

Indice

Introduzione al progetto	2
Cos'è un Malware	3
Analisi statica e dinamica.....	3
Linguaggio assembly	3
Cosa fa un Disassembler?	4
Cosa è un dropper	4
Setup e tools	5
CFF Explorer	7
IDA Pro Free	8
OlllyDBG.....	8
Process Monitor.....	9
Task Giorno 1.....	10
Analisi statica con IDA Pro	10
Parametri nella funzione main()	13
Variabili nella funzione main()	14
Analisi statica con CFF Explorer	15
Sezioni	16
Quali librerie importa il malware?	17
Deduzioni sulle funzionalità implementate dal malware	20
Conclusioni.....	20
Task Giorno 2.....	21
Qual è lo scopo della funzione RegCreateKeyExA ?	22
Come vengono passati i parametri alla locazione indicata?	23
Qual è l'oggetto rappresentato alla locazione indicata?	24
Qual è il significato delle istruzioni comprese tra gli indirizzi 00401027 e 00401029	25
Con riferimento all'ultimo quesito, tradurre il codice Assembly nel corrispondente costruito	25
Valutazione della chiamata alla locazione 00401047, qual è il valore del parametro "ValueName"?.....	26
Task Giorno 3.....	26
Valore del parametro "ResourceName" passato alla funzione FindResourceA	26
Funzionalità implementate dal Malware	28

È possibile identificare questa funzionalità utilizzando l'analisi statica basica? Nel caso elencare le evidenze a supporto.....	31
Conclusioni:.....	31
Task Giorno 4.....	32
Analisi dinamica con Process Monitor	32
Cosa avviene all'interno della cartella dove è situato il Malware?	35
Cosa avviene quando il malware viene avviato?	35
Regedit	37
Conclusioni	38
Task Giorno 5.....	39
Sostituzione del File .dll	39
Intercezione delle Credenziali:.....	39
Conseguenze Aggiuntive:.....	40
Misure di Mitigazione e Difesa:	40
Monitoraggio Continuo:.....	42
Conclusioni e Grafico Malware:.....	43

Introduzione al progetto

Benvenuti nella Build Week 3 dedicata all'analisi dei malware. In questo progetto, un gruppo di studenti si impegnerà nell'affrontare un task al giorno per cinque giorni consecutivi, esplorando il complesso mondo dei malware. Ogni giorno della settimana sarà dedicato a una specifica task, progettata per sviluppare le competenze degli studenti nell'individuare, analizzare e mitigare le minacce malware. Queste attività includeranno l'identificazione di nuovi tipi di malware, l'analisi del loro comportamento, l'estrazione di indizi digitali utili per comprendere le loro origini e la loro funzionalità, nonché lo sviluppo di strategie di difesa e di mitigazione.

Attraverso questo progetto, gli studenti avranno l'opportunità di applicare conoscenze teoriche acquisite in classe a situazioni reali, affrontando sfide concrete e affinando le proprie abilità pratiche nell'ambito della sicurezza informatica. Inoltre, collaborando e condividendo le loro esperienze, potranno imparare gli uni dagli altri e ampliare le proprie prospettive.

Prima di cominciare il report con la task numero 1 sarebbe più opportuno iniziare con un po' di teoria sui malware, sui concetti chiave che vedremo durante questa BW3 e i tools che utilizzeremo:

Cos'è un Malware

Un malware è un termine che deriva dalla contrazione delle parole "malicious software" (software maligno). Si tratta di un tipo di software progettato specificamente per danneggiare, interrompere, rubare o altrimenti compromettere i dati, i dispositivi o le reti informatiche senza il consenso dell'utente.

I malware possono assumere molte forme e svolgere diverse funzioni dannose. Alcuni esempi comuni includono virus informatici che si replicano e danneggiano file o interi sistemi, trojan che si fingono innocui ma poi aprono una "porta posteriore" sul dispositivo per consentire l'accesso ai criminali informatici, ransomware che crittografano i file degli utenti e richiedono un riscatto per decifrarli, e spyware che furtivamente raccolgono informazioni personali degli utenti.

In breve, il malware è come un parassita digitale che si insinua nei sistemi informatici con l'obiettivo di causare danni o sfruttare le risorse senza l'autorizzazione dell'utente.

Analisi statica e dinamica

L'analisi statica e dinamica sono due approcci distinti utilizzati nell'analisi dei malware e del software in generale.

