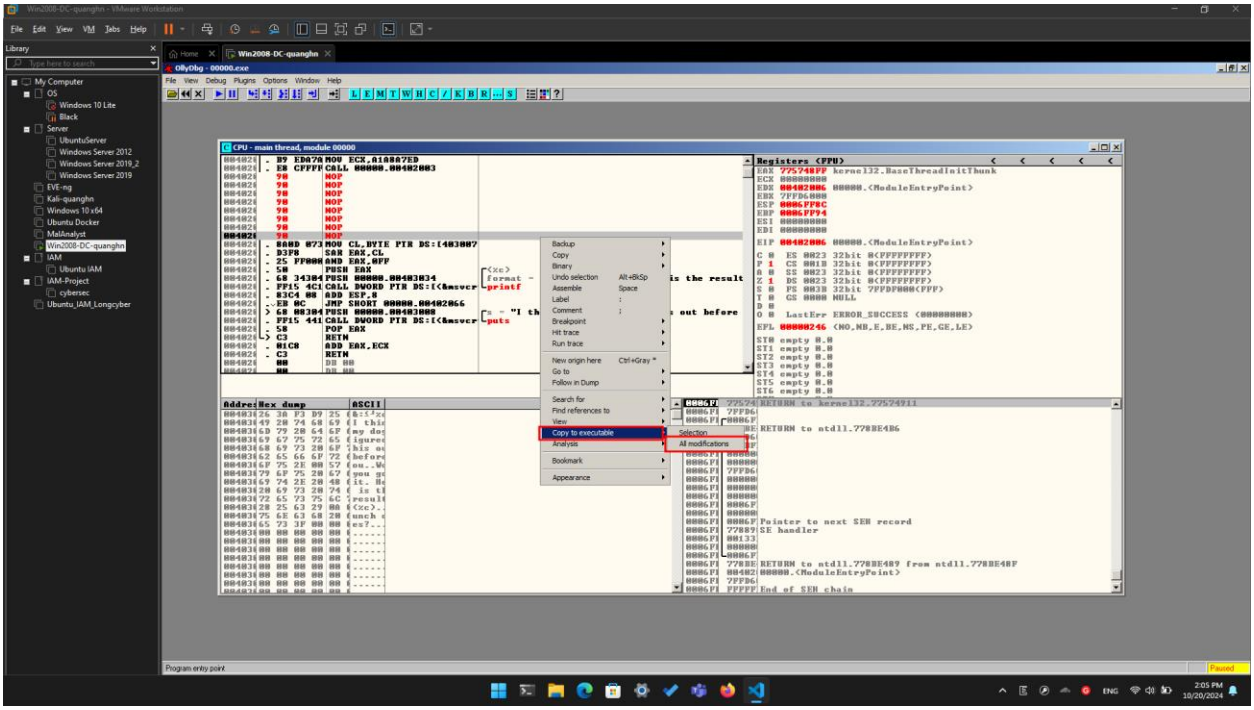
Modifying the EXE:

