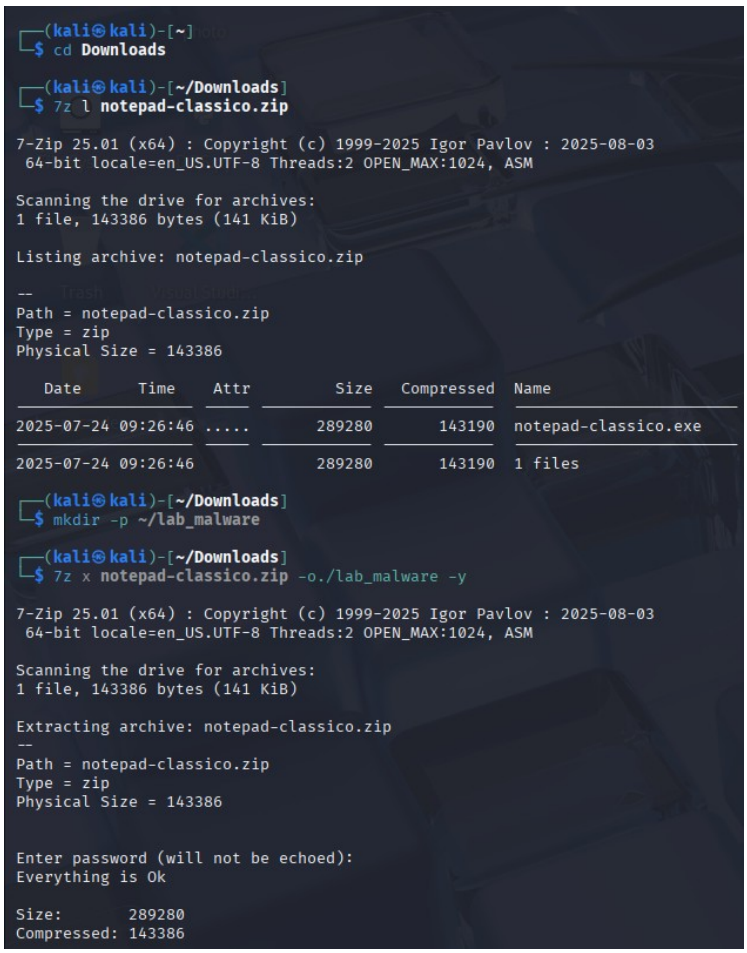
Pratica S9/L2

1. Introduzione

L’analisi condotta ha avuto come obiettivo lo studio del file eseguibile *notepad-classico.exe*, fornito in un archivio compresso nell’ambito dell’esercitazione. Lo scopo principale dell’attività è stato quello di applicare un’analisi statica ad un potenziale malware, al fine di identificare le librerie importate e le sezioni che compongono il file, in un ambiente controllato. Questa fase preliminare consente di raccogliere informazioni fondamentali sulla natura del programma senza doverlo eseguire, riducendo i rischi associati ad un comportamento malevolo. Inoltre, l’impiego di un supporto AI ha permesso di produrre descrizioni dettagliate e contestualizzate delle API e delle strutture individuate, rendendo l’analisi più chiara e comprensibile.

2. Preparazione ed estrazione dei dati

Il file *notepad-classico.zip*, contenente l’eseguibile da analizzare, è stato scaricato in un ambiente controllato (macchina virtuale Kali Linux) al fine di ridurre al minimo i rischi legati all’apertura di potenziali malware. L’analisi era inizialmente prevista in ambiente Windows tramite strumenti dedicati come CFF Explorer. Tuttavia, a causa di difficoltà tecniche riscontrate con la macchina virtuale Windows, non è stato possibile seguire direttamente tale approccio. Per garantire comunque la prosecuzione dell’attività, l’analisi è stata condotta in Kali Linux, utilizzando strumenti da riga di comando per estrarre le informazioni necessarie dal file. Dopo aver scaricato l’archivio compresso, sono state eseguite le operazioni preliminari da terminale per predisporre l’ambiente di lavoro e procedere all’analisi statica.



- Spostamento nella cartella di download: *cd Downloads*
- Verifica del contenuto dell'archivio elencando i file presenti nello zip senza estrarli: *7z l notepad-classico.zip*
- Creazione di una cartella dedicata all'analisi: *mkdir -p ~/lab_malware*
- Estrazione del file nella cartella dedicata: *7z x notepad-classico.zip -o./lab_malaware -y*

3. Estrazione delle informazioni dal file eseguibile

Una volta ottenuto l'eseguibile notepad-classico.exe nella cartella di laboratorio, si è proceduto con l'analisi statica del file utilizzando strumenti da terminale. L'obiettivo era quello di estrarre due insiemi di informazioni fondamentali:

- Le librerie importate (DLL e relative funzioni).
- Le sezioni interne del formato PE (Portable Executable).