L'**analisi statica** coinvolge l'esame del codice sorgente o del file eseguibile senza eseguirlo effettivamente. Durante l'analisi statica, vengono esaminate le caratteristiche del file, come le stringhe, le sezioni, le chiamate di funzione e i metadati, per identificare comportamenti sospetti o pericolosi. Questo tipo di analisi è utile per rilevare le minacce senza il rischio di attivare il malware stesso, ma può essere limitato nel rilevare tecniche di evasione dinamiche o comportamenti che si manifestano solo durante l'esecuzione.

L'**analisi dinamica**, d'altra parte, coinvolge l'esecuzione del malware in un ambiente controllato, come una macchina virtuale, per osservare il suo comportamento in tempo reale. Durante l'esecuzione, vengono monitorate le attività del malware, come le chiamate di sistema, le comunicazioni di rete e le modifiche del sistema, per identificare le azioni dannose o anomale. Questo approccio offre una visione più completa del comportamento del malware, ma può essere soggetto a tecniche di evasione che rilevano l'ambiente di analisi e alterano il comportamento del malware in risposta.

Linguaggio assembly

L'assembly è un linguaggio di programmazione a basso livello che è strettamente correlato all'architettura hardware di un computer. È una rappresentazione mnemonica dei codici macchina, il linguaggio binario che il processore esegue direttamente. In sostanza, quando scrivi un programma in un linguaggio di alto livello come C, C++, Java o Python, il codice sorgente viene tradotto in codice macchina dal compilatore. L'assembly è un passo più vicino a questo codice macchina e quindi è più vicino alla comunicazione diretta con l'hardware del computer. L'assembly è molto potente ma richiede una conoscenza dettagliata dell'architettura hardware del computer su cui si sta lavorando. Ogni istruzione assembly corrisponde direttamente a un'operazione eseguita dal processore, come l'aggiunta di due numeri o il trasferimento di dati tra registri. L'assembly viene anche utilizzato per scrivere codice macchina leggibile dagli esseri umani e comprensibile dai processori. Per l'analisi dei malware, l'assembly è prezioso perché consente agli analisti di comprendere il comportamento interno dei malware stessi. Studiare il codice assembly di un malware può rivelare le sue funzionalità, le tecniche di evasione, le vulnerabilità sfruttate e altro ancora. Inoltre, l'analisi dell'assembly può aiutare gli esperti di sicurezza a sviluppare contromisure efficaci e a migliorare le capacità di rilevamento e rimozione delle minacce informatiche.

Cosa fa un Disassembler?

Nell'ambito dell'analisi del malware, un disassembler è uno strumento essenziale per comprendere il comportamento interno del malware e identificare le sue funzionalità, il suo funzionamento e le possibili azioni dannose. Ecco cosa fa un disassembler specificamente in malware analysis:

Decompilazione del codice binario: Il malware di solito è distribuito come file binario eseguibile, il che significa che il suo codice sorgente originale non è disponibile. Un disassembler può essere utilizzato per decompilare il codice binario del malware in codice assembly leggibile, permettendo agli analisti di comprendere l'operazione dettagliata del malware.

Analisi statica del codice assembly: Una volta ottenuto il codice assembly del malware, gli analisti possono esaminare staticamente il codice per identificare le istruzioni, le funzioni, le chiamate di sistema e altre caratteristiche del malware. Questa analisi fornisce informazioni cruciali sul comportamento del malware e sulle sue capacità.

Identificazione delle funzionalità del malware: Gli analisti possono utilizzare il disassembler per identificare le funzionalità specifiche del malware, come la capacità di rubare informazioni sensibili, interrompere il funzionamento del sistema, eseguire attacchi di tipo ransomware o svolgere altre azioni dannose.

Analisi del flusso di esecuzione: Studiando il codice assembly del malware, gli analisti possono analizzare il flusso di esecuzione del programma, osservare come il malware interagisce con il sistema operativo e altri software presenti nel sistema, e identificare eventuali comportamenti anomali o sospetti.

Reverse engineering: Il codice assembly generato dal disassembler può essere utilizzato come base per la reverse-engineering del malware, consentendo agli analisti di comprendere il funzionamento interno del malware senza avere accesso al suo codice sorgente originale. Questo processo è fondamentale per sviluppare contromisure efficaci contro il malware e proteggere i sistemi da futuri attacchi.

In breve, un disassembler è uno strumento essenziale nell'analisi del malware, poiché consente agli analisti di comprendere il comportamento e le capacità del malware, identificare le sue funzionalità dannose e sviluppare strategie di difesa per proteggere i sistemi e i dati dagli attacchi informatici.