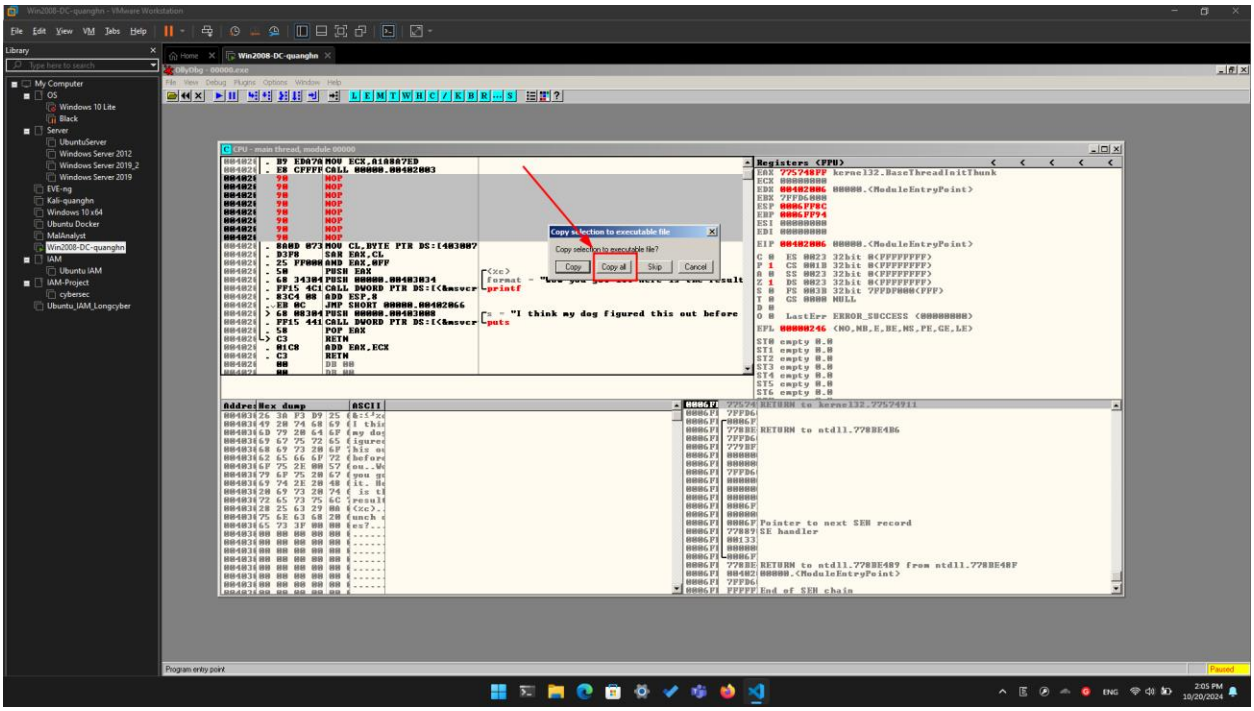


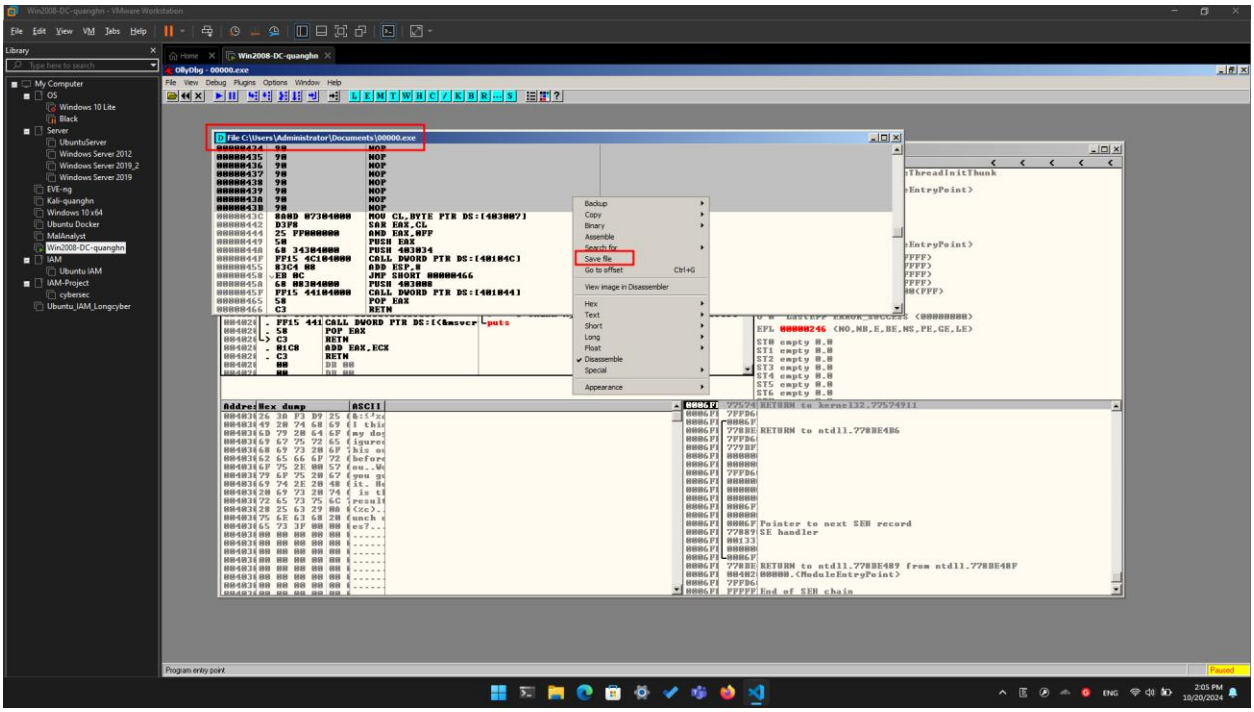

