

Tecnológico de Costa Rica

Escuela de Ingeniería en Computación

Seguridad del Software - IC8071

Tarea 4 - Try Hack Me

Estudiante:

Axel Alexander Chaves Reyes

Profesor:


Herson Esquivel Vargas

Semestre I

2024

Task 1



406




x86 Architecture Overview

A crash course in x86 architecture to enable us in malware reverse engineering.

Help



7%

Task 1  Introduction

Malware often works by abusing the way systems are designed. Therefore, to understand how malware works, it is essential that we know the architecture of the systems they are running in. In this room, we will get a brief overview of the x86 architecture from a malware analysis point of view. Please note that there might be a lot of details of the x86 architecture that we will be skipping over, but that is because they are not related to malware analysis.

Learning Objectives:

Summing up, we will be covering the following topics in this room.

- Overview of CPU architecture and its components
- Different types of CPU registers and their usage
- Memory layout as viewed by a program
- Stack layout and stack registers


So let's dive into the room and learn about the above-mentioned topics.


Answer the questions below


Go through the Learning Objectives

No answer needed

Correct Answer

Task 2  CPU architecture overview

Task 3  Registers overview

Task 4  Registers - Continued


You've started a streak. Keep it up!

ESP
LAA

23:05
4/3/2024

Task 2

tryhackme.com/room/x8654arch



Control Unit:

The Control Unit gets instructions from the main memory, depicted here outside the CPU. The address to the next instruction to execute is stored in a register called the Instruction Pointer or IP. In 32-bit systems, this register is called EIP, and in 64-bit systems, it is called RIP.

Arithmetic Logic Unit:

The arithmetic logic unit executes the instruction fetched from the Memory. The results of the executed instruction are then stored in either the Registers or the Memory.

Registers:

The Registers are the CPU's storage. Registers are generally much smaller than the Main Memory, which is outside the CPU, and help save time in executing instructions by placing important data in direct access to the CPU.

Memory:

The Memory, also called Main Memory or Random Access Memory (RAM), contains all the code and data for a program to run. When a user executes a program, its code and data are loaded into the Memory, from where the CPU accesses it one instruction at a time.

I/O devices:

I/O devices or Input/Output devices are all other devices that interact with a computer. These devices include Keyboards, Mice, Displays, Printers, Mass storage devices like Hard Disks and USBs, etc.

In short, when a program has to be executed, it is loaded into the memory. From there, the Control Unit fetches one instruction at a time using the Instruction Pointer Register, and the Arithmetic Logic Unit executes it. The results are stored in either the Registers or the Memory.

Answer the questions below

In which part of the Von Neumann architecture are the code and data required for a program to run stored?

Memory

Correct Answer

What part of the CPU stores small amounts of data?

Registers

Correct Answer

In which unit are arithmetic operations performed?

Arithmetic Logic Unit

Correct Answer

ESP
LAA

23:10
4/3/2024

Task 3

tryhackme.com/room/x8664arch

the stack. It is also used in conjunction with the Stack Segment register. It is a 32-bit register called EBP in 32-bit systems and a 64-bit register called RBP in 64-bit systems.

ESI or RSI:
This register is called the Source Index register. It is used for string operations. It is used with the Data Segment (DS) register as an offset. It is a 32-bit register called ESI in 32-bit systems and a 64-bit register called RSI in 64-bit systems.

EDI or RDI:
This register is called the Destination Index register. It is also used for string operations. It is used with the Extra Segment (ES) register as an offset. It is a 32-bit register called EDI in 32-bit systems and a 64-bit register called RDI in 64-bit systems.

R8-R15:
These 64-bit general-purpose registers are not present in 32-bit systems. They were introduced in the 64-bit systems. They are also addressable in 32-bit, 16-bit, and 8-bit modes. For example, for the R8 register, we can use R8D for lower 32-bit addressing, R8W for lower 16-bit addressing, and R8B for lower 8-bit addressing. Here, the suffix D stands for Double-word, W stands for Word, and B stands for Byte.

Answer the questions below

Which register holds the address to the next instruction that is to be executed?
Instruction Pointer
Correct Answer

Which register in a 32-bit system is also called the Counter Register?
ECX
Correct Answer Hint

Which registers from the ones discussed above are not present in a 32-bit system?
R8-R15
Correct Answer Hint

Task 4 Registers - Continued

Task 5 Memory overview

Task 6 Stack Layout

ESP LAA 23:15 4/3/2024

Task 4

tryhackme.com/room/x8664arch

RBP, EBP, BP	FS		
RSP, ESP, SP	GS		
RSI, ESI, SI			
RDI, EDI, DI			
R8-R15			

Segment Registers:
Segment Registers are 16-bit registers that convert the flat memory space into different segments for easier addressing. There are six segment registers, as explained below:
Code Segment: The Code Segment (CS) register points to the Code section in the memory.
Data Segment: The Data Segment (DS) register points to the program's data section in the memory.
Stack Segment: The Stack Segment (SS) register points to the program's Stack in the memory.
Extra Segments (ES, FS, and GS): These extra segment registers point to different data sections. These and the DS register divide the program's memory into four distinct data sections.

