Progetto Finale
Cyber Security Analist 2024

Fabrizio Meini

Analisi di un malware su OS WINDOWS

1. Analisi Statica

L'analisi statica del malware consiste nell'esaminare il file eseguibile senza eseguirlo, utilizzando strumenti di reverse engineering per determinare le sue caratteristiche e funzionalità. In questa fase, ci concentreremo su vari aspetti, tra cui la struttura del file, le librerie importate e le variabili dichiarate.

a. Parametri della funzione `main()`

Per identificare i parametri passati alla funzione `main()`, abbiamo utilizzato IDA Pro, uno strumento avanzato di disassemblaggio. I parametri passati alla funzione `main()` sono cruciali per comprendere come il malware interagisce con il sistema operativo e gestisce le sue operazioni. Dalla nostra analisi, abbiamo scoperto che la funzione `main()` riceve tre parametri principali:

- **`argc`:** Questo parametro rappresenta il numero di argomenti passati al programma tramite la riga di comando. È di tipo `int` e fornisce un conteggio degli argomenti.
- **`argv`:** Questo è un array di stringhe (puntatori a `char`) che contiene gli argomenti stessi passati alla riga di comando. Ogni elemento dell'array rappresenta un argomento.
- **`envp`:** Questo parametro è un array di stringhe che rappresentano le variabili d'ambiente dell'utente. Queste variabili possono influenzare il comportamento del programma e possono essere utilizzate per determinare le condizioni di esecuzione.

b. Variabili dichiarate all'interno della funzione `main()`

All'interno della funzione `main()`, sono dichiarate cinque variabili locali. Queste variabili sono utilizzate per gestire dati temporanei e per facilitare le operazioni durante l'esecuzione del malware. Le variabili dichiarate sono:

- **`hmodule`:** Utilizzata per memorizzare un handle di modulo, probabilmente relativo al caricamento di una DLL o di un altro modulo.
- **`data`:** Potrebbe essere utilizzata per gestire dati temporanei o per immagazzinare risultati intermedi.

- **`var_117`:** Una variabile il cui scopo specifico non è immediatamente chiaro senza ulteriori dettagli, ma è dichiarata e utilizzata all'interno della funzione.
- **`var_8`:** Simile a `var_117`, è un'altra variabile locale che può essere usata per operazioni temporanee.
- **`var_4`:** Questa variabile potrebbe essere usata per gestire piccoli dati o risultati di operazioni.

Queste variabili vengono allocate nello stack e gestite tramite istruzioni di manipolazione dello stack, come `push` e `pop`.

c. Sezioni del file eseguibile

Il file eseguibile del malware è strutturato in diverse sezioni, ognuna delle quali ha uno scopo specifico. Le sezioni principali identificate sono:

- **`.text`:** Questa sezione contiene il codice eseguibile del malware. È la parte del file dove risiedono le istruzioni che la CPU eseguirà quando il malware verrà avviato. La sezione `.text` è essenziale perché include la logica del malware e le sue operazioni principali.
- **`.rdata`:** Questa sezione include dati di sola lettura, come le informazioni sulle librerie importate e le stringhe di testo. Le informazioni contenute in `.rdata` possono includere nomi di funzioni e variabili che il malware utilizza durante la sua esecuzione. È utile per identificare le dipendenze del malware e le risorse di cui ha bisogno.
- **`.data`:** Contiene le variabili globali e i dati che devono essere accessibili da qualsiasi parte del programma. Questa sezione viene utilizzata per memorizzare informazioni che devono persistere durante l'esecuzione del malware.
- **`.rsrc`:** Include le risorse non di codice, come icone, immagini, e altri dati che non sono direttamente eseguibili ma necessari per l'interfaccia utente del malware o altre funzioni.

d. Librerie importate e loro funzionalità

Il malware importa due librerie fondamentali, ognuna delle quali offre funzionalità cruciali che possono essere sfruttate per ottenere determinati obiettivi malevoli:

- **`kernel32.dll`:** Questa libreria è una delle più importanti del sistema operativo Windows e fornisce funzioni per la gestione della memoria, dei processi e dei thread. Le funzioni di `kernel32.dll` permettono al malware di manipolare processi e ottenere persistenza. Per esempio, la funzione `CreateProcess` potrebbe essere utilizzata per avviare ulteriori istanze del malware, mentre `VirtualAlloc` potrebbe essere impiegata per allocare memoria per il codice malevolo.