1
2
3
4

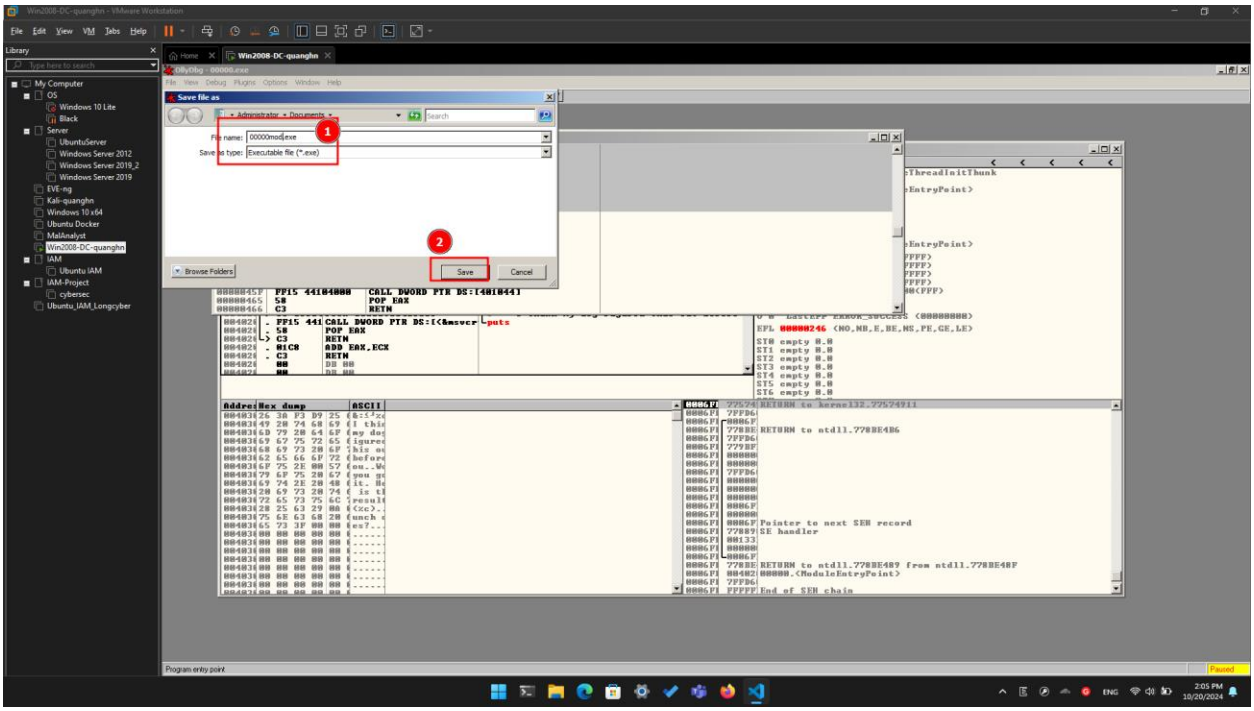


