

S10 – L5

Traccia

Con riferimento al file “Malware_U3_W2_L5” presente all’interno della cartella «Esercizio_Pratico_U3_W2_L5» sul desktop della macchina virtuale dedicata per l’analisi dei malware, rispondere ai seguenti quesiti:

1. Quali librerie vengono importate dal file eseguibile?
2. Quali sono le sezioni di cui si compone il file eseguibile del malware?

Con riferimento alla figura nella slide successiva, rispondere ai seguenti quesiti:

3. Identificare i costrutti noti (creazione dello stack, eventuali cicli, altri costrutti).
4. Ipotesizzare il comportamento della funzionalità implementata
5. BONUS fare tabella con significato delle singole righe di codice assembly



```
push    ebp
mov     ebp, esp
push    ecx
push    0          ; dwReserved
push    0          ; lpdwFlags
call    ds:InternetGetConnectedState
mov     [ebp+var_4], eax
cmp     [ebp+var_4], 0
jz      short loc_40102B
```

```
push    offset aSuccessInterne ; "Success: Internet Connection\n"
call    sub_40117F
add     esp, 4
mov     eax, 1
jmp     short loc_40103A
```

```
loc_40102B:          ; "Error 1.1: No Internet\n"
push    offset aError1_1NoInte
call    sub_40117F
add     esp, 4
xor     eax, eax
```

```
loc_40103A:
mov     esp, ebp
pop     ebp
retn
sub_401000 endp
```



Cos'è un malware?

Cos'è un malware?

Un malware (abbreviazione di "malicious software") è un software progettato per danneggiare, disturbare, rubare informazioni o ottenere accesso non autorizzato a sistemi informatici. I malware possono essere utilizzati da cybercriminali per vari scopi, inclusi il furto di dati, il sabotaggio di sistemi e anche la monetizzazione tramite estorsione (come nel caso dei ransomware). La diffusione dei malware è una minaccia costante per la sicurezza informatica e richiede una costante vigilanza e aggiornamento delle difese.

Tipi di malware

- 1** **Virus**: software che si replica inserendosi in altri programmi o file e si diffonde quando questi vengono eseguiti. I virus possono danneggiare i file o i sistemi in cui si replicano.
- 2** **Worm**: malware che si replica autonomamente senza bisogno di un file ospite, spesso attraverso reti. I worm possono causare rallentamenti o blocchi nei sistemi e nelle reti.
- 3** **Trojan Horse**: software che appare legittimo ma contiene funzioni dannose. I trojan possono creare backdoor per l'accesso non autorizzato o rubare dati sensibili.
- 4** **Ransomware**: malware che cripta i dati della vittima e richiede un riscatto per ripristinare l'accesso. Questo tipo di malware è particolarmente pericoloso per le aziende, poiché può paralizzare operazioni critiche.
- 5** **Spyware**: software che raccoglie informazioni sull'utente senza il suo consenso. Può monitorare l'attività online, rubare informazioni di accesso e raccogliere dati personali.

Tipi di malware

- 6 **Adware**: malware che mostra annunci pubblicitari indesiderati. Anche se meno dannoso, l'adware può compromettere la privacy e le prestazioni del sistema.
- 7 **Rootkit**: software progettato per ottenere e mantenere l'accesso amministrativo nascosto su un sistema. I rootkit possono essere particolarmente difficili da rilevare e rimuovere.
- 8 **Keylogger**: malware che registra tutto ciò che viene digitato sulla tastiera. Sono spesso utilizzati per rubare informazioni sensibili come password e numeri di carte di credito.
- 9 **Botnet**: una rete di computer infetti controllati centralmente da un malintenzionato. Possono essere utilizzati per attacchi distribuiti come il DDoS (Distributed Denial of Service).
- 10 **Backdoor**: strumento che permette l'accesso remoto non autorizzato a un sistema. Possono essere installate da altri tipi di malware per mantenere l'accesso persistente.

Analisi sui malware

Tipi di analisi sui malware

Statica

Basica

Analisi del codice del malware, le sue strutture e le sue funzionalità attraverso disassemblatori, decompilatori, strumenti di firma digitale, analizzatori di file binari. Ispezione del codice sorgente, analisi delle librerie e delle dipendenze, estrazione di stringhe, analisi delle API utilizzate. Identificazione delle funzionalità di base del malware per ottenere una comprensione preliminare delle sue capacità.

