Instituto Tecnológico de San Juan del Río

Tópicos de Ciberseguridad

[R005_Parchar_Ejecutable_Lima]

P R E S E N T A:

[Oscar Alberto Leal Ramírez] [22590042]
[Isaac Castro Islas] [B19140346]
[Hortencia Sánchez Carlos] [B23590533]
[Ingeniería en Sistemas Computacionales]

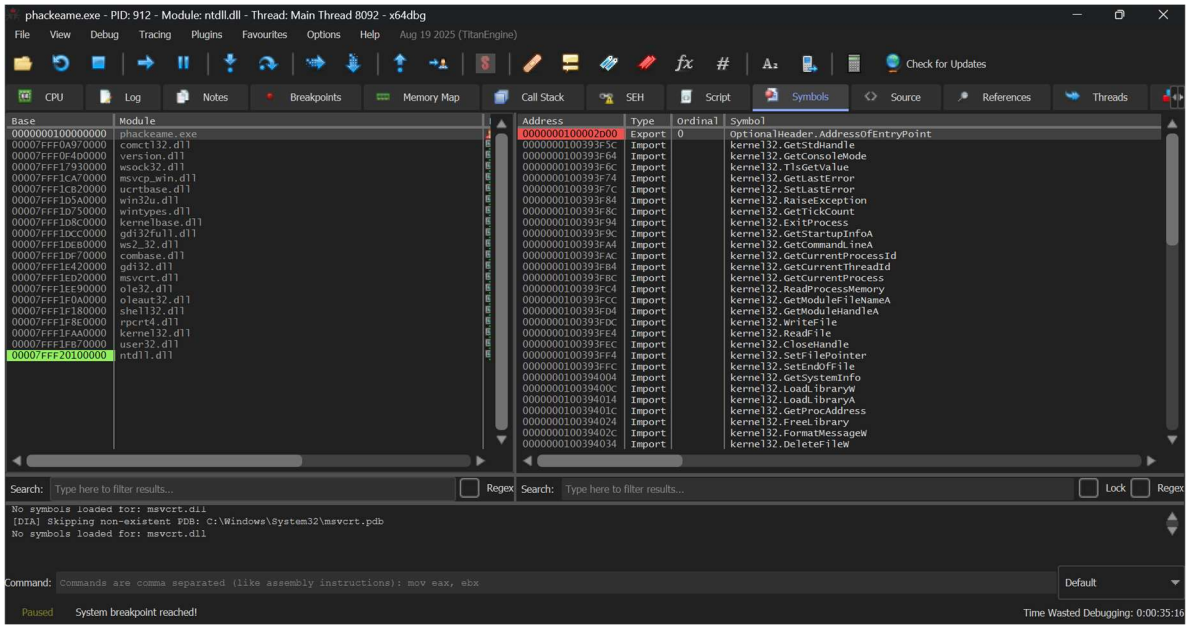
PERIODO [Enero-Junio (2026)]

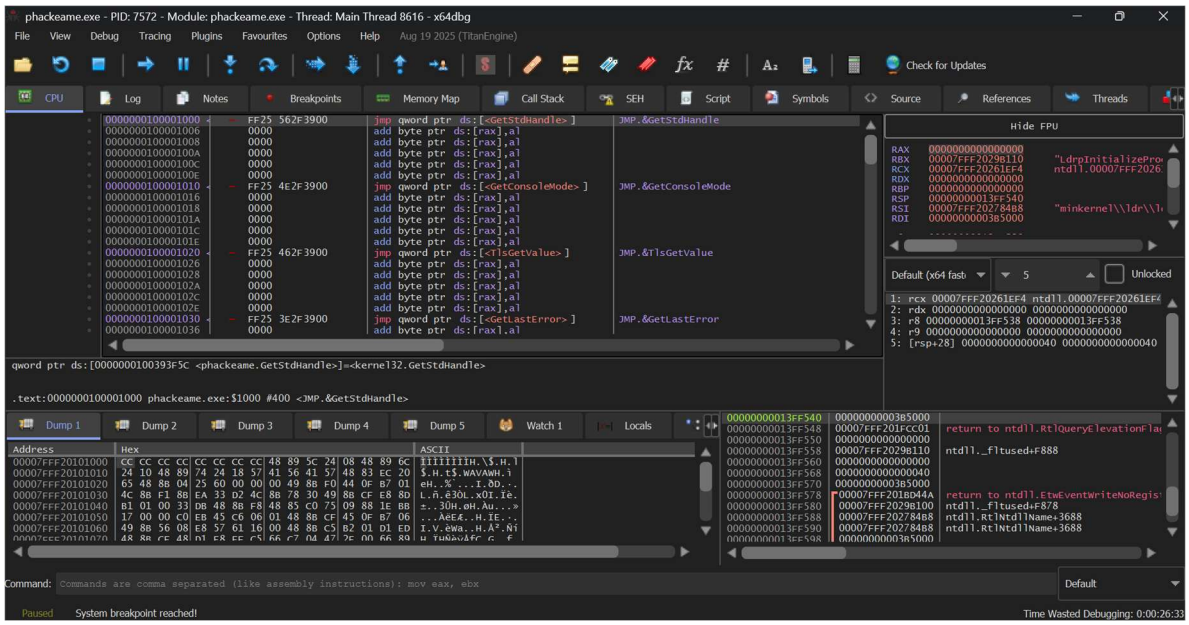
1. Creamos el archivo servidor.py, donde desarrollamos la lógica principal del programa.

```python
from flask import Flask, request, jsonify

app = Flask(__name__)

@app.route('/login', methods=['POST'])
def login():
    datos = request.json
    print(f"Intento de login con: {datos}")
    return jsonify({"status": "error", "message": "Credenciales
inválidas"}), 401

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
```

