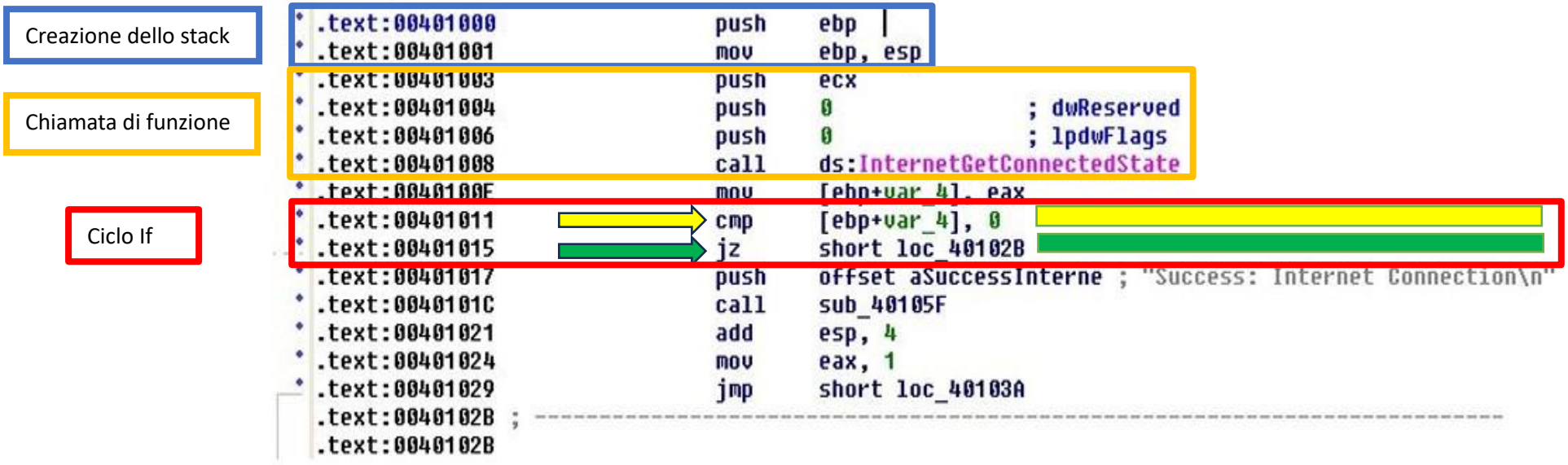
S10/L4 Esercizio 30/11/2023

Nell’esercizio odierno ci viene richiesto di identificare i costrutti noti del seguente estratto di un codice malware:



Creazione dello stack:

La creazione dello stack si riferisce all'organizzazione e alla gestione della memoria stack, in questo caso possiamo vedere l’istruzione di “**push**” che inserisce un valore nello stack e sucessivamente lo sposta attraverso l’istruzione “**mov**”. La creazione dello stack è particolarmente importante durante le chiamate di funzione. Prima di chiamare una funzione, i parametri vengono spesso caricati nello stack

Chiamata di funzione:

L'istruzione "push" va a preparare i dati per una chiamata di funzione mentre "**call**" viene utilizzata per effettuare la chiamata di funzione. Questa istruzione salva l'indirizzo di ritorno nello stack e trasferisce il controllo all'inizio della funzione chiamata

Ciclo if

Il costrutto certamente presente in questo estratto di codice è il ciclo "**if**", lo si può capire dal dalle diciture ".text:00401011 cmp [ebp+var_4], 0" (in giallo) e ".text:00401015 jz short loc_40102B" (in verde), infatti possiamo notare che l'istruzione "jz loc" (jz sta per "jump if zero") salta alla locazione di memoria specificata e "short" è definibile come un salto breve. Quindi, se il risultato dell'istruzione precedente (cmp [ebp+var_4], 0) è zero, allora il salto condizionato jz viene eseguito. Pertanto la linea in questione può essere considerata parte di una struttura di controllo di flusso condizionale che dice: "se [ebp+var_4] è zero, esegui le istruzioni a loc_40102B".

Specifiche delle istruzioni:

push ebp (salva il valore del registro base EBP nello stack)

mov ebp, esp (sposta il valore del registro ESP nel registro base EBP)

push ecx (salva il valore del registro ECX nello stack)

push 0 ; dwReserved (l'istruzione push 0 indica che il valore zero viene inserito nello stack. Questo valore sembra associato ad un parametro denominato dwReserved)

push 0 ; lpdwFlags (salva il valore zero nello stack. Probabilmente lodwFlags verrà impostato a zero)

call ds:InternetGetConnectedState ("call è un'istruzione che esegue una chiamata quindi questa istruzione indica che il programma sta chiamando la funzione "InternetGetConnectedState", che viene utilizzata per determinare se il sistema è connesso ad internet)

mov [ebp+var_4], eax (sposta il valore contenuto nel registro EAX in una posizione specifica dello stack – [ebp+var_4] – dove EBP rappresenta il registro e var_4 la posizione)

cmp [ebp+var_4], 0 (confronta il valore memorizzato nella variabile locale – [ebp+var_4] – con zero, qui inizia il ciclo if)

jz short loc_40102B ("jump if zero" rappresenta un'istruzione di salto condizionato – e qui finisce il ciclo if – quindi, come detto, a seguito della precedente istruzione, questa istruzione effettuerà un breve salto - "short" – essendo zero il risultato dell'istruzione precedente)

push offset aSuccessInterne ; "Success: Internet Connection\n" (questa istruzione inserisce l'indirizzo di memoria della stringa "Success:Internet Connection\n" nello stack)

call sub_40105F (questa istruzione è utilizzata per chiamare una subroutine etichettata come 40105F)

add esp, 4 (aggiunge il valore 4 al registro ESP. In questo caso, dopo la chiamata, può deallocare lo spazio sullo stack per liberarlo)

mov eax, 1 (assegna il valore 1 al registro EAX)

jmp short loc_40103A (indica un salto breve che trasferisce il programma all'indirizzo 40103A