Per ottenere le informazioni sono stati utilizzati i seguenti comandi:

1. Estrazione delle librerie importate tramite *rabin2*

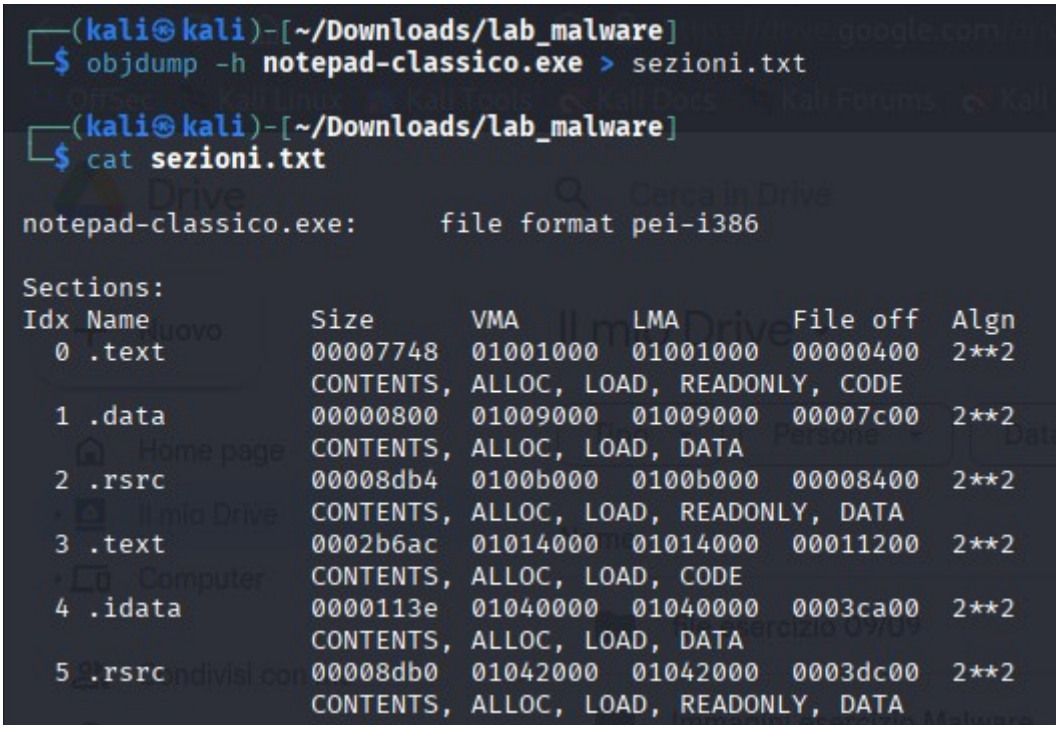
```
┌──(kali㉿kali)-[~/Downloads]
└─$ cd lab_malware

┌──(kali㉿kali)-[~/Downloads/lab_malware]
└─$ rabin2 -i notepad-classico.exe > librerie.txt

┌──(kali㉿kali)-[~/Downloads/lab_malware]
└─$ cat librerie.txt
[Imports]
nth vaddr      bind type lib          name
-----------------------------------------
1   0x010012c4 NONE FUNC comdlg32.dll PageSetupDlgW
2   0x010012c8 NONE FUNC comdlg32.dll FindTextW
3   0x010012cc NONE FUNC comdlg32.dll PrintDlgExW
4   0x010012d0 NONE FUNC comdlg32.dll ChooseFontW
5   0x010012d4 NONE FUNC comdlg32.dll GetFileTitleW
6   0x010012d8 NONE FUNC comdlg32.dll GetOpenFileNameW
7   0x010012dc NONE FUNC comdlg32.dll ReplaceTextW
8   0x010012e0 NONE FUNC comdlg32.dll CommDlgExtendedError
9   0x010012e4 NONE FUNC comdlg32.dll GetSaveFileNameW
1   0x01001174 NONE FUNC SHELL32.dll  DragFinish
2   0x01001178 NONE FUNC SHELL32.dll  DragQueryFileW
3   0x0100117c NONE FUNC SHELL32.dll  DragAcceptFiles
4   0x01001180 NONE FUNC SHELL32.dll  ShellAboutW
1   0x010012b4 NONE FUNC WINSPOOL.DRV GetPrinterDriverW
2   0x010012b8 NONE FUNC WINSPOOL.DRV ClosePrinter
3   0x010012bc NONE FUNC WINSPOOL.DRV OpenPrinterW
1   0x01001020 NONE FUNC COMCTL32.dll CreateStatusWindowW
1   0x010012ec NONE FUNC msvcrt.dll   _XcptFilter
2   0x010012f0 NONE FUNC msvcrt.dll   _exit
3   0x010012f4 NONE FUNC msvcrt.dll   _c_exit
4   0x010012f8 NONE FUNC msvcrt.dll   time
5   0x010012fc NONE FUNC msvcrt.dll   localtime
6   0x01001300 NONE FUNC msvcrt.dll   _cexit
7   0x01001304 NONE FUNC msvcrt.dll   iswctype
8   0x01001308 NONE FUNC msvcrt.dll   _except_handler3
9   0x0100130c NONE FUNC msvcrt.dll   _wtol
10  0x01001310 NONE FUNC msvcrt.dll   wcsncmp
11  0x01001314 NONE FUNC msvcrt.dll   _snwprintf
12  0x01001318 NONE FUNC msvcrt.dll   exit
13  0x0100131c NONE FUNC msvcrt.dll   _acmdln
14  0x01001320 NONE FUNC msvcrt.dll   __getmainargs
15  0x01001324 NONE FUNC msvcrt.dll   _initterm
16  0x01001328 NONE FUNC msvcrt.dll   __setusermatherr
17  0x0100132c NONE FUNC msvcrt.dll   _adjust_fdiv
18  0x01001330 NONE FUNC msvcrt.dll   __p__commode
19  0x01001334 NONE FUNC msvcrt.dll   __p__fmode
20  0x01001338 NONE FUNC msvcrt.dll   __set_app_type
21  0x0100133c NONE FUNC msvcrt.dll   _controlfp
22  0x01001340 NONE FUNC msvcrt.dll   wcsncpy
1   0x01001000 NONE FUNC ADVAPI32.dll RegQueryValueExW
```