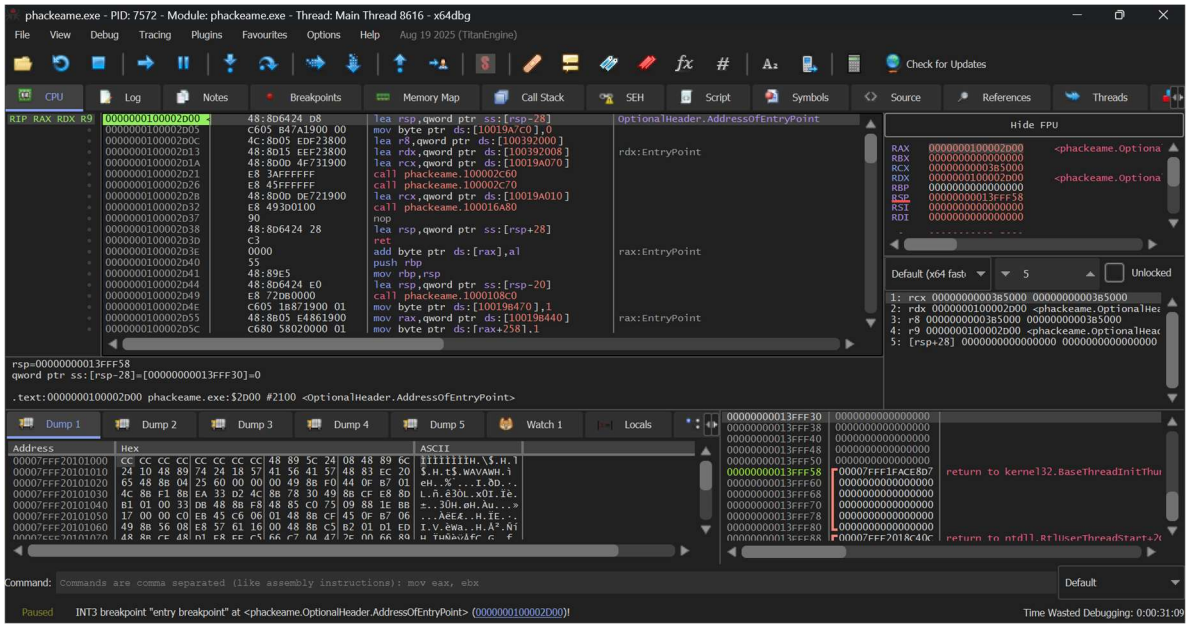
Después, entramos a la carpeta db64 y abrimos el archivo ejecutable .exe para comenzar con el análisis correspondiente.

2). Primero hacemos clic en la pestaña Symbols (arriba, entre SEH y Source); en la lista de la izquierda buscamos y damos doble clic en phackeame.exe, con esto le indicamos al debugger que queremos trabajar con el código del programa y no con el de Windows; finalmente, regresamos a la pestaña CPU (la primera a la izquierda) para continuar con el análisis.