Answer the questions below

Which flag is used by the program to identify if it is being run in a debugger?
Trap Flag
Correct Answer

Which flag will be set when the most significant bit in an operation is set to 1?
Sign Flag
Correct Answer

Which Segment register contains the pointer to the code section in memory?
Code Segment
Correct Answer

ESP LAA 23:18 4/3/2024

Task 5

tryhackme.com/room/AB664arch

The Code Section, as the name implies, contains the program's code. Specifically, this section refers to the text section in a Portable Executable file, which includes instructions executed by the CPU. This section of the Memory has execute permissions, meaning that the CPU can execute the data in this section of the program memory.

Data:

The Data section contains initialized data that is not variable and remains constant. It refers to the data section in a Portable Executable file. It often contains Global variables and other data that are not supposed to change during the program's execution.

Heap:

The heap, also known as dynamic Memory, contains variables and data created and destroyed during program execution. When a variable is created, memory is allocated for that variable at runtime. And when that variable is deleted, the memory is freed. Hence the name dynamic memory.

Stack:

The Stack is one of the important parts of the Memory from a malware analysis point of view. This section of the Memory contains local variables, arguments passed on to the program, and the return address of the parent process that called the program. Since the return address is related to the control flow of the CPU's instructions, the stack is often targeted by malware to hijack the control flow. You can look at the [Buffer Overflows](#) room to learn how this happens. We will cover more details about the stack in the next task.

Woop woop! Your answer is correct.

Answer the questions below

When a program is loaded into Memory, does it have a full view of the system memory? Y or N?

n Correct Answer

Which section of the Memory contains the code?

Code Correct Answer

Which Memory section contains information related to the program's control flow?

Stack Correct Answer

Task 6 Stack Layout

ESP LAA 23:20 4/3/2024

Task 6

With an address of the malware author's choice, this technique is called a Stack Buffer Overflow.

Arguments:

The Arguments being passed to a function are pushed to the stack before the function starts execution. These arguments are present right below the Return Address on the stack.

Function Prologue and Epilogue:

When a function is called, the stack is prepared for the function to execute. This means that the arguments are pushed to the stack before the start of the function execution. After that, the Return Address and the Old Base Pointer are pushed onto the stack. Once these elements are pushed, the Base Pointer address is changed to the top of the stack (which will be the Stack Pointer of the caller function at that time). As the function executes, the Stack Pointer moves as per the requirements of the function. This portion of code that pushes the arguments, the Return Address, and the Base Pointer onto the Stack and rearranges the Stack and Base Pointers is called the Function Prologue.

Similarly, the Old Base Pointer is popped off the stack and onto the Base Pointer when the function exits. The Return address is popped off to the Instruction Pointer, and the Stack Pointer is rearranged to point to the top of the stack. The part of the code that performs this action is called the Function Epilogue.

Click the View Site button at the top of the task to launch the static site in split view. Now, hop on the attached static site and find the flag by arranging the stack correctly.

Woop woop! Your answer is correct.

Answer the questions below

Follow the instructions in the attached static site and find the flag. What is the flag?

THM{SMASHED_THE_STACK} Correct Answer

Task 7 Conclusion

Created by tryhackme and umairalizafar

This is a free room which means anyone can deploy virtual machines in the room (without being subscribed) 0306 users are in here and this room is 231 days old.

ESP LAA 23:25 4/3/2024

Task 7

100%

Task 1 Introduction

Task 2 CPU architecture overview

Task 3 Registers overview

Task 4 Registers - Continued

Task 5 Memory overview

Task 6 Stack Layout

Task 7 Conclusion

That concludes this room on the primer of x86-64 system architecture. In this room, we learned:

- The Von Neumann CPU architecture
- Different components of a CPU
- Different types of CPU registers
- Memory and its different sections
- Stack layout of a program in memory

Let us know what you think about this room on our [Discord channel](#) or [Twitter account](#). See you around.

Answer the questions below


Join the discussion on our social channels

No answer needed



Correct Answer

Created by [tryhackme](#) and [umairalizaFar](#)

This is a *free* room, which means anyone can deploy virtual machines in the room (without being subscribed)! 9306 users are in here and this room is 231 days old.



ESP
LAA

23:26
4/3/2024