Traccia:

Con riferimento al codice presente nelle slide successive, rispondere ai seguenti quesiti:

1. Spiegate, motivando, quale salto condizionale effettua il Malware.

2. Disegnare un diagramma di flusso (prendete come esempio la visualizzazione grafica di IDA) identificando i salti condizionali (sia quelli effettuati che quelli non effettuati). Indicate con una linea verde i salti effettuati, mentre con una linea rossa i salti non effettuati.

3. Quali sono le diverse funzionalità implementate all'interno del Malware?

4. Con riferimento alle istruzioni «call» presenti in tabella 2 e 3, dettagliare come sono passati gli argomenti alle successive chiamate di funzione . Aggiungere eventuali dettagli tecnici/teorici.

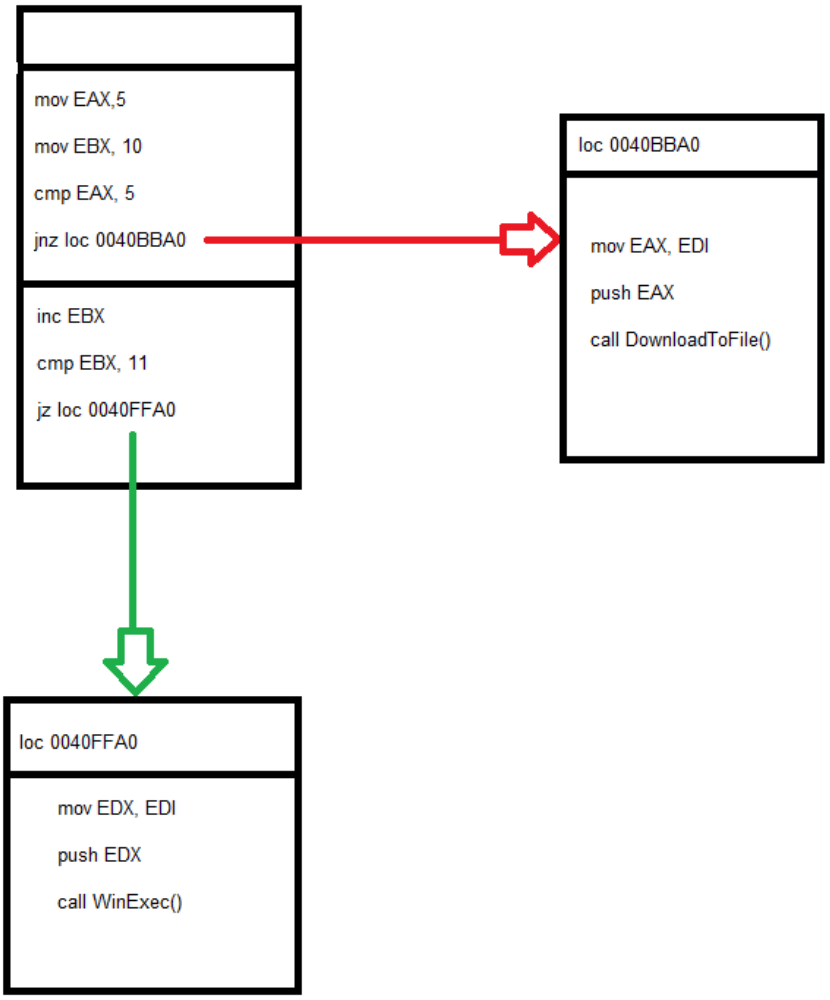
Soluzione:

Salto condizionale:

Primo salto condizionale 00401048 cmp EAX, 5 L'istruzione cmp confronta il valore 5 con il contenuto del registro EAX, senza modificarli direttamente, ma imposta i flag ZF e CF in base al risultato del confronto. Se il confronto tra il valore 5 e il contenuto del registro EAX è uguale, allora il flag ZF sarà settato a 1 e il flag CF sarà settato a 0. 00401058 jnz loc 0040BBA0 L'istruzione salta alla locazione di memoria se ZF è diverso da 1. In questo caso, non viene effettuato il salto poiché gli operandi sono uguali.

Secondo salto condizionale 00401064 cmp EBX, 11 L'istruzione confronta il valore 11 con il contenuto del registro EBX, senza modificarli direttamente, ma imposta i flag ZF e CF in base al risultato del confronto. Se il confronto tra il valore 11 e il contenuto del registro EBX è uguale, allora il flag ZF sarà settato a 1 e il flag CF sarà settato a 0. 00401068 jnz loc 0040FFA0 L'istruzione salta alla locazione di memoria se ZF è uguale a 1. In questo caso, viene effettuato il salto poiché gli operandi sono uguali.

Diagramma di flusso:



Funzionalità del codice:

Dopo un'attenta analisi del codice preso in esame, si evince la presenza di un tipo di malware downloader che manda in esecuzione un ransomware.

Le chiamate di funzione presenti sono:
>**DownloadToFile()** per scaricare bit da internet e salvarli in un file (anche se in realtà se si osserva bene il diagramma di flusso non entra in esecuzione).

>**WinExec()** per eseguire il malware precedentemente scaricato.

Chiamate di funzione:

Di solito il passaggio degli argomenti avviene grazie all’istruzione push, il problema è che nel codice fornito non sono presenti istruzioni per la creazione e rimozione dello stack.
Comunque essendo che si tratta di un'architettura x86 su un sistema Windows molto probabilmente si tratta di call functions con convenzione STDCALL.
Spiegazione STDCALL In una convenzione stdcall, i parametri vengono passati alla funzione attraverso lo stack, e la responsabilità di ripulire lo stack dopo il completamento della funzione è della funzione chiamata stessa.

BONUS:

EDI è un registro che contiene l’indirizzo di destinazione nel caso di operazioni con stringhe.
Dopo aver analizzato attentamente il codice mi sono reso conto che il codice è incompleto perché manca l’utilizzo del registro ESI dove è allocato l’indirizzo del codice sorgente per la manipolazione delle stringhe.
Di conseguenza viene utilizzato “DF” (Direction Flag) che viene utilizzato per determinare la direzione con l’utilizzo delle istruzioni “cld” e “std”.