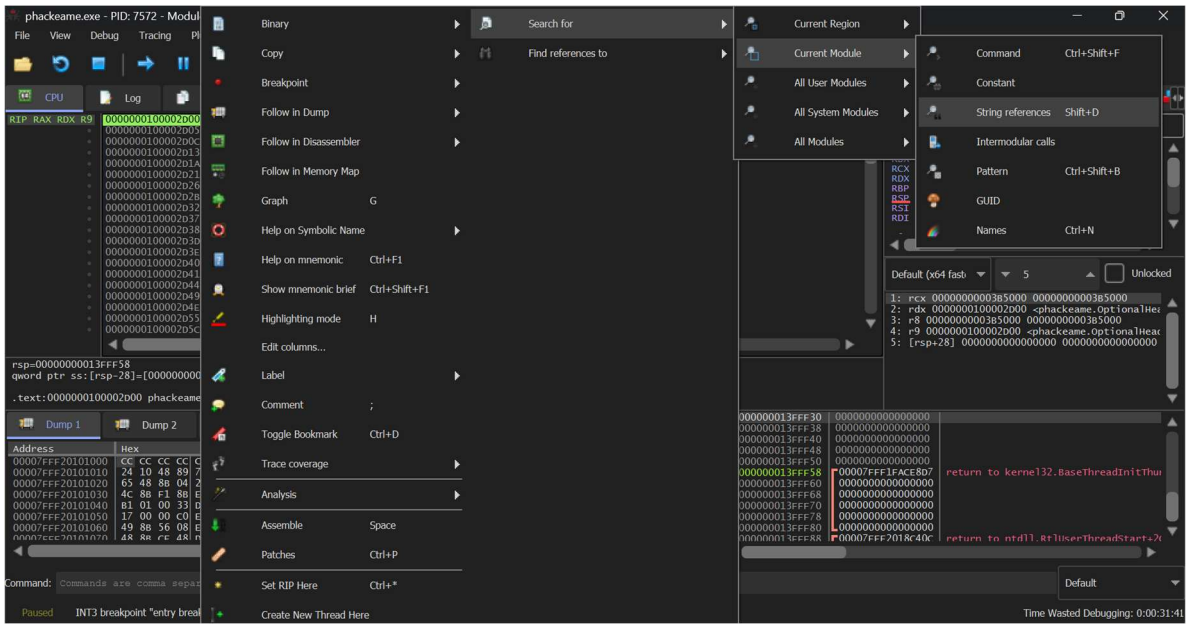




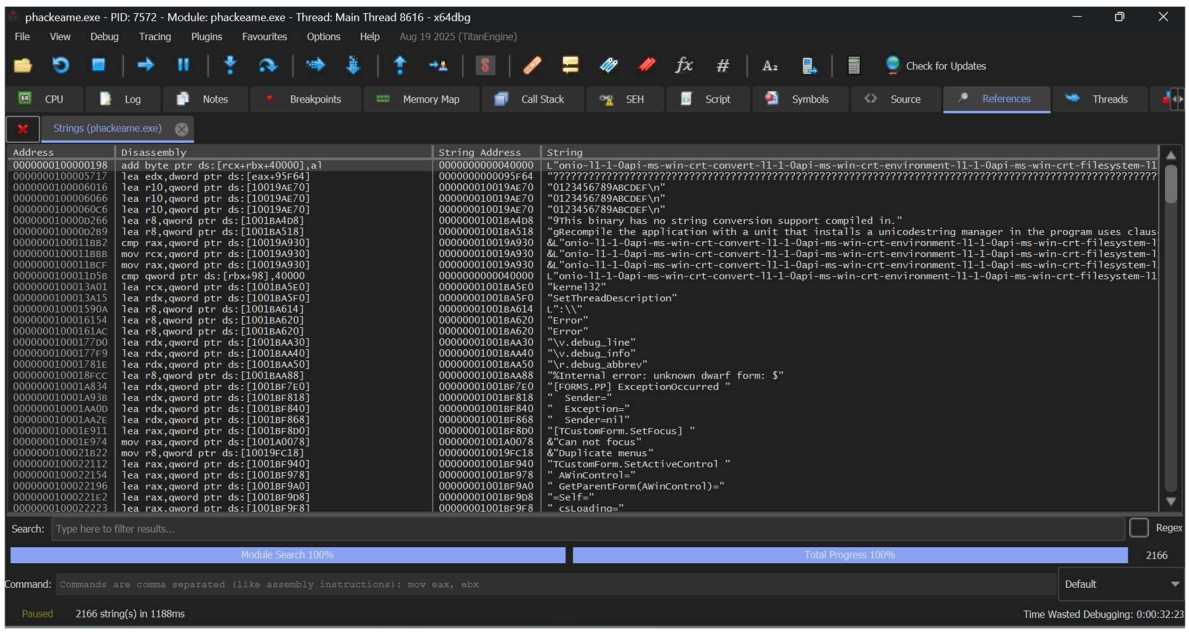
3). Presionamos la tecla F9 (o el botón azul de “Play” que está en la parte superior izquierda) una o dos veces para ejecutar el programa y avanzar en el proceso de depuración.



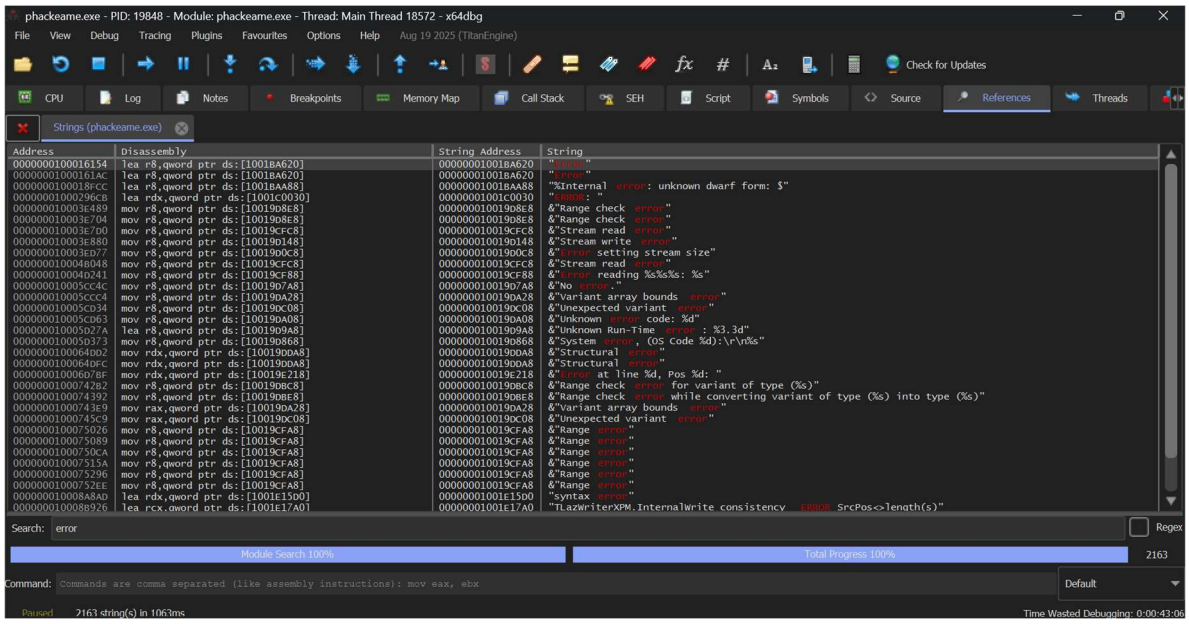
4). Ahora que el proceso está corriendo y el módulo es el correcto, hacemos clic derecho en la ventana negra de código (en la pestaña CPU) y seleccionamos la opción Search for -> Current Module -> String references para buscar las cadenas de texto dentro del módulo actual.