Avanzata

Analisi del codice del malware, le sue strutture e le sue funzionalità attraverso disassemblatori avanzati (come IDA Pro), deobfuscatori, strumenti per l'analisi del flusso di controllo. Analisi approfondita delle tecniche di offuscamento, decodifica delle routine crittografiche, studio dettagliato del comportamento del codice. Identificazione del funzionamento interno del malware e le sue componenti critiche.

Malware non eseguito

Dinamica

Basica

Esecuzione del malware in un ambiente controllato per osservare il suo comportamento in tempo reale attraverso sandboxes, strumenti di monitoraggio dei processi e del traffico di rete. Osservazione delle modifiche al file system, monitoraggio delle chiamate di rete, analisi del comportamento del processo in memoria, registrazione delle attività del malware.

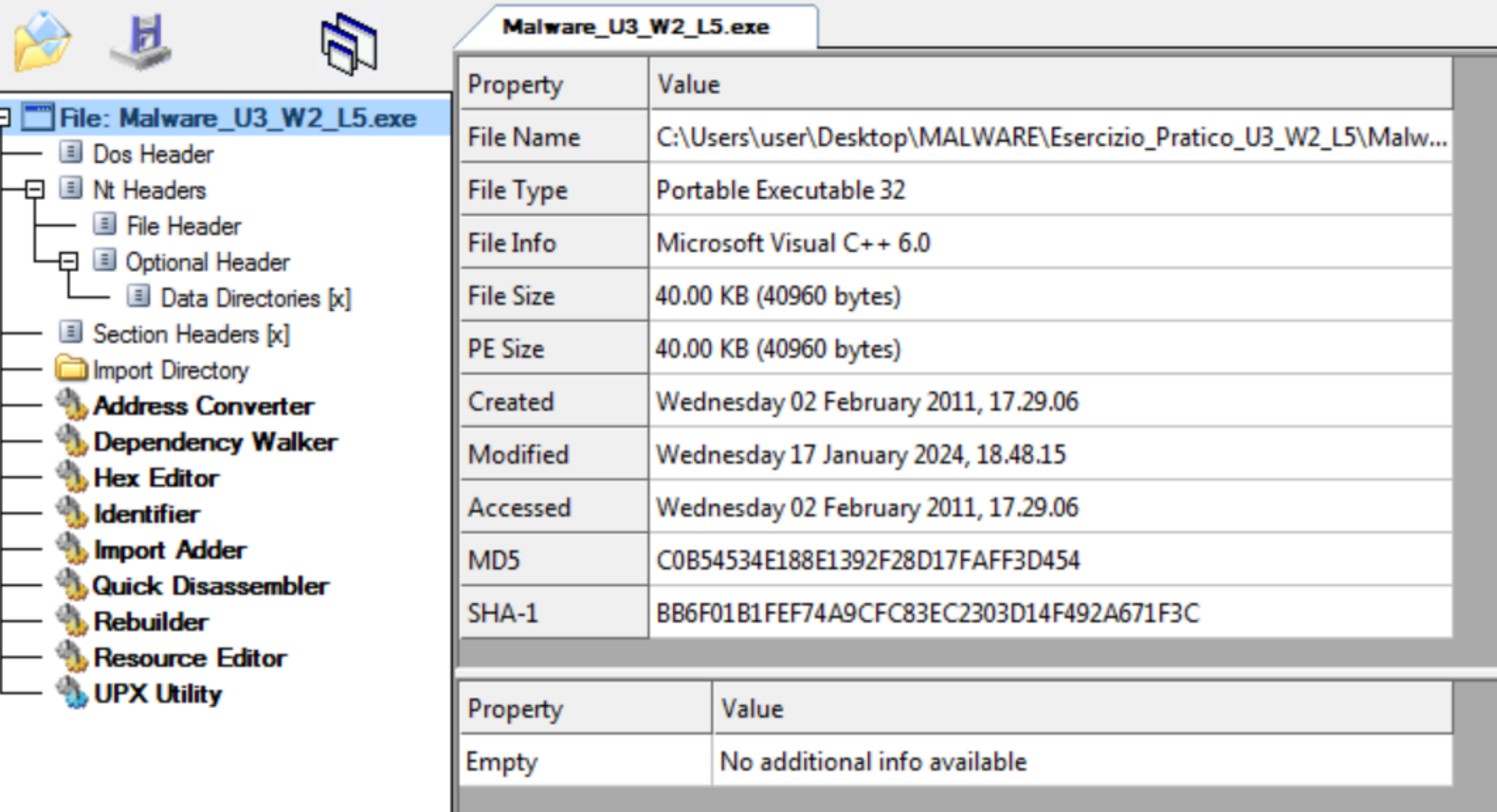
Avanzata

Esecuzione del malware in un ambiente controllato per osservare il suo comportamento in tempo reale attraverso debugger avanzati (come OllyDbg), strumenti per la reverse engineering del traffico di rete, monitoraggio dettagliato delle API di sistema. Debugging passo-passo del malware, analisi del comportamento dinamico sotto varie condizioni, studio delle tecniche di evasione e anche vulnerabilità nel malware stesso.