- **`advapi32.dll`:** Fornisce funzioni relative alla sicurezza, alla gestione degli account e alle operazioni sul registro di sistema. Le funzioni di `advapi32.dll` potrebbero essere utilizzate dal malware per alterare le chiavi di registro, modificare le autorizzazioni e accedere a token di sicurezza. Funzioni come `RegSetValueEx` e `RegCreateKeyEx` sono particolarmente rilevanti per il malware che cerca di ottenere persistenza attraverso modifiche nel registro di sistema.

2. Analisi Dinamica

L'analisi dinamica implica l'esecuzione del malware in un ambiente controllato per osservare direttamente il suo comportamento. Questa fase fornisce informazioni sulle modifiche al file system, alle chiavi di registro e alle connessioni di rete.

a. Osservazioni nella cartella del malware

Dopo aver eseguito il malware, abbiamo osservato modifiche nella cartella in cui si trovava l'eseguibile. I cambiamenti principali notati sono:

- **Creazione di nuovi file e cartelle:** Il malware potrebbe creare nuovi file o cartelle all'interno della sua directory per memorizzare ulteriori payloads o per configurare il sistema in modo che il malware venga eseguito automaticamente.
- **Ridenominazione di file:** Il malware potrebbe rinominare file esistenti o l'eseguibile stesso per mascherare la sua presenza e rendere più difficile la sua identificazione. La ridenominazione dei file può essere una tecnica utilizzata per evitare la rilevazione da parte di strumenti di sicurezza.

b. Risultati di Process Monitor

Process Monitor (ProcMon) è stato utilizzato per monitorare le attività del malware durante l'esecuzione. I risultati dell'analisi mostrano:

- **Modifiche al registro:** Il malware ha apportato modifiche alle chiavi di registro per ottenere persistenza. Questo potrebbe includere la creazione di nuove chiavi o la modifica di chiavi esistenti per garantire che il malware venga eseguito automaticamente ad ogni avvio del sistema.
- **Attività sui file:** Il malware potrebbe aver creato, modificato o eliminato file per nascondere la propria presenza o per preparare il sistema per ulteriori attacchi. Le modifiche ai file possono includere la creazione di file di log, la scrittura di dati temporanei o la cancellazione di file per evitare la scoperta.

c. Filtraggio dell’attività del registro

Filtrando le attività di registro in Process Monitor, è possibile ottenere informazioni specifiche su:

- **Chiave di registro creata:** Esempio di chiave: `HKCU\Software\Microsoft\Windows\CurrentVersion\Run\Malware`
- **Valore associato alla chiave di registro:** Il valore potrebbe essere il percorso dell'eseguibile del malware, come `C:\ProgramData\Malware.exe`. Questa chiave nel registro è utilizzata per assicurare che il malware venga eseguito ogni volta che il sistema si avvia.

d. Modifiche al file system

Durante l'analisi del file system, è emerso che:

- **Chiamata di sistema che ha modificato il contenuto della cartella del malware:** Chiamate di sistema come `MoveFileEx` o `CreateFile` sono state utilizzate per rinominare, creare o modificare file. Queste operazioni sono parte della strategia del malware per offuscare la sua presenza e rendere più difficile il rilevamento e la rimozione.

3. Analisi Dettagliata della Funzione Specifica

a. Scopo della funzione alla locazione di memoria 00401021

Alla locazione di memoria 00401021, il malware chiama la funzione `RegCreateKeyExA`. Questa funzione è utilizzata per creare una nuova chiave di registro o aprire una chiave esistente. Il suo scopo principale è ottenere persistenza, creando o modificando chiavi nel registro di sistema per garantire che il malware venga eseguito automaticamente all'avvio del sistema.

b. Passaggio dei parametri alla funzione

I parametri sono passati alla funzione `RegCreateKeyExA` tramite lo stack. Alla locazione 00401017, l'indirizzo della sottochiave di registro viene spinto nello stack. I parametri vengono letti dalla funzione in ordine inverso, quindi il valore più recente è il primo ad essere utilizzato.