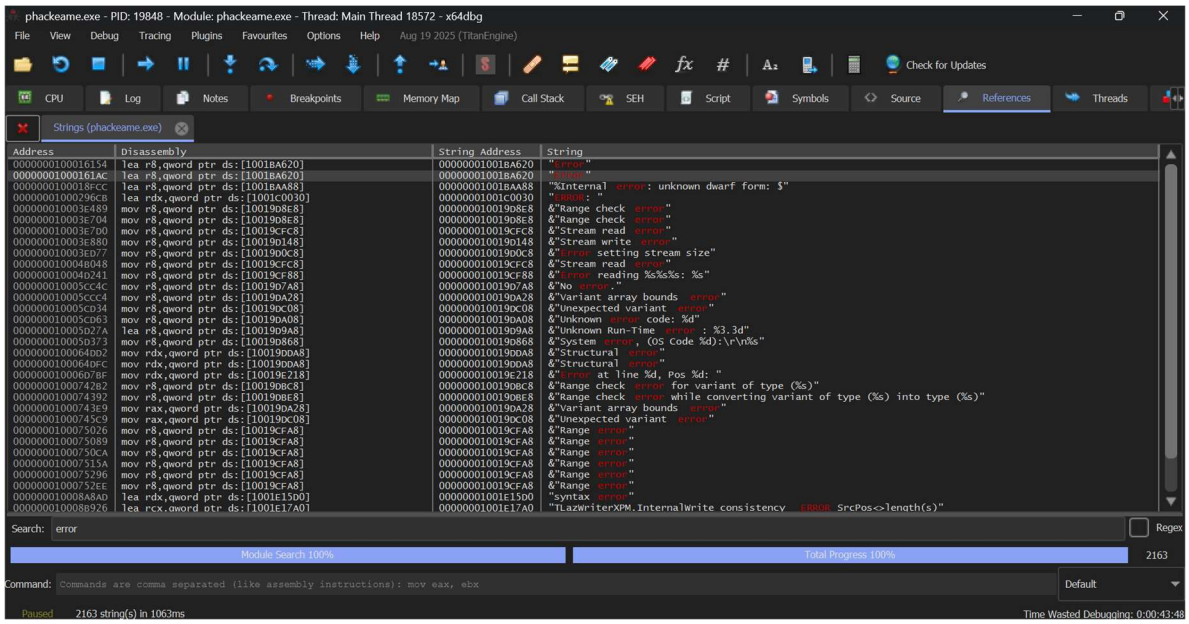
5). Cuando se abra la lista de strings, usamos el filtro para buscar palabras clave: http para localizar la API, incorrect para encontrar el código que rechaza la clave y welcome o bienvenido para ubicar la parte que queremos forzar para permitir el acceso.



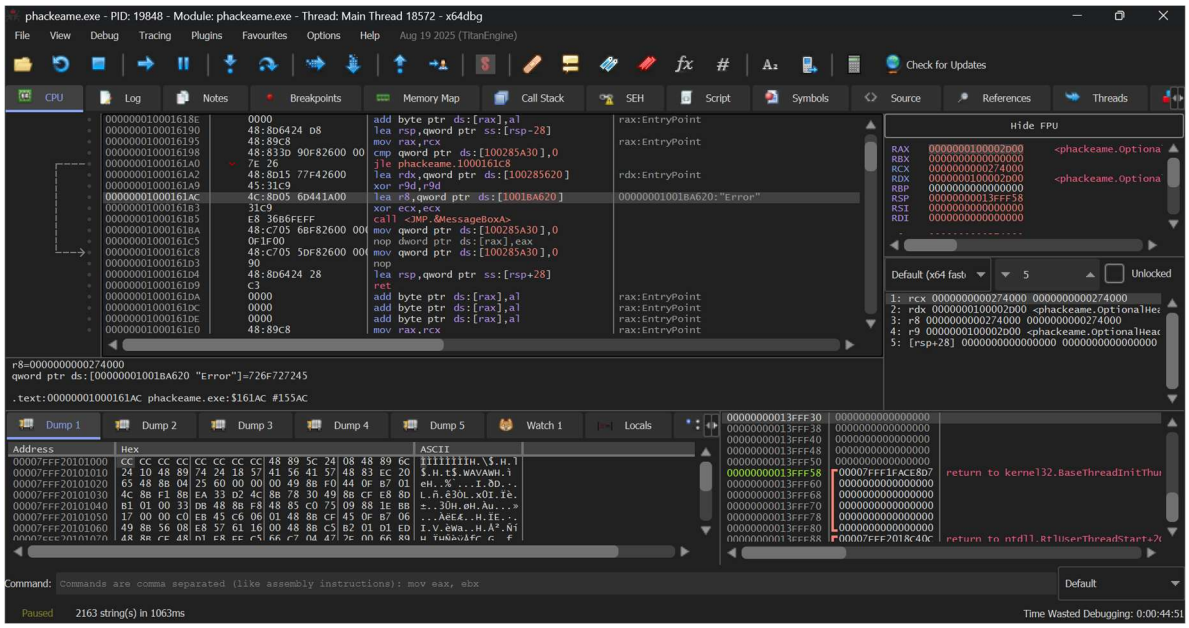
6). localizamos el código de validación escribiendo Error en la barra de búsqueda de la lista de strings, luego damos doble clic en la línea que esté relacionada con mensajes de acceso o login y el debugger nos regresa a la pestaña CPU, donde veremos resaltada la instrucción que prepara el mensaje de error que muestra el programa.