Malware eseguito

Malware_U3_W2_L5

Grazie a CFF Explorer, possiamo effettuare un'analisi statica del malware. Nella figura possiamo notare diverse caratteristiche come la data di creazione e modifica che possono aiutare a capire il contesto temporale del file, mentre le informazioni sul tipo di file e le dimensioni possono fornire indizi sul comportamento e l'origine del malware. In particolare utilizziamo gli hash MD5 e SHA-1 per identificare univocamente il file e confrontarlo con altre minacce note.



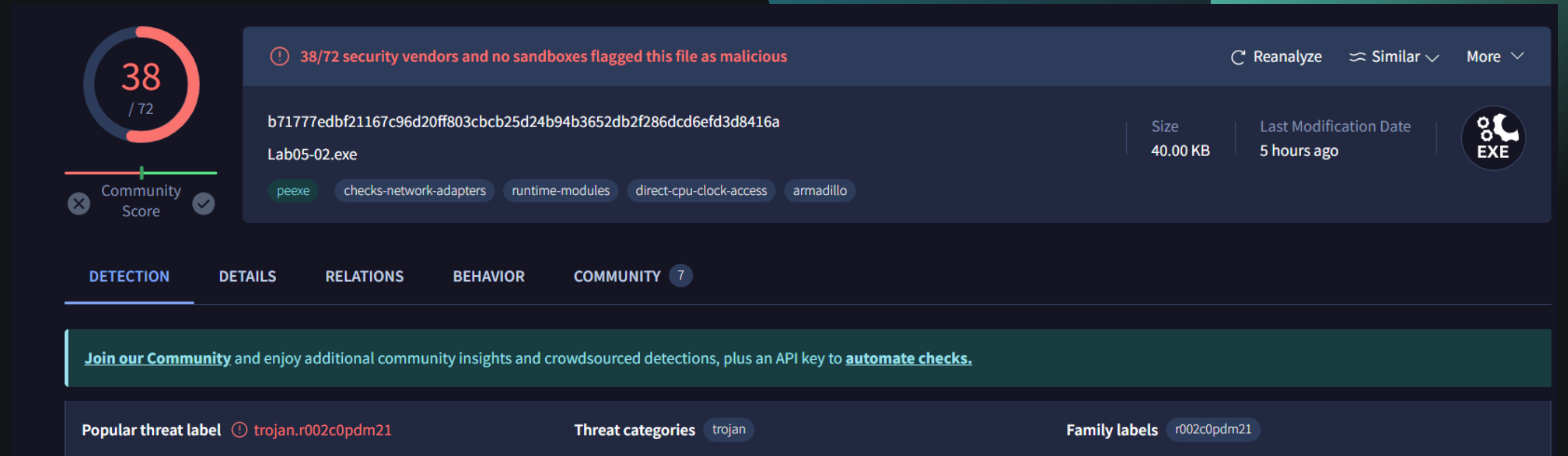
The screenshot displays the CFF Explorer application window. The left pane shows a tree view of the file's structure, including headers, sections, and various utilities. The right pane shows the 'Properties' tab for the selected file, 'Malware_U3_W2_L5.exe'.

Property	Value
File Name	C:\Users\user\Desktop\MALWARE\Esercizio_Pratico_U3_W2_L5\Malw...
File Type	Portable Executable 32
File Info	Microsoft Visual C++ 6.0
File Size	40.00 KB (40960 bytes)
PE Size	40.00 KB (40960 bytes)
Created	Wednesday 02 February 2011, 17.29.06
Modified	Wednesday 17 January 2024, 18.48.15
Accessed	Wednesday 02 February 2011, 17.29.06
MD5	C0B54534E188E1392F28D17FAFF3D454
SHA-1	BB6F01B1FEF74A9CFC83EC2303D14F492A671F3C

Property	Value
Empty	No additional info available

Malware_U3_W2_L5

Effettivamente, ricercando l'hash su VirusTotal, possiamo vedere che il file è stato già identificato come un Trojan con un rilevamento di 38 su 72 (una significativa maggioranza di antivirus).