c. Oggetto del parametro alla locazione 00401017

Alla locazione 00401017, il parametro rappresenta il nome della sottochiave di registro che deve essere creata. Questo parametro è un puntatore a una stringa che specifica il nome della chiave, e può influenzare il comportamento del malware modificando le impostazioni di esecuzione.

d. Significato delle istruzioni tra 00401027 e 00401029

Le istruzioni tra 00401027 e 00401029 sono:

- **`test` (00401027):** Questa istruzione verifica se un valore è zero. Non modifica il valore, ma imposta il flag ZF (zero flag) a 1 se il risultato dell'operazione è zero.
- **JZ (00401029):** Salta a un'altra locazione di memoria se il flag ZF è impostato, indicando che il risultato della precedente istruzione `test` era zero.

Traduzione del codice Assembly in C:

```
if (EAX == 0) {
    // Esegui azione specifica
}
```

e. Valore del parametro `ValueName` alla locazione 00401047

Alla locazione 00401047, il parametro `ValueName` è la stringa "ginadll". Questo valore rappresenta il nome della voce di registro che il malware sta cercando di impostare. L'uso di nomi come "ginadll" può essere un tentativo di mascherare l'attività malevola, rendendo più difficile per gli utenti e per i software di sicurezza identificare la vera natura del malware.

Conclusioni Finali

L'analisi combinata statica e dinamica del malware rivela diverse tecniche utilizzate per garantire la persistenza e per nascondere la sua attività. Il malware sfrutta funzioni del registro di sistema per mantenere la propria esecuzione attraverso modifiche alle chiavi di registro e valori. Inoltre, il comportamento osservato, come la creazione di file e la modifica del file system, suggerisce che il malware adotta strategie per mascherare la sua presenza e per complicare il rilevamento.

In sintesi, il malware utilizza tecniche di persistenza basate sul registro e modifica il file system per garantire la propria esecuzione e nascondere le sue tracce. Questa comprensione è essenziale per sviluppare contromisure efficaci e proteggere i sistemi compromessi da ulteriori attacchi.

Questa conclusione si basa su diversi elementi dell'analisi:

1. ***Librerie Importate:***
    - ***`kernel32.dll`*** *e* ***`advapi32.dll`*** *sono librerie fondamentali di Windows.* `kernel32.dll` *gestisce operazioni di base come la gestione dei processi e della memoria, mentre* `advapi32.dll` *è coinvolta nella gestione della sicurezza e delle operazioni sul registro di sistema. L'importazione di queste librerie suggerisce che il malware è progettato per essere eseguito su un ambiente Windows.*
2. ***Struttura e Funzioni dell'API di Windows:***
    - *La funzione* `RegCreateKeyExA` *e* `RegSetValueEx` *sono API di Windows utilizzate per manipolare il registro di sistema. Il malware fa uso di queste funzioni per creare e modificare chiavi di registro, un'indicazione chiara che l'ambiente target è Windows.*
3. ***Processi e File System:***
    - *L'uso di percorsi di file e chiavi di registro tipici di Windows, come* `HKCU\Software\Microsoft\Windows\CurrentVersion\Run\`*, dimostra che il malware è progettato per interagire con il registro di Windows e i percorsi di file system specifici di Windows.*
4. ***Architettura del Malware:***
    - *Il malware è stato compilato per architetture di processori compatibili con Intel 386 e successivi, che sono architetture comuni nei sistemi Windows.*
5. ***Analisi Statica e Dinamica:***
    - *Strumenti e tecniche di analisi come CFF Explorer, EXEinfoPE, e IDA Pro sono utilizzati specificamente per analizzare file eseguibili Windows. Inoltre, l'uso di strumenti come Process Monitor e Cuckoo Sandbox per monitorare le attività del malware è comune per l'analisi di malware su Windows.*

In fine, tutte le evidenze suggeriscono che il malware è progettato per operare su sistemi operativi Windows, sfruttando le sue API e caratteristiche specifiche per eseguire e mantenere la propria persistenza.