Analisi Assembly Language

Identificare lo scopo di ogni istruzione

0×00001141 <+8>: mov EAX, 0×20
- Attribuisce il valore 0x20 (32) al registro EAX

Registro	Esadecimale	Decimale
EAX	Ox20	32

0×00001148 <+15>: mov EDX,0×38
- Attribuisce il valore 0x38 (56) al registro EDX

Registro	Esadecimale	Decimale
EDX	Ox38	56
EAX	Ox20	32

0×00001155 <+28>: add EAX,EDX
- Aggiunge il valore di EDX ad EAX

Registro	Esadecimale	Decimale
EAX	Ox58	88
EDX	Ox38	56

0x00001157 <+30>: mov EBP, EAX
- Sposta il valore di EAX sullo stack, nell EBP (base pointer)

Registro	Esadecimale	Decimale
EBP	Ox58	88
EDX	Ox38	56
EAX	Ox58	88

0×0000115a <+33>: cmp EBP, 0xa
- Compara il valore sul base pointer con il valore 0xa (10) eseguendo una sottrazione

Se il risultato = 0 lo ZF viene impostato a uno e il CF a 0
Se il risultato > 0 ZF impostato a 0 e CF impostato a 0
Se il risultato < 0 ZF impostato a 0 e CF impostato a 1

Zero Flag e Carry Flag sono contenuti nel registro EFLAGS (status flag) e servono per gestire operazioni aritmetiche

0x0000115e <+37>: jge 0×1176 <main+61>
- Se il risultato è uguale o maggiore a 0, lo zero flag è 1 e di conseguenza fa un jump verso l'istruzione fornita -> 0x1176 (4470)

0×0000116a <+49>: mov eax, 0x0
- eax assume il valore 0x0 (0)

Registro	Esadecimale	Decimale
EAX	Ox0	0

0×0000116f <+54>: call 0x1030 <printf@plt>
- Chiamata di funzione all istruzione 0x1030 (4144)

• Legenda:
- 0×0000116a -> istruzione di memoria
- <+49> -> offset di byte = indica quanti byte sono stati utilizzati dalla prima istruzione
- <main+61> -> main è la funziona principale
- <printf@plt> -> é associato alla funzione printf.printf@plt e fa riferimento alla Procedure Linkage Table (PLT) utilizzata in molti sistemi Unix-like per gestire le chiamate a funzioni dinamiche