Cosa è un dropper

Un dropper è un tipo di malware progettato per installare e lanciare altri componenti dannosi sul sistema target. Il termine "dropper" deriva dal fatto che il suo obiettivo principale è quello di "rilasciare" o "depositare" il payload (una parte del software dannoso che esegue le azioni desiderate dal creatore del malware) malware sul sistema infetto. Il dropper in sé può non essere dannoso o malevolo, ma svolge un ruolo cruciale nel processo di infezione e di compromissione del sistema.

Ma nella fattispecie, come funziona un dropper?

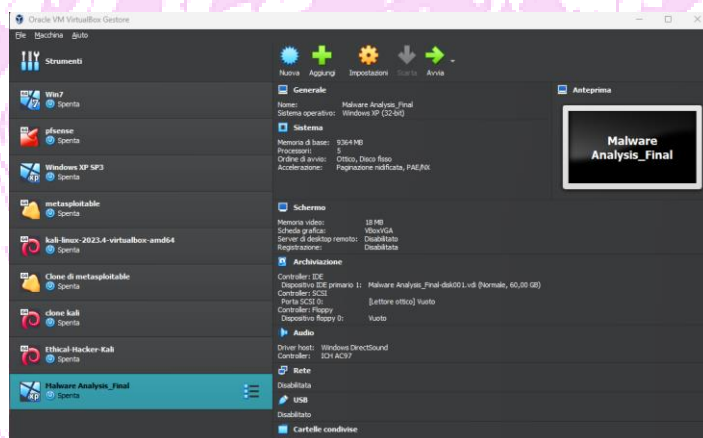
1. **Consegna del payload:** Il dropper viene spesso distribuito attraverso email di phishing, siti web compromessi, annunci dannosi o altri vettori di attacco. Una volta eseguito sul sistema target, il dropper avvia il processo di installazione del payload malware.

2. **Installazione del payload:** Una volta eseguito, il dropper esegue una serie di operazioni per installare il payload malware sul sistema. Questo potrebbe includere la decompressione di file, la modifica delle impostazioni di sistema, l'iniezione di codice maligno in processi legittimi o altre tecniche per eludere le difese del sistema.
3. **Esecuzione del payload:** Dopo aver installato il payload, il dropper avvia la sua esecuzione sul sistema target. Il payload può essere un malware più sofisticato, come un trojan, un ransomware o una backdoor, che svolge le vere e proprie azioni dannose sul sistema, come il furto di dati, il blocco dei file o il controllo remoto del sistema.
4. **Camouflage e evasione:** Alcuni dropper sono progettati per nascondere la presenza del malware sul sistema o per eludere le tecniche di rilevamento e mitigazione delle minacce. Questi dropper possono utilizzare tecniche di cifratura, polimorfismo, o tecniche di evasione comportamentale per sfuggire all'individuazione da parte degli strumenti di sicurezza.

Setup e tools

Di seguito un buon setup di base per creare un ambiente sandbox per l'analisi di malware ottimale:

Macchina virtuale (VM): Utilizza un software di virtualizzazione come VirtualBox, VMware o Hyper-V per creare una macchina virtuale isolata:



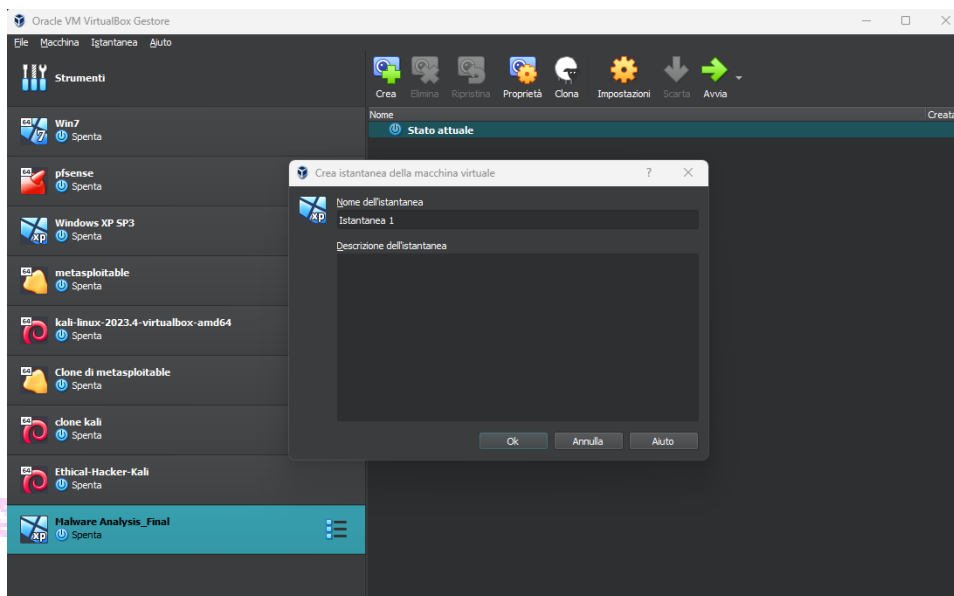
Noi utilizzeremo una macchina virtuale basata su Windows XP 32 bit che ci è stata precedentemente fornita da EPICODE:



