# S10/L3

Stefano Di Prospero
NetRaiders

TRACCIA:

*Nella lezione teorica del mattino, abbiamo visto i fondamenti del linguaggio Assembly. Dato il codice in Assembly per la CPU x86 allegato qui di seguito, identificare lo scopo di ogni istruzione, inserendo una descrizione per ogni riga di codice. Ricordate che i numeri nel formato 0xYY sono numeri esadecimali. Per convertirli in numeri decimali utilizzate pure un convertitore online, oppure la calcolatrice del vostro computer (per programmatori).*

**1. 0x00001141 <+8>: mov EAX,0x20**
**2. 0x00001148 <+15>: mov EDX,0x38**
**3. 0x00001155 <+28>: add EAX,EDX**
**4. 0x00001157 <+30>: mov EBP, EAX**
**5. 0x0000115a <+33>: cmp EBP,0xa**
**6. 0x0000115e <+37>: jge 0x1176 <main+61>**
**7. 0x0000116a <+49>: mov EAX,0x0**
**8. 0x0000116f <+54>: call 0x1030 <printf@plt>**

1. **0x00001141 <+8>: mov EAX,0x20**

- 0x00001141, indica l'indirizzo di memoria
- <+8>, indica la posizione in cui si trova, ovvero a 8 bit dall'inizio della funzione
- Questa istruzione sposta il valore esadecimale 0x20, corrispondente a 32 in decimale, nel registro EAX.

2. **0x00001148 <+15>: mov EDX,0x38**

- 0x00001148, indirizzo di memoria
- <+15>, posizione , ovvero 15 bit
- Questa istruzione sposta il valore esadecimale 0x38, corrispondente a 56 in decimale, nel registro EDX.

3. **0x00001155 <+28>: add EAX,EDX**

   - 0x00001155 <+28>, indirizzo e posizione
   - Questa istruzione somma il valore contenuto in EDX al valore contenuto in EAX e memorizza il risultato in EAX.

4. **0x00001157 <+30>: mov EBP, EAX**
   - 0x00001157 <+30>, indirizzo e posizione
   - Questa istruzione sposta il valore contenuto in EAX nel registro EBP.

5. **0x0000115a <+33>: cmp EBP,0xa**

   - 0x0000115a <+33>, indirizzo e posizione
   - Questa istruzione confronta il valore contenuto in EBP con il valore esadecimale 0xa che corrisponde a 10 in decimale.

6. **0x0000115e <+37>: jge 0x1176 <main+61>**

   - 0x0000115e <+37>, indirizzo e posizione
   - Questa istruzione indica che il programma salta all'indirizzo 0x1176, che corrisponde a 61 byte dall'inizio della funzione main, se il risultato del confronto precedente è maggiore o uguale a zero.

7. **0x0000116a <+49>: mov EAX,0x0**

   - 0x0000116a <+49>, indirizzo e posizione
   - Questa istruzione sposta il valore 0x0 che corrisponde a 0 in decimale nel registro EAX.

8. **0x0000116f <+54>: call 0x1030 <printf@plt>**

   - 0x0000116f <+54>, indirizzo e posizione
   - Questa istruzione chiama la funzione `printf` all'indirizzo `0x1030`.