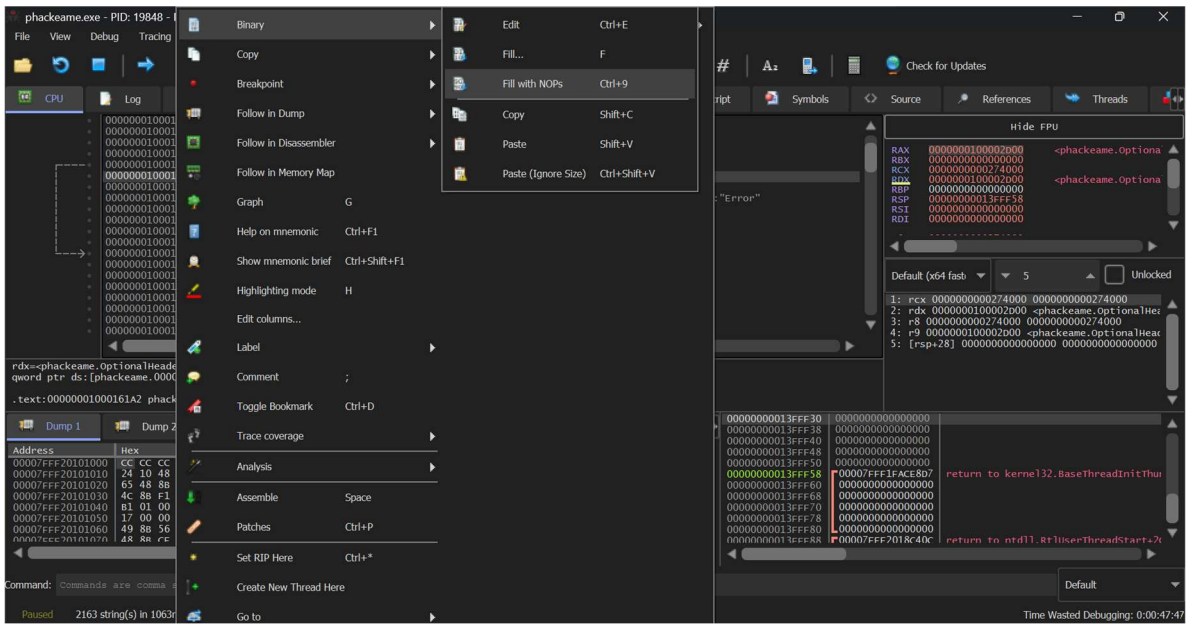
7). Damos doble clic a la segunda opción de la lista, la que tiene la dirección 00000001000161AC y el string “Error”, porque está ubicada en el bloque de memoria 1000..., que es donde Lazarus guarda la lógica personalizada del formulario de login y, por lo tanto, es la que nos interesa analizar.



