Malware analysis: Assembly x86
S10/13

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

```
0×00001141 <+8>:    mov    EAX,0×20
0×00001148 <+15>:   mov    EDX,0×38
0×00001155 <+28>:   add    EAX,EDX
0×00001157 <+30>:   mov   EBP, EAX
0×0000115a <+33>:   cmp   EBP,0xa
0×0000115e <+37>:   jge     0×1176 <main+61>
0×0000116a <+49>:   mov   eax,0×0
0×0000116f <+54>:    call    0×1030 <printf@plt>
```

Svolgimento

```
0×00001141 <+8>:    mov    EAX,0×20
0×00001148 <+15>:   mov    EDX,0×38
```

Descrizione: Copia e sposta il valore dell'indirizzo (32 e 56) di memoria specificato nel registro EAX e EDX.

```
0×00001155 <+28>:   add    EAX,EDX
```

Descrizione: Somma i valori EAX, EDX

```
0×00001157 <+30>:   mov   EBP, EAX
```

Descrizione: Muove il contenuto EAX in EBP

```
0×0000115a <+33>:   cmp   EBP,0xa
0×0000115e <+37>:   jge     0×1176 <main+61>
0×0000116a <+49>:   mov   eax,0×0
0×0000116f <+54>:    call    0×1030 <printf@plt>
```

Descrizione: All'inizio controlla l'uguaglianza tra i valori, poi effettua un salto, sovrascrive il valore di EAX con il valore 0 e infine chiama la funzione.