The screenshot displays the VirusTotal analysis interface for the file `Lab05-02.exe`. On the left, a circular progress indicator shows a detection score of 38 out of 72, with a 'Community Score' bar below it. A warning message states: '38/72 security vendors and no sandboxes flagged this file as malicious'. The file's SHA-256 hash is `b71777edbf21167c96d20ff803cbcb25d24b94b3652db2f286dcd6efd3d8416a`. Metadata includes a size of 40.00 KB and a last modification date of 5 hours ago. The file icon is labeled 'EXE'. Below the header, tabs for 'DETECTION', 'DETAILS', 'RELATIONS', 'BEHAVIOR', and 'COMMUNITY' (with 7 items) are visible. A banner encourages joining the community for more insights. At the bottom, the 'Popular threat label' is `trojan.r002c0pdm21`, the 'Threat categories' include 'trojan', and the 'Family labels' include 'r002c0pdm21'.

38
/ 72

Community
Score

38/72 security vendors and no sandboxes flagged this file as malicious

b71777edbf21167c96d20ff803cbcb25d24b94b3652db2f286dcd6efd3d8416a

Lab05-02.exe

peexe checks-network-adapters runtime-modules direct-cpu-clock-access armadillo

Size
40.00 KB

Last Modification Date
5 hours ago

EXE

DETECTION

DETAILS

RELATIONS

BEHAVIOR

COMMUNITY 7

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Popular threat label ⓘ trojan.r002c0pdm21

Threat categories trojan

Family labels r002c0pdm21

Malware_U3_W2_L5

Sempre con CFF Explorer, possiamo analizzare le informazioni riguardanti le librerie utilizzate e le funzioni importate dal malware. In questo caso ne individuiamo 2: **KERNEL32.dll** (da cui il malware importa 44 funzioni) e **WININET.dll** (con 5 funzioni importate).

Malware_U3_W2_L5.exe						
Module Name	Imports	OFTs	TimeStamp	ForwarderChain	Name RVA	FTs (IAT)
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.dll	44	00006518	00000000	00000000	000065EC	00006000
WININET.dll	5	000065CC	00000000	00000000	00006664	000060B4

KERNEL32.dll

La KERNEL32.dll è una delle principali librerie di sistema di Windows e fornisce funzioni fondamentali per la gestione della memoria, dei processi e dei thread, dell'I/O (input/output) e di altre operazioni di sistema basilari. È essenziale per il funzionamento delle applicazioni su Windows.

Funzioni comuni in KERNEL32.dll includono:

- Creazione e gestione dei processi (CreateProcess)
- Gestione della memoria (VirtualAlloc, VirtualFree)
- Gestione dei file (CreateFile, ReadFile, WriteFile)
- Operazioni su thread (CreateThread, ExitThread)

I malware spesso utilizzano KERNEL32.dll per eseguire operazioni fondamentali come creare nuovi processi o thread, manipolare file, allocare memoria, e molto altro. L'uso intensivo di questa libreria è comune in quasi tutti i tipi di software, inclusi i malware.

WININET.dll

La WININET.dll (Windows Internet) è una libreria che fornisce funzioni per l'accesso a Internet e per l'implementazione di protocolli come HTTP e FTP. Questa libreria permette alle applicazioni di interagire con le risorse di rete e di effettuare operazioni come il download e l'upload di file via Internet.

Funzioni comuni in WININET.dll includono:

- Connessione a server HTTP e FTP (InternetOpen, InternetConnect)
- Richieste HTTP (HttpOpenRequest, HttpSendRequest)
- Gestione di sessioni Internet (InternetOpenUrl, InternetCloseHandle)

I malware che necessitano di comunicare con server remoti per scaricare ulteriori payload, esfiltrare dati, o ricevere comandi spesso utilizzano WININET.dll. Questa libreria permette al malware di connettersi a Internet, scaricare file, inviare informazioni, e altro, senza dover implementare direttamente i complessi protocolli di rete.

Malware_U3_W2_L5

Per quanto riguarda le sezioni del file eseguibile PE (Portable Executable) e i relativi attributi, possiamo individuarne 3: **.text**, **.rdata** e **.data**.

