ESERCIZIO S10-L3

Lo scopo dell'esercizio di oggi è quello di identificare lo scopo di ogni istruzione riguardo il seguente codice assembly:

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

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

0x00001155 <+28>: add EAX,EDX

0x00001157 <+30>: mov EBP, EAX

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

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

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

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

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

Questa riga di istruzione assegna 0x20=32 a EAX

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

Assegna 0x38=56 a EDX

0x00001155 <+28>: add EAX,EDX

Questa riga esegue la somma di 32+56=88

0x00001157 <+30>: mov EBP, EAX

Assegna il valore 88 da EAX a EBP

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

Confronta tra 88 (destinazione) – 10 (sorgente) = 78 l'istruzione "cmp" è simile all'istruzione "sub", ma a differenza di quest'ultima non modifica gli operandi. Tuttavia, l'operazione "cmp" modifica i flag ZF (zero flag) e CF (carry flag, che si utilizza per gestire eventuali riporti in un'operazione aritmetica).

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

La funzione "jge" è una condizione. Se "jge" viene eseguita, significa che il confronto precedente (cmp EBP, 0xa) risulta vero, cioè il valore è maggiore o uguale a 10. In questo caso, il flusso del programma salterà all'indirizzo di memoria 0x1176, che è l'inizio di un blocco di codice etichettato come <main+61>.

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

Assegna 0 a EAX

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

Richiama la funzione "printf".