EPICODE

CYBERSECURITY COURSE

BY MAX ALDROVANDI

17/05/2024

Web-site: https://epicode.com/it

Locality:
Via dei Magazzini Generali,
6 Roma, Lazio 00154, IT

PRACTICE EXERCISE S11/L3

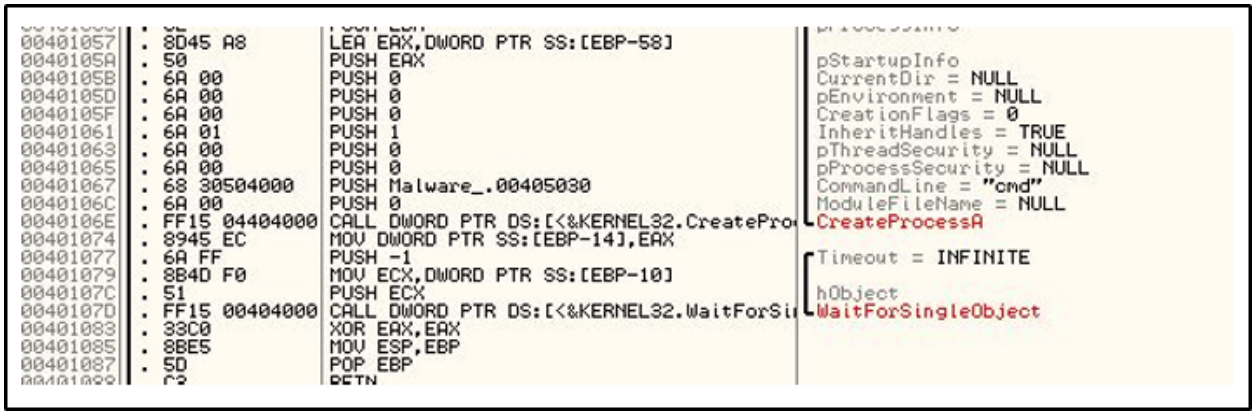
Track:

Refer to the malware: **Malware_U3_W3_L3**, found inside the **Exercise_Practice_U3_W3_L3** folder on the desktop of the virtual machine dedicated to malware analysis. Answer the following questions using OllyDBG.

- At address **0040106E** the **Malwareeffects** a function call to the "**CreateProcess**" function. What is the value of the "**CommandLine**" parameter that is passed on the stack?
- Enter a **breakpointsoftware** at address **004015A3**. What is the value of the EDX register? Perform a "step-into" at this point. Indicate what the value of the EDX register is now by giving reasons for your answer. What instruction was executed?
- Enter a second breakpoint at memory address **004015AF**. What is the value of the ECX register? Execute a step-into. What is now the value of ECX? Explain which instruction was executed.
- **BONUS**: Explain in broad terms how the malware works.

Solution:

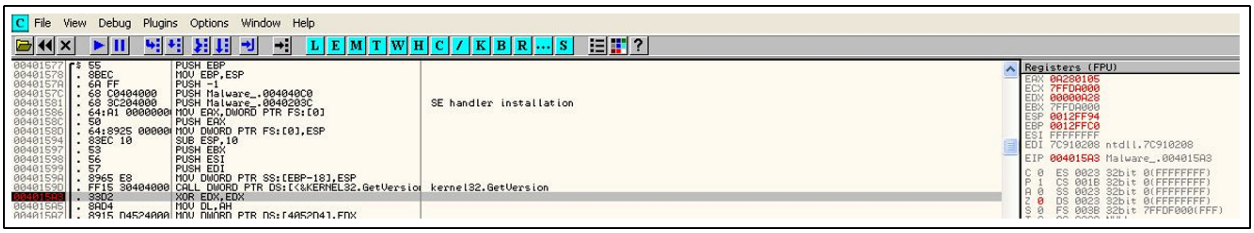
The value of the parameter is "CMD" or the Windows command prompt, as seen in the figure below at 00401067.



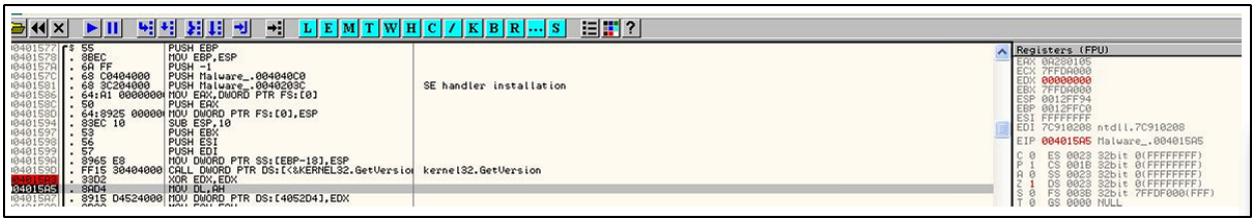
Once the breakpoint is configured, we click on "play", the program will stop at instruction **XOR EDX**,**EDX**. Before the instruction is executed, the register value is "**00000A28**".