- La sezione **.text** contiene il codice eseguibile del programma. È la sezione dove risiede il codice macchina che viene eseguito dalla CPU. Qualsiasi analisi o disassemblaggio del codice malevolo inizierà qui.
- La sezione **.rdata** contiene dati di sola lettura. Questo include spesso le tabelle di importazione e le stringhe costanti che indicano quali funzioni di sistema o di altre librerie vengono utilizzate dal malware.
- La sezione **.data** contiene dati inizializzati utilizzati dal programma come variabili globali e statiche che sono accessibili in lettura e scrittura. Qui troviamo dati modificabili che il malware utilizza durante la sua esecuzione

Malware_U3_W2_L5.exe									
Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations N...	Linenumbers ...	Characteristics
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	00004A78	00001000	00005000	00001000	00000000	00000000	0000	0000	60000020
.rdata	0000095E	00006000	00001000	00006000	00000000	00000000	0000	0000	40000040
.data	00003F08	00007000	00003000	00007000	00000000	00000000	0000	0000	C0000040



Analisi codice Assembly

Cos'è Assembly?

Assembly è un linguaggio di programmazione di basso livello che è strettamente legato all'architettura del computer su cui viene eseguito. Ogni istruzione in Assembly corrisponde direttamente a un'istruzione di macchina specifica, rendendolo molto efficiente in termini di prestazioni.

A differenza dei linguaggi di alto livello (come Python o Java), l'Assembly richiede una comprensione dettagliata dell'hardware del computer, come i registri della CPU, la memoria e le istruzioni macchina. Il codice Assembly è spesso utilizzato per scrivere software che richiedono una gestione precisa delle risorse hardware, come i driver di dispositivo, i sistemi operativi e i programmi in tempo reale. E' essenziale nell'analisi statica dei malware poiché ci consente di vedere esattamente come i malware interagiscono con il sistema operativo e le sue risorse. Questo livello di dettaglio è fondamentale per comprendere completamente il comportamento del malware e per sviluppare contromisure efficaci.

Costrutti noti

- **Prologo della Funzione**: salva il valore del puntatore di base corrente (ebp) e imposta ebp al valore dello stack pointer (esp).
- **Chiamata alla Funzione API di Windows**: chiama la funzione API InternetGetConnectedState di Windows, passando due argomenti con valore 0.
- **Confronto e Salto Condizionato**: Il valore restituito dalla funzione InternetGetConnectedState viene salvato in [ebp+var_4]. Poi viene confrontato con 0 e, se uguale a 0, il controllo viene trasferito a loc_40102B.

```
push    ebp
mov     ebp, esp
push    ecx
push    0          ; dwReserved
push    0          ; lpdwFlags
call    ds:InternetGetConnectedState
mov     [ebp+var_4], eax
cmp     [ebp+var_4], 0
jz      short loc_40102B
```

Costrutti noti

- **Messaggio di successo**: Se la connessione a Internet è presente, viene eseguito questo blocco. Viene inviato un messaggio di successo, eax viene impostato a 1 e viene eseguito un salto a loc_40103A.

```
push    offset aSuccessInterne ; "Success: Internet Connection\n"  
call    sub_40117F  
add     esp, 4  
mov     eax, 1  
jmp     short loc_40103A
```

- **Messaggio di errore**: Se la connessione a Internet non è presente, viene eseguito questo blocco. Viene inviato un messaggio di errore e eax viene azzerato.

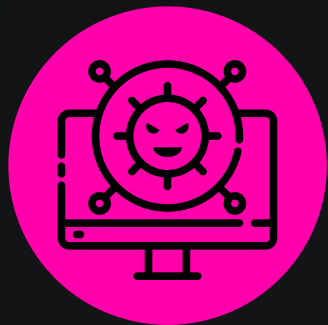
```
loc_40102B:                ; "Error 1.1: No Internet\n"  
push    offset aError1_1NoInte  
call    sub_40117F  
add     esp, 4  
xor     eax, eax
```

Costrutti noti

- **Epilogo della Funzione**: Questo è l'epilogo della funzione che ripristina il valore originale di ebp e ritorna al chiamante.

```
loc_40103A:  
mov     esp, ebp  
pop     ebp  
retn  
sub_401000 endp
```


Funzionalità



Questo codice in Assembly implementa una funzionalità che verifica la presenza di una connessione Internet e fornisce un feedback all'utente in base al risultato. Utilizzando la funzione API `InternetGetConnectedState`, se la connessione è presente, stampa un messaggio di successo e imposta il valore di ritorno a 1. Se invece la connessione non è presente, stampa un messaggio di errore e imposta il valore di ritorno a 0. Il flusso del programma è controllato tramite istruzioni di confronto e salti condizionali. In caso di connessione Internet attiva, il malware potrebbe procedere con ulteriori azioni, mentre in caso contrario, potrebbe terminare o tentare altre tecniche di connessione.