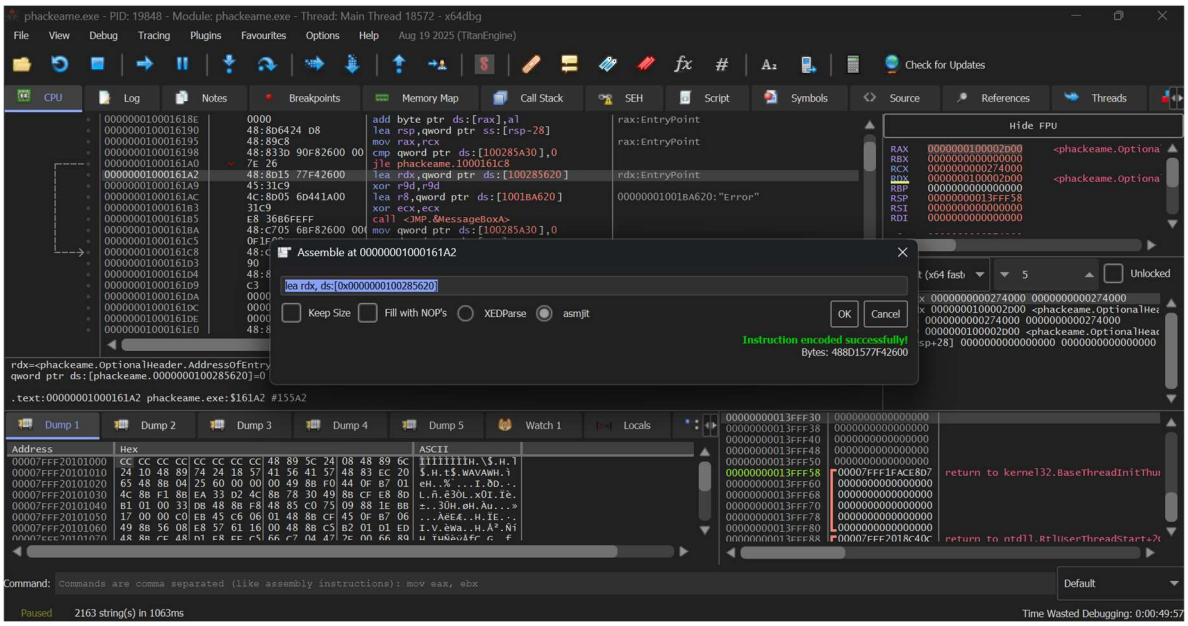
8). Para hacer el parche, en la pestaña CPU buscamos unas líneas arriba hasta encontrar la instrucción de salto (como jne o jle) que apunta al mensaje de error la identificamos por la flecha roja y luego hacemos un clic sobre esa línea (por ejemplo, en la dirección 00000001000161A2) para poder modificarla.

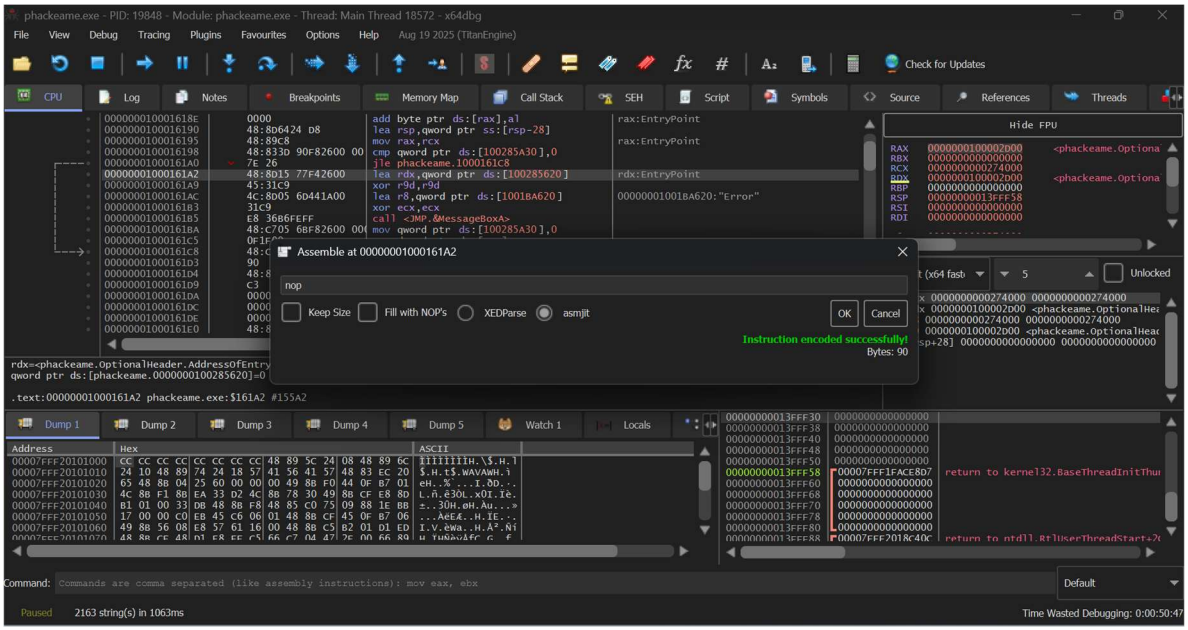


9). Con la línea seleccionada, hacemos clic derecho y elegimos Binary → Fill with NOPs; veremos que el código 7F 26 cambia a 90 90 y ahora aparece como nop, lo que significa que el programa ya no realizará el salto hacia el error y continuará su ejecución hacia la ruta de éxito.