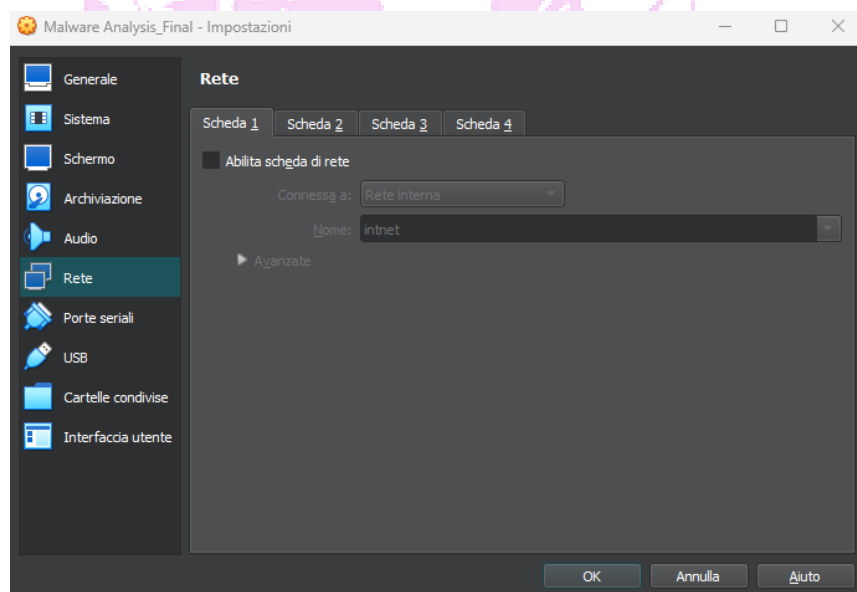
Per svolgere l'intera BW3 in sicurezza bisogna provvedere ad alcuni accorgimenti:

1. **Snapshot:** Prima di eseguire il malware nella macchina virtuale, creare uno snapshot, ovvero un salvataggio del primo avvio della nostra macchina, questo perché lavorando con vari malware potrebbe succedere che la Windows potrebbe smettere di funzionare regolarmente.

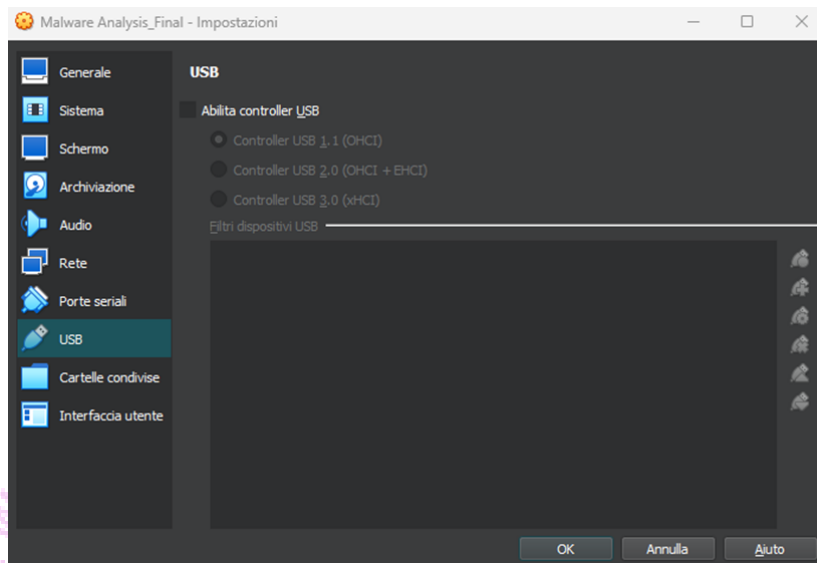
Procediamo quindi con la creazione del salvataggio, clicchiamo quindi su “crea” e successivamente su “Ok” dopo aver scelto il nome dello snapshot:



2. **Isolamento della rete:** Disabilitiamo o limitiamo l'accesso di rete della macchina virtuale per evitare la comunicazione del malware con altri sistemi. Per farlo bisogna disabilitare l'adattatore di rete della macchina virtuale o configurarlo come rete interna, in modo tale che non possa collegarsi alla nostra rete internet:



3. **Isolamento porte usb:** Per garantire l'isolamento del Malware e difendere anche il sistema principale su cui è attiva la VM è importante isolare le porte usb per impedire la propagazione del Malware su eventuali dispositivi connessi o reti, quindi disabilitiamo il "controller USB".