Questo comando analizza la Import Address Table (IAT) dell'eseguibile e salva l'elenco delle DLL e delle API richiamate nel file *librerie.txt*.

2. Estrazione delle sezioni PE tramite *objdump*



Questo comando elenca le sezioni del file, indicando indirizzi e dimensioni, salvando l'output in *sezioni.txt*.

4. Analisi assistita con ChatGPT

I dati contenuti nei file di testo generati nella fase precedente, *librerie.txt* e *sezioni.txt*, sono stati resi più leggibili e significativi con un'analisi svolta con ChatGPT richiedendogli di farmi notare eventuali librerie o sezioni sospette.

Analisi delle librerie importate

Il file importa librerie standard di Windows, tipiche di un'applicazione grafica con funzionalità di editing e stampa.

comdlg32.dll → gestione dei dialoghi comuni (Apri, Salva, Trova, Sostituisci, Stampa, Selezione font). Importazioni come GetOpenFileNameW, PrintDlgExW, ChooseFontW sono fortemente indicative di un editor di testo.

shell32.dll → supporto a drag & drop e dialoghi informativi (DragAcceptFiles, ShellAboutW). Coerente con un programma che permette di aprire file trascinati nella finestra.

winspool.drv → API di stampa (OpenPrinterW, GetPrinterDriverW). Indica la capacità di stampare direttamente su dispositivi locali.

comctl32.dll → controlli grafici comuni (CreateStatusWindowW). Serve per aggiungere una barra di stato nella GUI.

msvcrt.dll → runtime C: gestione di stringhe (wcsncmp), tempo (time, localtime), eccezioni e bootstrap (_XcptFilter, _initterm). Nulla di sospetto: è la libreria standard per eseguibili compilati in C/C++.

advapi32.dll → accesso al Registro di sistema (RegOpenKeyExA, RegQueryValueExW, RegSetValueExW). In un'app lecita serve a memorizzare preferenze e configurazioni; in un malware può essere usata per persistenza.

kernel32.dll → funzioni di basso livello: gestione file (CreateFileW, ReadFile, WriteFile, DeleteFileW), memoria (MapViewOfFile), processi (GetCurrentProcess, TerminateProcess), localizzazione (GetLocaleInfoW), caricamento dinamico di librerie (LoadLibraryA, GetProcAddress). Questo è il cuore delle API Windows, molto usato da qualsiasi programma. Tuttavia, le funzioni LoadLibraryA e GetProcAddress possono anche essere impiegate da malware per caricare codice offuscato o API sensibili in runtime.

gdi32.dll → API grafiche e di stampa (TextOutW, CreateFontIndirectW, StartDocW, EndPage). Tipico di un editor con funzioni di rendering del testo.

user32.dll → GUI e interazione utente (CreateWindowExW, DispatchMessageW, MessageBoxW, OpenClipboard). È la base di ogni applicazione grafica Win32. Da notare SetWinEventHook e UnhookWinEvent: API legittime, ma che possono anche essere usate per monitorare eventi della GUI, funzionalità sfruttata talvolta da keylogger o spyware.