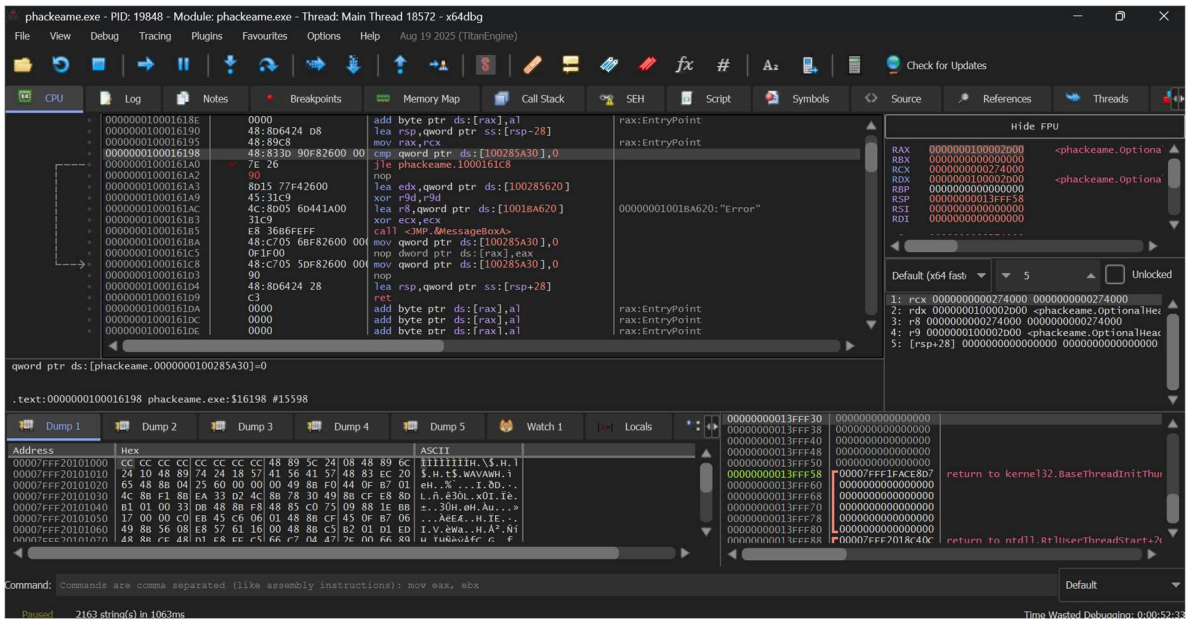


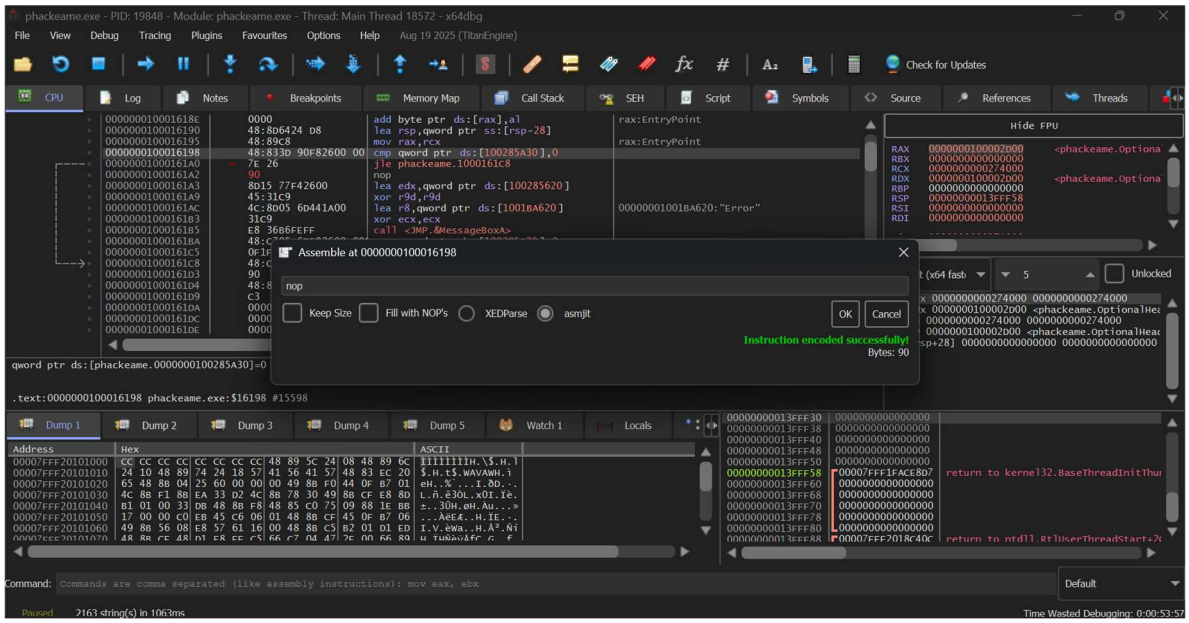
10). En lugar de modificar los bytes manualmente, cerramos la ventana de Size, seleccionamos la línea 00000001000161A2 (jle), presionamos Espacio, escribimos nop en el cuadro Assemble y confirmamos con OK; opcionalmente repetimos el proceso con la línea superior (jne) para eliminar cualquier otra validación.