After the step-into, the instruction **XOR EDX**,**EDX** is executed, which in fact is equivalent to initializing a variable to zero.

Thus, after step-into, the value of **EDX** will be 0.

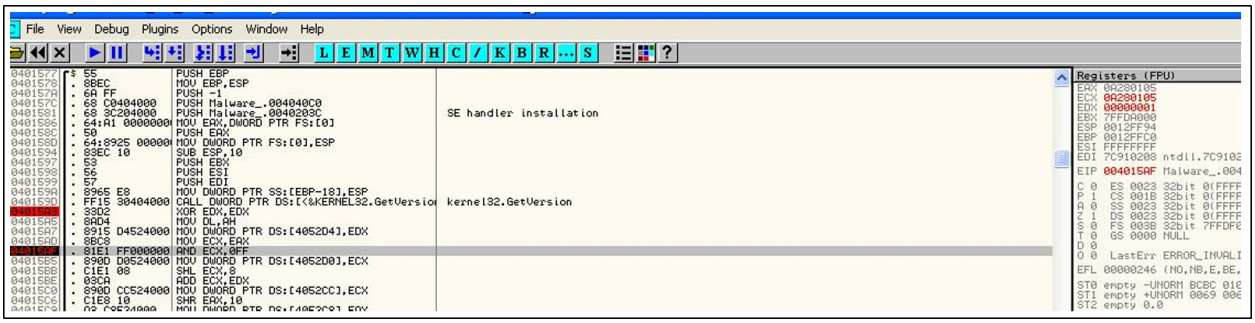


Before



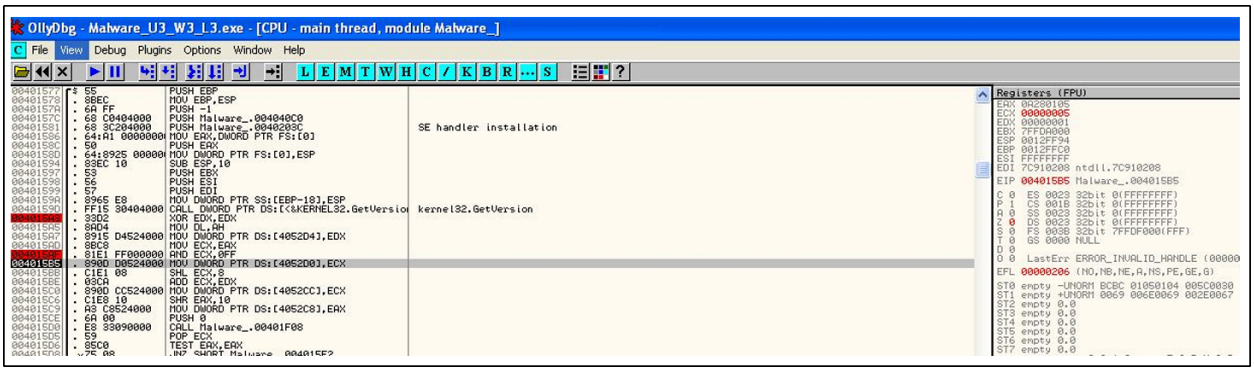
After

We configure the second breakpoint. The value of the **ECX** register is "**0A280105**".



Before

After step-into, the value of the **ECX** register was changed to "**00000005**" as the **AND ECX** instruction was executed, **FF**.



After

In detail, the instruction performs logical **AND** on the bits of **EAX** and the hexadecimal value **FF**. We first bring both values into binary format and then perform logical **AND** between the bits.

Esadecimale	Binario
0A280105	0000 1010 0010 1000 0000 0001 0000 0101
FF	0000 0000 0000 0000 0000 0000 1111 1111
Eseguendo l'AND logico tra i bit uno ad uno	0000 0000 0000 0000 0000 0000 0000 0101

Which in **Hexadecimal** is **00000005**

Here is explained the value of **ECX** after the **AND ECX** instruction, **0FF**.