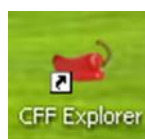
I Tools che utilizzeremo per questa Build Week saranno:

1. CFF Explorer;
2. IDA Pro Free;
3. OllyDBG
4. Process Monitor

CFF Explorer

CFF Explorer è uno strumento utilizzato principalmente per analizzare e modificare file eseguibili, come i file PE (Portable Executable) utilizzati in ambienti Windows. Questo strumento offre diverse funzionalità, tra cui:

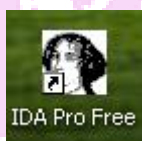
1. **Analisi delle strutture interne:** Permette agli utenti di visualizzare e analizzare le diverse strutture interne dei file eseguibili, come le sezioni, le intestazioni, le tabelle di esportazione e importazione, i dati di debug e altro ancora.
2. **Modifica dei file eseguibili:** Consente agli utenti di modificare varie proprietà dei file eseguibili, come le intestazioni, le sezioni, le tabelle di importazione e esportazione e altre informazioni correlate.
3. **Visualizzazione delle risorse:** Permette di visualizzare le risorse incorporate nei file eseguibili, come icone, bitmap, stringhe e manifesti.
4. **Analisi del codice:** Fornisce strumenti per analizzare il codice assembly incorporato nei file eseguibili, facilitando l'analisi dei malware e la comprensione del funzionamento interno dei programmi.



IDA Pro Free

IDA Pro è uno dei tool più potenti e ampiamente utilizzati per l'analisi dei malware e il reverse engineering dei programmi. Le sue funzionalità principali includono:

1. **Disassemblaggio:** IDA Pro converte il codice binario di un eseguibile in codice assembly leggibile, permettendo agli analisti di esaminare il funzionamento interno del programma.
2. **Analisi del flusso di controllo:** Il tool visualizza il flusso di controllo del programma, inclusi i salti condizionali e incondizionali, i loop e le chiamate di funzione, facilitando la comprensione della logica del programma.
3. **Analisi dei dati:** IDA Pro identifica e analizza le strutture dati utilizzate dal programma, come le variabili locali, le tabelle di funzioni e le stringhe di testo, aiutando gli analisti a comprendere come vengono utilizzati i dati nel programma.
4. **Reverse engineering:** IDA Pro supporta il reverse engineering dei programmi, consentendo agli utenti di modificare il codice sorgente, aggiungere commenti e annotazioni, e analizzare e modificare il comportamento del programma.
5. **Supporto per diverse architetture:** IDA Pro supporta un'ampia gamma di architetture di processori e formati di file eseguibili, rendendolo adatto per l'analisi di malware su diverse piattaforme.



OllyDBG

OllyDBG è uno strumento di debugging e reverse engineering ampiamente utilizzato nel campo della sicurezza informatica e dell'analisi dei malware. Le sue principali funzionalità includono:

1. **Debugging:** OllyDBG consente agli analisti di eseguire programmi in un ambiente controllato, interrompendo l'esecuzione del programma in punti specifici per ispezionare lo stato del sistema, come il contenuto della memoria e dei registri, e per analizzare il flusso di esecuzione del codice.
2. **Analisi del codice assembly:** Il tool visualizza il codice assembly del programma, consentendo agli analisti di esaminare il funzionamento interno del programma e identificare le istruzioni eseguite dal malware.
3. **Reverse engineering:** OllyDBG supporta il reverse engineering dei programmi, consentendo agli utenti di modificare il codice del programma, aggiungere commenti, eseguire patching e analizzare il comportamento del programma.
4. **Supporto per plugin:** Il tool supporta i plugin, che estendono le sue funzionalità base e consentono agli utenti di personalizzare l'ambiente di analisi in base alle proprie esigenze specifiche.

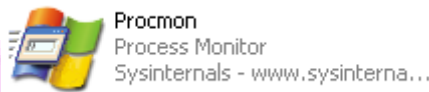


OLLYDBG
OllyDbg, 32-bit analysing deb...