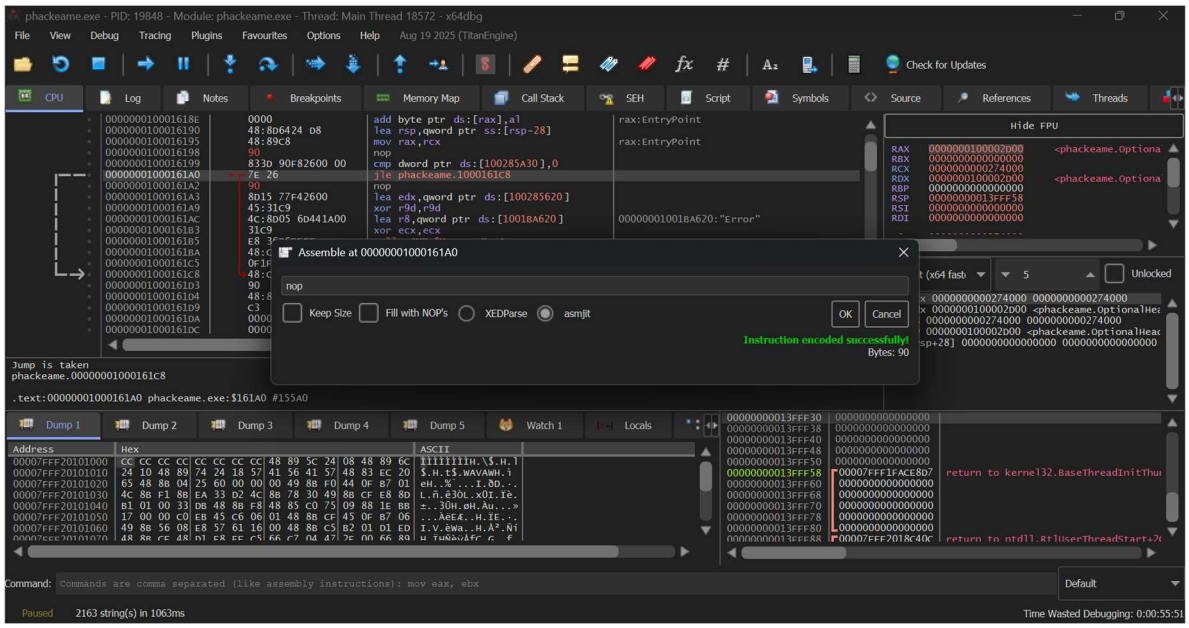


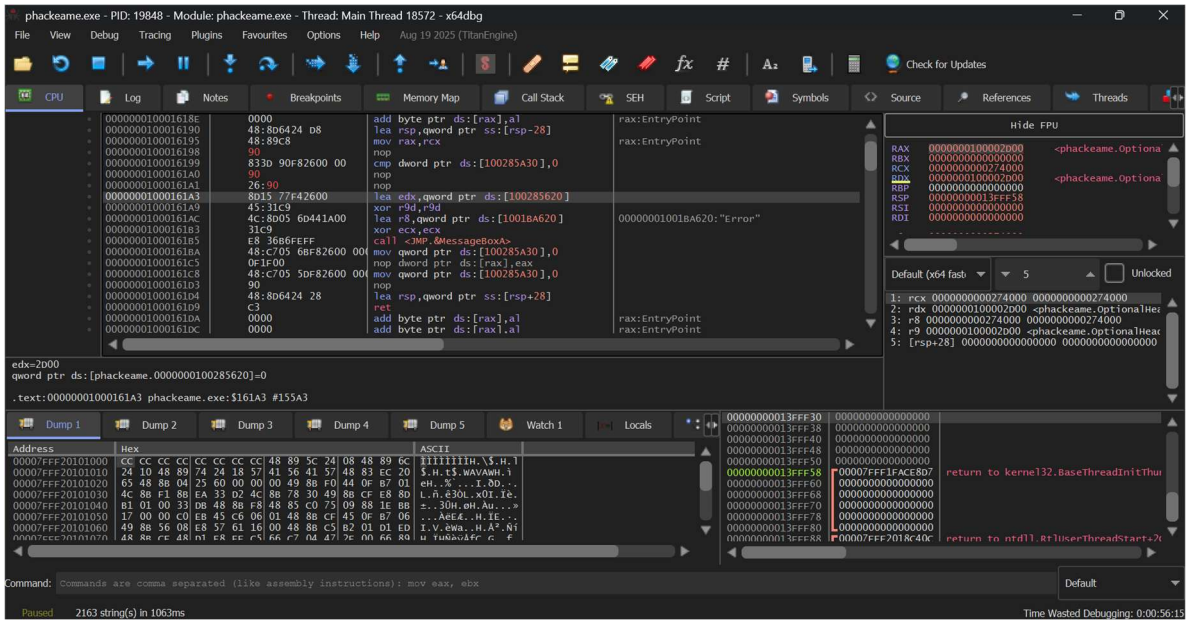
11). Es muy importante anular el segundo “guardia”: en la pestaña CPU, ubicamos unas líneas arriba la dirección 0000000100016198 donde aparece jne phackeame.1000161C8, la seleccionamos, presionamos Espacio y escribimos nop también, con esto nos aseguramos de que, sin importar lo que responda el servidor Python, el programa siempre continúe hacia la ruta de éxito.





12). Debajo de la línea cmp (0000000100016198) ubicamos la instrucción jne en 00000001000161A0, la seleccionamos, presionamos Espacio, escribimos nop y confirmamos; al ver varios nop en rojo, sabemos que el programa ya no podrá saltar al error.





13). Presionamos Ctrl + P para abrir la lista de direcciones modificadas (como 161A2, 161A0, etc.), luego hacemos clic en Patch File y guardamos el ejecutable con un nombre claro, por ejemplo phackeame_FINAL.exe.