Studio funzioni base dell'assembly

Nella lezione di oggi abbiamo visto alcune delle funzioni di base di assembly, pertanto andiamo ad analizzare le seguenti linee di codice.

0x00001141 <+8>: mov EAX,0x20

sposta il valore immediato **32** (in esadecimale 20) all'interno del del **registro** "general purpose" **EAX**

0x00001148 <+15>: mov EDX,0x38

sposta il valore immediato 56 all'interno del registro EDX

0x00001155 <+28>: add EAX,EDX

va a **sommare** il valore contenuto in **EDX** all'interno di **EAX** e va ad **aggiornare EAX** al nuovo valore (seguendo l'esercizio 32 + 56 = 88, quindi in EAX sarà ora contenuto 88)

0x00001157 <+30>: mov EBP, EAX

va a **spostare** il valore di EAX nel **registro EBP**

0x0000115a <+33>: cmp EBP,0xa

va a **comparare** (**senza sovrascrivere**) il valore presente in EBP con il valore 10 e se trova i due valori uguali va a mettere la **zero flag(zf)** 0 uguale a 1, se EBP (che in questo caso è la **destinazione**) fosse minore di 10 (che in questo caso è la **sorgente**) avremmo la **carry flag(cf)** uguale a 1, se la destinazione fosse maggiore della sorgente uguale a 0.
In questo caso chiede la comparazione tra 88 e 10. essendo 88 maggiore di 10 la zf è 0 e la cf è 0

0x0000115e <+37>: jge 0x1176 <main+61>

jge è il comando di **jump condizionale**, la sua attuale **condizione** è che la destinazione della precedente comparazione sia maggiore o uguale a 0, quindi che la cf sia uguale a 0. se questa condizione si rilevasse vera all'ora **salteremo** all'operazione contenuta nello **slot di memoria** specificato

0x0000116a <+49>: mov eax,0x0

sposta il valore 0 all'interno del **registro** EAX

0x0000116f <+54>: call 0x1030 <printf@plt>
va a **chiamare** la funzione printf@plt con il relativo stack di memoria