Process Monitor

Process Monitor (o Procmon) è uno strumento di monitoraggio del sistema sviluppato da Microsoft che consente agli utenti di visualizzare in tempo reale le attività dei processi e dei file nel sistema operativo Windows. Le sue funzionalità principali includono:

1. **Monitoraggio delle attività del processo:** Process Monitor cattura e visualizza informazioni dettagliate su tutte le attività dei processi in esecuzione, inclusi i processi in avvio, la creazione e la chiusura dei processi, nonché le operazioni effettuate da ciascun processo.
2. **Monitoraggio delle operazioni sui file:** Il tool registra tutte le operazioni eseguite sui file, come la lettura, la scrittura, la creazione e la cancellazione, consentendo agli utenti di comprendere come i processi interagiscono con i file sul sistema.
3. **Filtraggio e ricerca:** Process Monitor offre funzionalità avanzate di filtraggio e ricerca, che consentono agli utenti di concentrarsi sulle attività specifiche o di trovare rapidamente informazioni rilevanti tra un grande volume di dati di monitoraggio.
4. **Analisi delle prestazioni:** Il tool fornisce informazioni utili sulle prestazioni del sistema, inclusi i tempi di risposta dei processi e le statistiche sull'utilizzo della CPU e del disco, che possono essere utili per l'ottimizzazione delle prestazioni e il troubleshooting dei problemi di sistema.



Task Giorno 1

Traccia:

Con riferimento al file eseguibile Malware_Build_Week_U3, rispondete ai seguenti quesiti utilizzando i tool e le tecniche apprese nelle lezioni teoriche;

- Quanti parametri sono passati alla funzione Main()?
- Quante variabili sono dichiarate all'interno della funzione Main()?
- Quali sezioni sono presenti all'interno del file eseguibile?
- Quali librerie importa il Malware? Per ognuna delle librerie impostate, fate delle ipotesi sulla base della sola analisi statica delle funzionalità che il Malware potrebbe implementare.

Utilizzare le funzioni che sono richiamate all'interno delle librerie per supportare le vostre ipotesi.

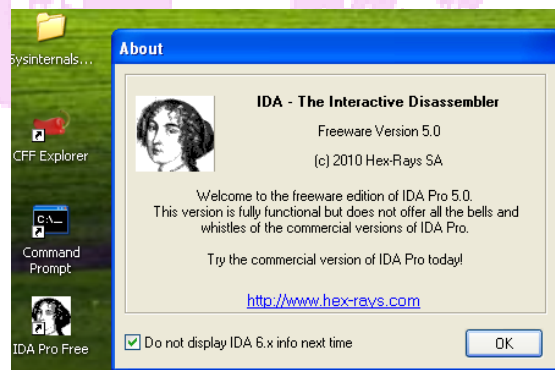
Analisi statica con IDA Pro

Per prima cosa utilizziamo il **disassembler IDA Pro** per aprire il file fornitoci ad inizio della BW3.


IDA Pro è installato sul desktop della Virtual Machine Windows XP fornitaci per effettuare operazioni di malware analysis in condizioni di sicurezza.

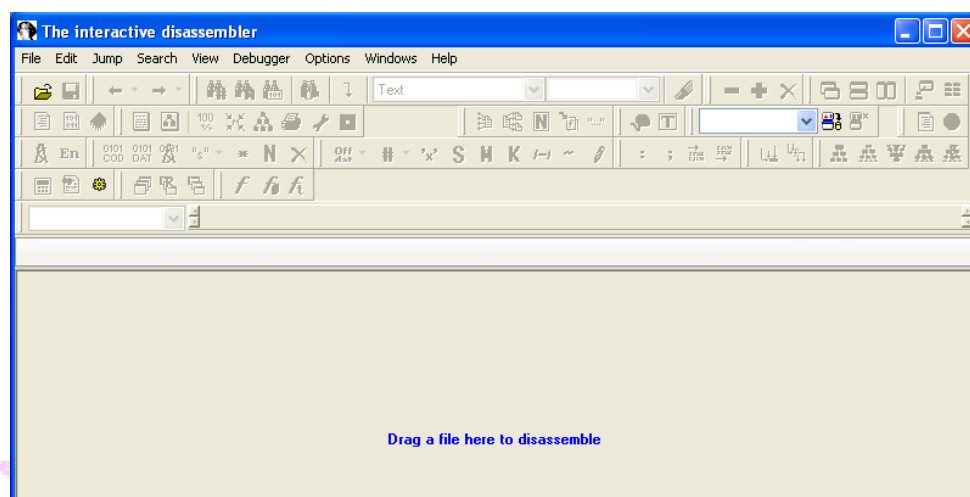
Di seguito si riportano i passaggi necessari per procedere all'analisi.