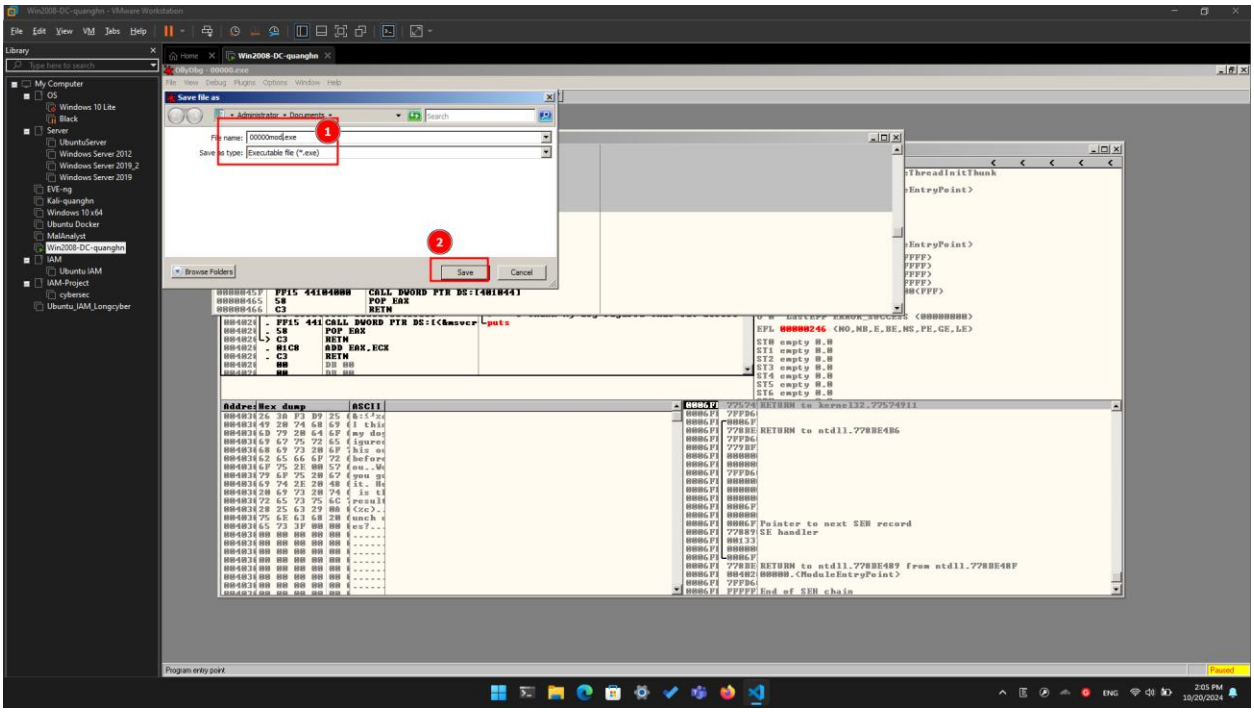
Saving the Modified File:



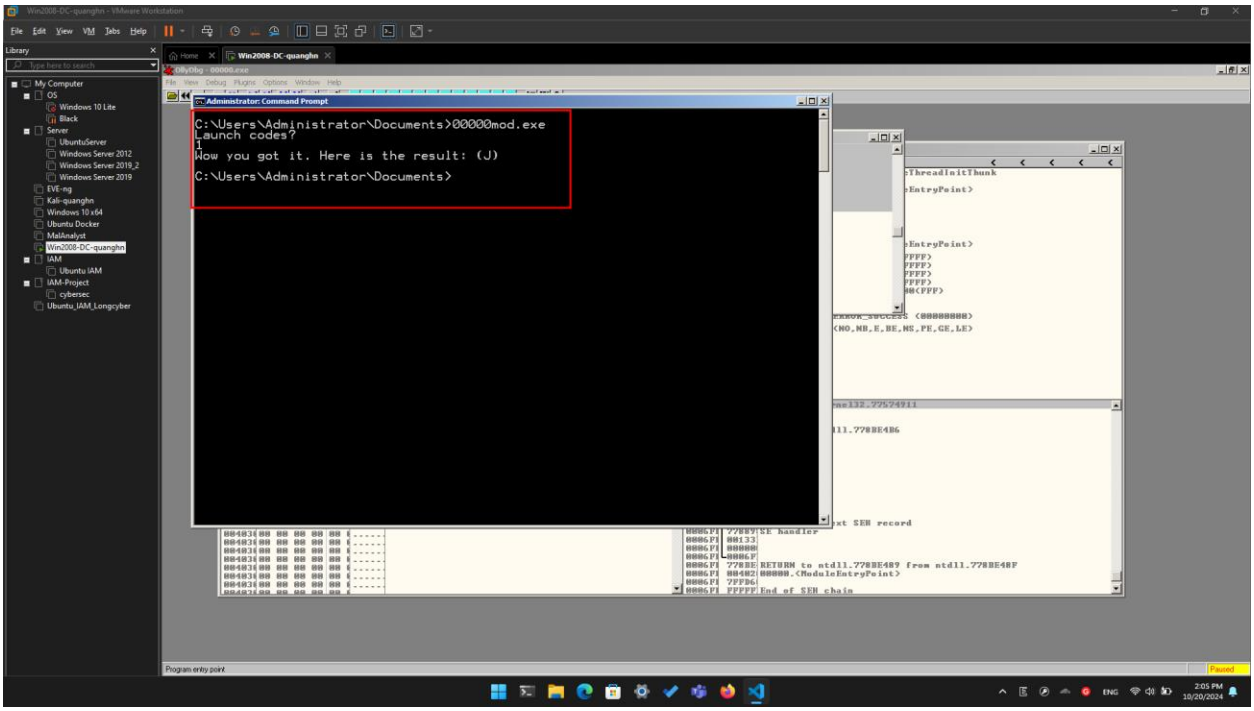




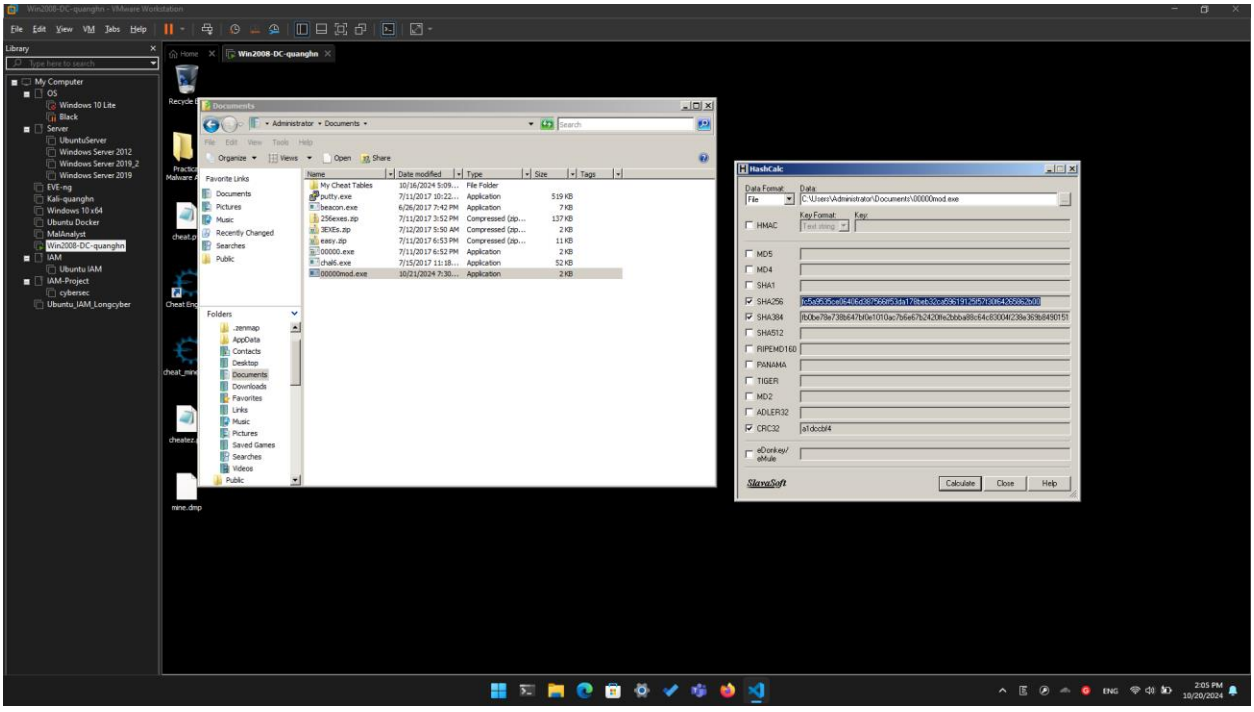




Running the Modified File