Analisi

push ebp	Salva il valore del registro ebp (Base Pointer) sullo stack.
mov ebp, esp	Copia il valore del registro esp (Stack Pointer) in ebp. Questo stabilisce un nuovo frame dello stack.
push ecx	Salva il valore del registro ecx sullo stack.
push 0 :dwReserved	Carica il valore 0 sullo stack come primo argomento per la chiamata alla funzione InternetGetConnectedState.
push 0 :lpdwFlags	Carica il valore 0 sullo stack come secondo argomento per la chiamata alla funzione InternetGetConnectedState.
call ds:InternetGetConnectedState	Chiama la funzione InternetGetConnectedState utilizzando l'address dal segmento dei dati.
mov [ebp+var_4], eax	Salva il valore di ritorno della funzione InternetGetConnectedState (contenuto in eax) nella variabile locale var_4.
cmp [ebp+var_4], 0	Confronta il valore della variabile var_4 con 0.
jz short loc_40102B	Se il valore di var_4 è 0 (nessuna connessione Internet), salta all'etichetta loc_40102B.
push offset aSuccessInterne	Carica l'offset della stringa "Success: Internet Connection\n" sullo stack.

Analisi

call sub_40117F	Chiama una funzione (sub_40117F) che probabilmente stampa il messaggio sullo schermo.
add esp, 4	Libera lo spazio dello stack riservato per l'argomento della funzione chiamata.
mov eax, 1	Imposta il registro eax a 1 per indicare successo.
jmp short loc_40103A	Salta all'etichetta loc_40103A, bypassando il blocco di errore.
loc_40102B	Etichetta per il blocco di errore.
push offset aError1_1NoInte	Carica l'offset della stringa "Error 1.1: No Internet\n" sullo stack.
call sub_40117F	Chiama una funzione (sub_40117F) che probabilmente stampa il messaggio di errore sullo schermo.
add esp, 4	Libera lo spazio dello stack riservato per l'argomento della funzione chiamata.
xor eax, eax	Imposta il registro eax a 0 per indicare fallimento.
loc_40103A	Etichetta per il punto di fine della funzione

Analisi

mov esp, ebp	Ripristina il valore originale di esp usando ebp.
pop ebp	Ripristina il valore originale di ebp dallo stack.
retn	Ritorna al chiamante della funzione.
sub_401000 endp	Indica la fine della funzione.

Come difendersi dai malware?

- Software Antivirus e Anti-Malware:

Installare e mantenere aggiornati software antivirus e anti-malware su tutti i dispositivi. Questi strumenti possono rilevare e bloccare il malware prima che possa causare danni.

- Aggiornamenti di Sicurezza:

Assicurarsi che tutti i sistemi operativi, i software e le applicazioni siano aggiornati con le ultime patch di sicurezza. Le vulnerabilità non corrette possono essere sfruttate dal malware.

- Firewall:

Utilizzare firewall per monitorare e controllare il traffico di rete. I firewall possono bloccare comunicazioni non autorizzate e impedire al malware di comunicare con server di comando e controllo.

- Formazione degli Utenti:

Educare gli utenti sui rischi del malware e su come evitare comportamenti rischiosi, come cliccare su link sospetti o scaricare allegati da email non verificate.

Come difendersi dai malware?

- Controllo degli Accessi:

Implementare politiche di controllo degli accessi per limitare i privilegi degli utenti. Gli account con privilegi elevati dovrebbero essere utilizzati solo quando necessario e con attenzione.

- Backup Regolari:

Eseguire regolarmente backup dei dati importanti. In caso di attacco malware, i backup possono consentire un rapido ripristino dei dati senza pagare un riscatto o perdere informazioni critiche.

- Analisi del Comportamento:

Utilizzare soluzioni di sicurezza che monitorano il comportamento delle applicazioni e dei processi. Questo può aiutare a identificare attività sospette o anomale che potrebbero indicare la presenza di malware.

- Ambienti Isolati per Test:

Eseguire software sconosciuti o sospetti in ambienti isolati o sandbox per osservare il loro comportamento senza rischiare l'infezione del sistema principale.



Grazie