Si deve, innanzitutto, cliccare sull'icona del desktop e "ok" per avviare il tool.



Nella schermata di cui alla figura sotto IDA offre la possibilità di impostare l'eseguibile del malware da analizzare in due modi:

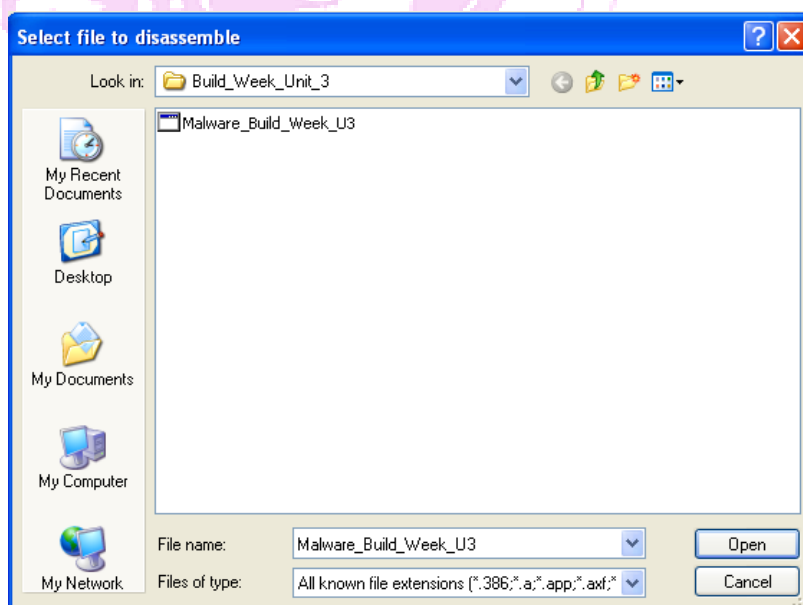
1. Trascinando direttamente il file per il disassemblaggio.
2. selezionando l'icona  per aprire il file eseguibile specificando il percorso, o semplicemente, navigando il file system.



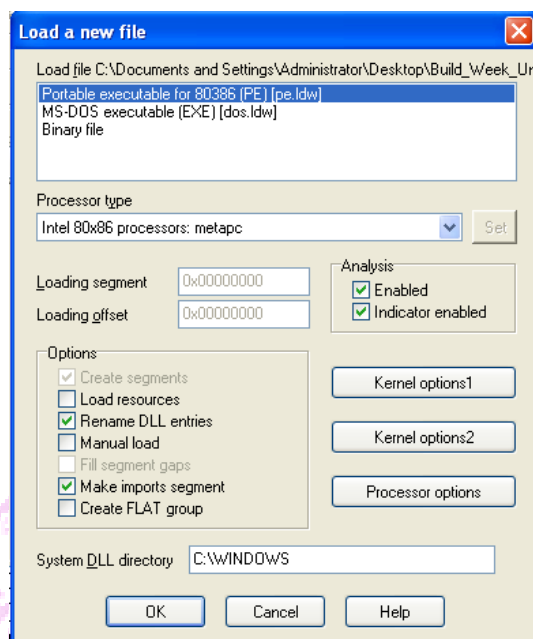
Scegliamo di navigare il file system cliccando sull'icona a forma di cartella.

Nella finestra di cui alla figura in basso si può scegliere il percorso del file eseguibile da scaricare, che nel nostro caso è il malware contenuto nella cartella del Desktop "Build_Week_Unit_3".

A questo punto è sufficiente selezionare l'eseguibile e cliccare su "Open".



IDA Pro presenta una finestra nella quale identifica di default sia l'architettura del processore (Intel x86) che il formato del file (PE). Non aggiungiamo nulla e clicchiamo su ok per procedere al caricamento del file.



Completato il caricamento del file eseguibile, IDA PRO presenta l'interfaccia di analisi che dispone di 9 schede con diverse funzioni:

1. **IDA View-A:** è il Disassembly panel, il pannello principale nel quale viene mostrata la traduzione del codice macchina dell'eseguibile in codice Assembly.
2. **Hex View-A:** è la scheda che permette di visualizzare il contenuto del file come insieme di dati in forma esadecimale.
3. **Exports:** mostra le funzioni create dal malware e condivisibili per l'utilizzo da altri programmi.
4. **Imports:** mostra le funzioni importate dall'eseguibile.
5. **Names:** associa ogni indirizzo con un nome (funzione, variabile, parametro o string).
6. **Functions:** mostra tutte le funzioni all'interno dell'eseguibile.
7. **Strings:** mostra tutte le stringhe di cui è composto il malware.
8. **Structures:** consente di identificare e gestire la struttura dei dati.
9. **Enums:** permette di visualizzare e gestire gli enumeratori, cioè strutture dati che definiscono un insieme di costanti con nomi significati associati a valori numerici.