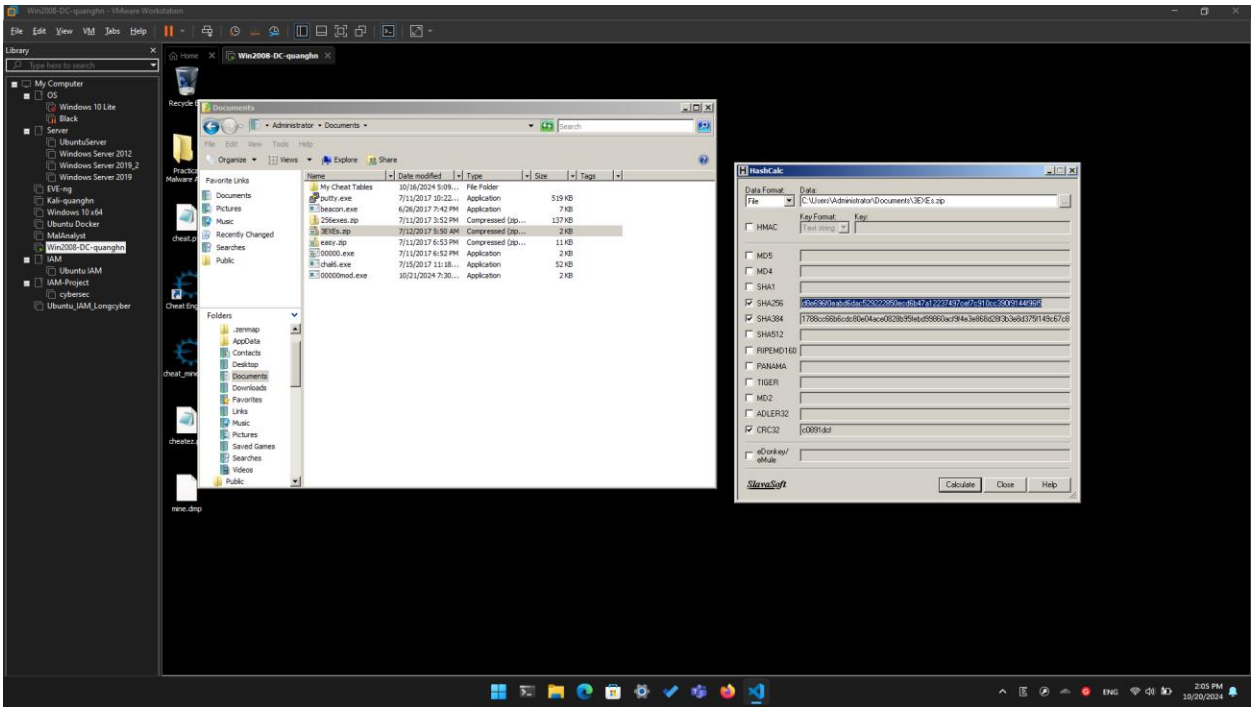
Checking the Hash



CRC32: a1dccbf4

B. Patching three EXEs:

Checking the Hash:

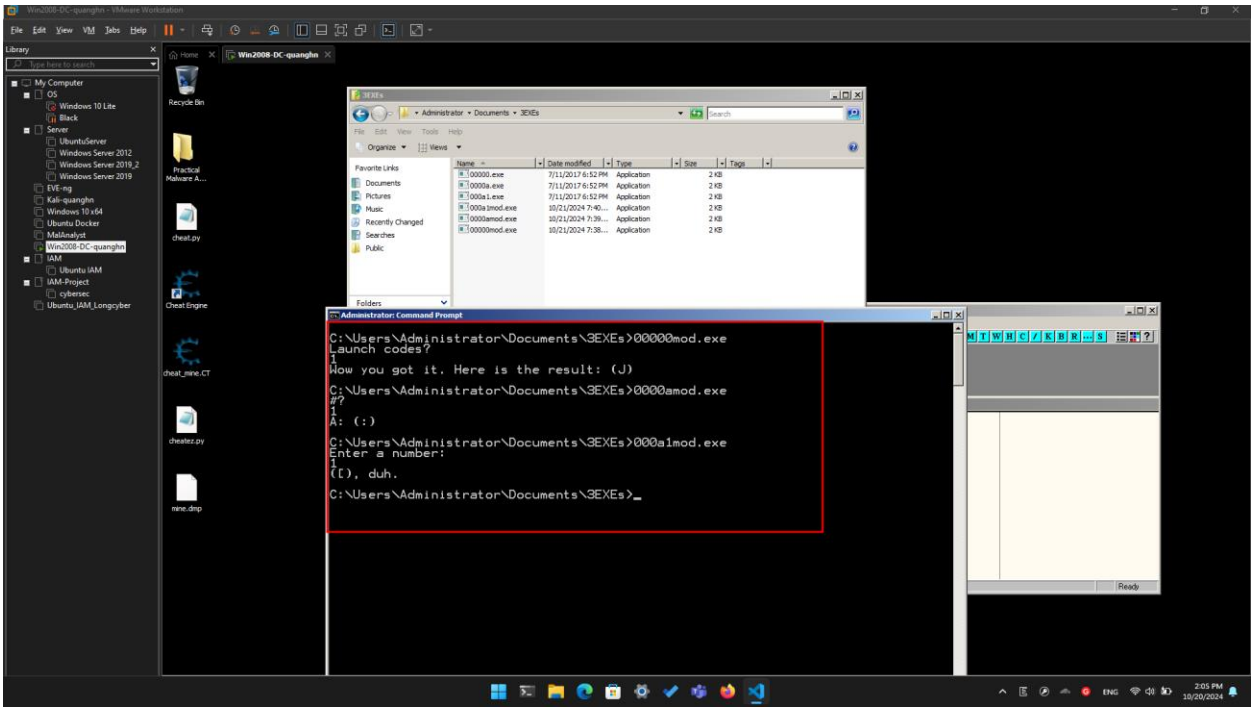


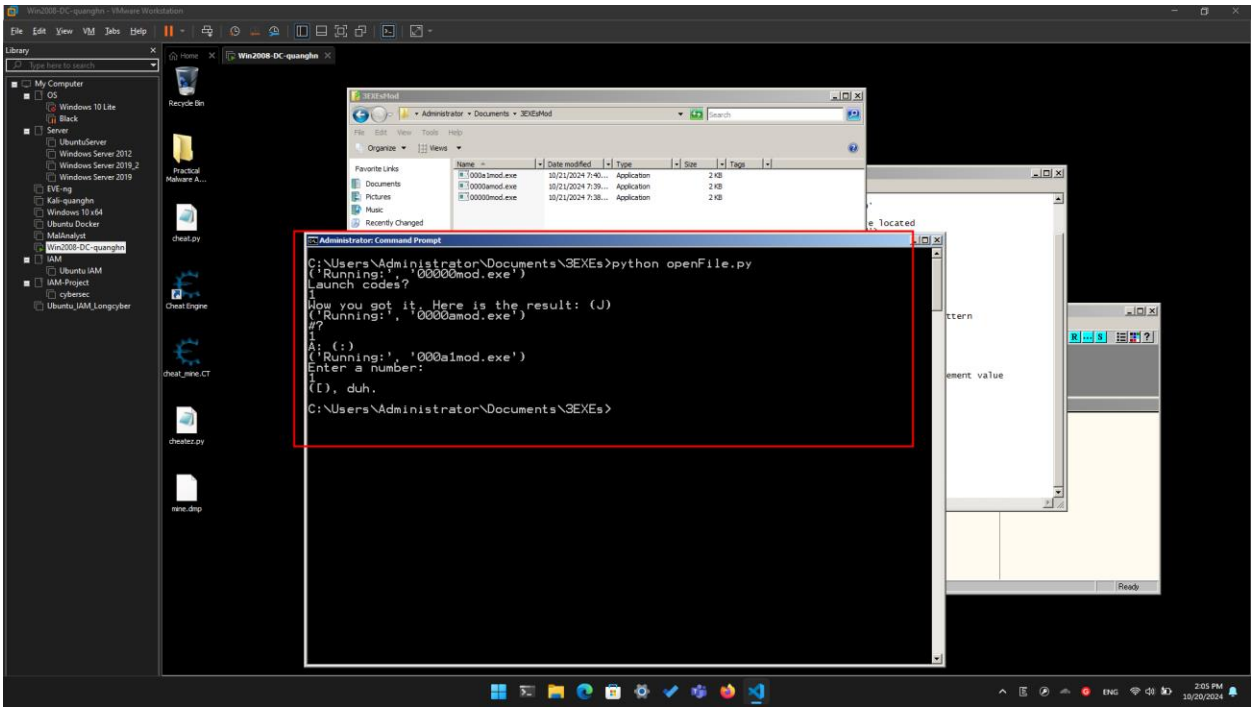
SHA256:

d8e696f0eabd6dac529222850ecd6b47a12237497cef7c910cc390f9144f96f5

Patch the Files: like above

Gather the Results:





C. Patching 19 EXEs:

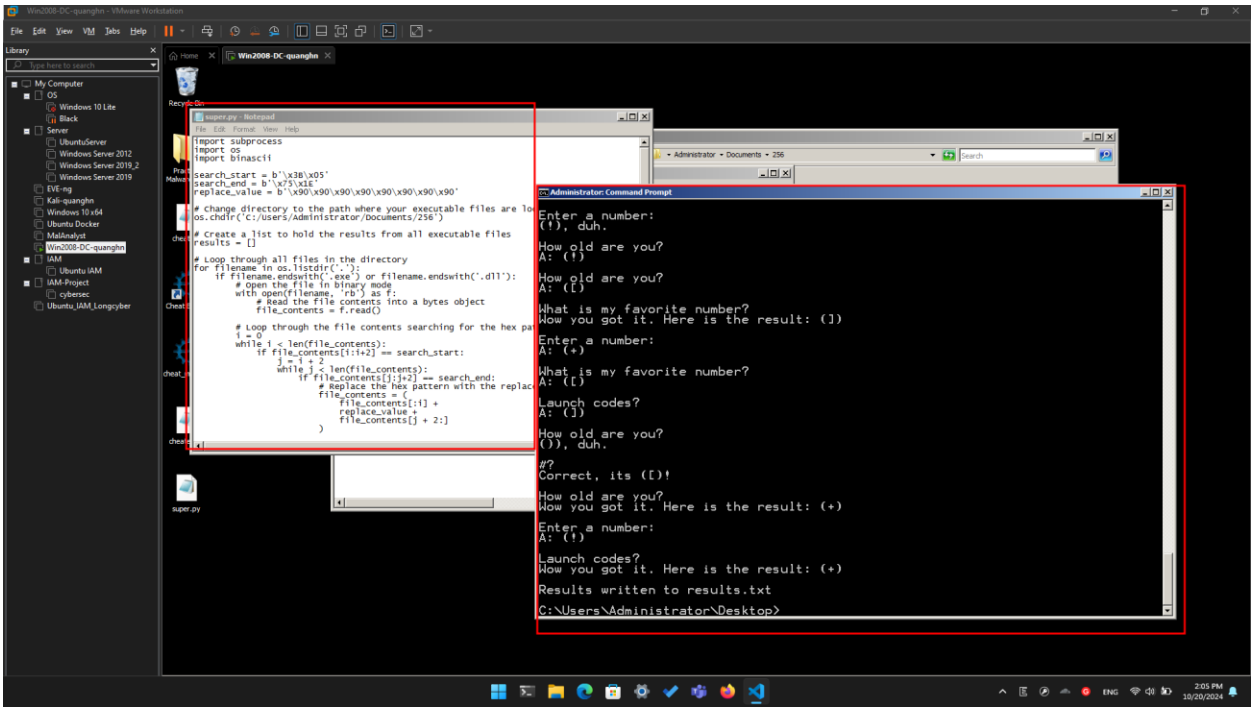
I created a super.py script to automatically complete all step above.

```
import subprocess
import os
import binascii

search_start = b'\x3B\x05'
search_end = b'\x75\x1E'
replace_value = b'\x90\x90\x90\x90\x90\x90\x90\x90'

# Change directory to the path where your executable files are located
os.chdir('C:/Users/Administrator/Documents/256')

# Create a list to hold the results from all executable files
results = []

# Loop through all files in the directory
for filename in os.listdir('.'):
    if filename.endswith('.exe') or filename.endswith('.dll'):
        # Open the file in binary mode
        with open(filename, 'rb') as f:
            # Read the file contents into a bytes object
            file_contents = f.read()

        # Loop through the file contents searching for the hex pattern
        i = 0
        while i < len(file_contents):
            if file_contents[i:i+2] == search_start:
                j = i + 2
                while j < len(file_contents):
```

```
                    if file_contents[j:j+2] == search_end:
                        # Replace the hex pattern with the replacement value
                        file_contents = (
                            file_contents[:i] +
                            replace_value +
                            file_contents[j + 2:]
                        )
                        break
                    j += 1
            i += 1

        # Write the modified file contents back to the file
        with open(filename, 'wb') as f:
            f.write(file_contents)

      # Run the executable file with input "18" using subprocess
        proc = subprocess.Popen([filename], stdin=subprocess.PIPE, stdout=subprocess.PIPE)

        # Send input to the subprocess
        proc.stdin.write(b'18\n')
        proc.stdin.close()

        # Wait for the subprocess to finish and get its output
        result = proc.stdout.read()
      print result

        # Extract the string inside the parentheses and add it to the results
        result_str = result.decode('utf-8')
        start_index = result_str.find('(')
        end_index = result_str.find(')')

        if start_index != -1 and end_index != -1:
            results.append(result_str[start_index + 1:end_index])


# Write the concatenated results to a text file
with open('results.txt', 'w') as f:
    f.write(''.join(results))

print('Results written to results.txt')

```

D. Patching 256 EXEs:

Use the super.py code above to automatically patch, run program and collect the output into the result.txt



The content of result.txt:

Javascript:

[][(![]+[][+[]]+([![]]+[][[]][+!+[]+[+[]]]+(![]+[][!+[]+!+[]]+(!![]+[][+[]]+(!![]+[][!+[]+!+[]+!+[]]+(
!![]+[][+!+[]]][([][(![]+[][+[]]+([![]]+[][[]][+!+[]+[+[]]]+(![]+[][!+[]+!+[]]+(!![]+[][+[]]+(!![]+[][!+
[]+!+[]+!+[]]+(!![]+[][+!+