Sintesi librerie

Nel complesso, l'insieme delle librerie è coerente con un editor classico (tipo Notepad avanzato). Non compaiono API marcatamente sospette (come WSAStartup per il networking o CreateRemoteThread per injection). Tuttavia, l'uso di ADVAPI32 (registro) e gli hook in USER32 può destare sospetto se associati ad altri indizi.

Analisi delle sezioni del file eseguibile

Il file notepad-classico.exe è un Portable Executable (PE) a 32 bit che si compone di 6 sezioni principali.

.text (prima istanza)
Indirizzo virtuale: 0x01001000
Dimensione: circa 30 KB
Contiene il codice eseguibile, in particolare le routine di avvio e inizializzazione. La dimensione ridotta fa pensare al bootstrap e all'entry point del programma.

.data
Indirizzo virtuale: 0x01009000
Dimensione: 2 KB
Contiene variabili globali inizializzate e strutture dati scrivibili a runtime. La dimensione contenuta è coerente con un'applicazione leggera.

.rsrc (prima istanza)
Indirizzo virtuale: 0x0100B000
Dimensione: circa 36,9 KB
Contiene risorse grafiche dell'interfaccia utente, come icone, menu, dialoghi e stringhe. La dimensione è tipica di un editor testuale con molte finestre di dialogo.

.text (seconda istanza)
Indirizzo virtuale: 0x01014000
Dimensione: circa 178 KB
Contiene ulteriore codice eseguibile. La presenza di una seconda sezione .text è inusuale. Potrebbe essere frutto di scelte di compilazione o suddivisione del codice in moduli separati. In alternativa, in un contesto sospetto, la duplicazione potrebbe indicare la volontà di nascondere porzioni di codice o tecniche di offuscamento.

.idata
Indirizzo virtuale: 0x01040000
Dimensione: circa 4,4 KB
Contiene la Import Address Table (IAT), ovvero la tabella delle librerie e delle funzioni importate. La grandezza è coerente con il numero elevato di API individuate nell'analisi delle librerie.

.rsrc (seconda istanza)
Indirizzo virtuale: 0x01042000
Dimensione: circa 36,9 KB
Contiene ulteriori risorse. Anche in questo caso la presenza di una seconda sezione .rsrc è insolita ma non illegittima. Può derivare dall'uso di strumenti di compilazione diversi, dalla gestione delle localizzazioni o da altre scelte di progetto. Tuttavia, in casi malevoli, una sezione risorse duplicata può essere sfruttata per nascondere payload o dati cifrati.

Sintesi delle sezioni

La struttura complessiva è compatibile con quella di un'applicazione Windows a interfaccia grafica, che richiede codice, dati, import e risorse. Le anomalie principali sono la presenza di due sezioni .text e due sezioni .rsrc. In scenari legittimi ciò può essere spiegato da particolarità del compilatore o della build, ma in un'ottica di analisi malware queste duplicazioni possono essere interpretate come indicatori di offuscamento o di codice nascosto.

5. Considerazioni

Dall'analisi effettuata, il file *notepad-classico.exe* presenta una serie di caratteristiche tipiche di un normale editor di testo con interfaccia grafica: le librerie importate riguardano la gestione dei file, della GUI, delle finestre di dialogo standard, delle operazioni sul registro e delle funzioni di stampa. Si tratta quindi di un insieme coerente con un'applicazione legittima di tipo "Notepad avanzato". L'analisi delle sezioni del PE conferma la presenza delle componenti canoniche di un eseguibile Windows (codice, dati, import, risorse). L'unico elemento anomalo riscontrato è la duplicazione delle sezioni .text e .rsrc. In contesti leciti, questo fenomeno può derivare da specifiche scelte del compilatore o da moduli gestiti in maniera distinta, ad esempio per localizzazioni o per separare componenti funzionali. In scenari malevoli, invece, tali duplicazioni possono essere utilizzate per mascherare codice nascosto, dati cifrati o tecniche di offuscamento. In conclusione, le evidenze raccolte non sono sufficienti a dichiarare il file come sicuramente malevolo sulla sola base delle librerie e della struttura delle sezioni. Tuttavia, la presenza delle sezioni duplicate giustifica un approfondimento ulteriore, che dovrebbe comprendere analisi dinamiche in ambiente isolato e verifiche sul comportamento in esecuzione.