|  |  |  |
| --- | --- | --- |
| Week | Reverse Engineering Malware | Duration |
| 2 | Introduction to Assembly Language | 30 mins |

Marks allocation: 2/100 for CA tutorial submission

**Lesson Objectives**

* Understand basic assembly language

1. Provide examples of three different instruction mnemonics.

ADD, SUB, MOV

2. Explain why the term assembler language is not quite correct.

Assembler is the program that translate your source code into machine code, not a language. A more correct term is "assembly language"

3. Explain the difference between big endian and little endian. Also, look up the origins of this term on the Web.

Little endian places the least significant byte in the lowest memory addressed, on the right side of the number. Big endian does the opposite. The origin of this term actually came from Gulliver's Travels, it refers to resistance to an imperial edict to break soft-boiled eggs on the "little end" escalates to civil war.

4. Why might you use a symbolic constant rather than an integer literal in your code?

An integer literal has no direct meaning to someone reading the program's source code. But a symbolic constant can assign an integer value and is self-documenting, making it easier to use.

5. How are data labels and code labels different?

A code label is followed by a colon, but a data label does not followed by a colon.

6. (True/False): An identifier cannot begin with a numeric digit.

True

7. (True/False): A hexadecimal literal may be written as 0x3A.

False

8. (True/False): Assembly language directives execute at runtime.

False

9. (True/False): Assembly language directives can be written in any combination of uppercase and lowercase letters.

True

10. Name the four basic parts of an assembly language instruction.

Label, mnemonic, operand(s), comment

11. (True/False): MOV is an example of an instruction mnemonic.

True

12. (True/False): A code label is followed by a colon (:), but a data label does not end with a colon.

True

END