Per rispondere ai primi due quesiti della traccia si è utilizzata la **modalità testuale del disassembly panel**, semplicemente cliccando sulla barra spaziatrice della tastiera mentre si è sul pannello grafico.

In questo modo si sono raccolte e analizzate le seguenti informazioni, di cui andiamo a fornire una descrizione specifica:

```
.text:004011D0 ; int __cdecl main(int argc,const char **argv,const char *envp)
.text:004011D0 _main          proc near          ; CODE XREF: start+AF1p
.text:004011D0
.text:004011D0 hModule      = dword ptr -11Ch
.text:004011D0 Data        = byte ptr -118h
.text:004011D0 var_8       = dword ptr -8
.text:004011D0 var_4       = dword ptr -4
.text:004011D0 argc        = dword ptr 8
.text:004011D0 argv        = dword ptr 0Ch
.text:004011D0 envp        = dword ptr 10h
.text:004011D0
```

Chiamata alla funzione main ()

```
.text:004011D0 ; int __cdecl main(int argc,const char **argv,const char *envp)
.text:004011D0 _main          proc near          ; CODE XREF: start+AF1p
```

Quante variabili sono passate dalla funzione main ()? Variabili: 4 con offset negativo

```
.text:004011D0 hModule      = dword ptr -11Ch
.text:004011D0 Data        = byte ptr -118h
.text:004011D0 var_8       = dword ptr -8
.text:004011D0 var_4       = dword ptr -4
```

Quanti parametri sono passati dalla funzione main()? Parametri: 3 con offset positivo

```
.text:004011D0 argc        = dword ptr 8
.text:004011D0 argv        = dword ptr 0Ch
.text:004011D0 envp        = dword ptr 10h
```

Parametri nella funzione main()

```
.text:004011D0 argc        = dword ptr 8
.text:004011D0 argv        = dword ptr 0Ch
.text:004011D0 envp        = dword ptr 10h
```

I parametri sono argomenti che vengono passati alla funzione per eseguire operazioni specifiche e sono individuati da un **offset positivo**, indicante la distanza rispetto ad un valore di riferimento, presumibilmente il frame pointer.

La funzione main(), come possiamo vedere dall'immagine evidenziata dal rettangolo rosso, accetta tre parametri secondo la sua firma standard: int argc, const char *argv[], e const char *envp[].

1. **argc**: Questo parametro indica il numero totale di argomenti passati dalla riga di comando al programma. È un valore intero che include il nome del programma stesso.
2. **argv[]**: Si tratta di un array di stringhe (puntatori a caratteri) che contiene gli argomenti effettivi passati dalla riga di comando al programma. L'elemento argv[0] rappresenta il nome del programma, mentre gli altri elementi contengono gli argomenti aggiuntivi.
3. **envp[]**: Questo è un array di stringhe che contiene le variabili di ambiente disponibili per il processo. Anche se non è sempre utilizzato o richiesto in tutte le implementazioni della funzione main(), la sua presenza suggerisce che il programma potrebbe interagire con le variabili d'ambiente.

In sintesi, questi parametri forniscono al programma informazioni importanti sulla riga di comando e sulle variabili di ambiente, consentendo al programma di interagire in modo dinamico con l'ambiente operativo.

Variabili nella funzione main()

Quante variabili sono passate dalla funzione main ()? Variabili: 4 con offset negativo

```
.text:00401100 hModule      = dword ptr -11Ch  
.text:00401100 Data        = byte ptr -118h  
.text:00401100 var_8       = dword ptr -8  
.text:00401100 var_4       = dword ptr -4
```

Definizione di variabile e variabili di una funzione

Le **variabili** sono locazioni di memoria identificate da un nome e utilizzate per memorizzare dati durante l'esecuzione di un programma. Inoltre, ogni variabile può essere utilizzata per accedere ai dati memorizzati nella sua posizione di memoria.

Una **variabile di una funzione** è un'area di memoria utilizzata per memorizzare dati durante l'esecuzione di quella funzione all'interno di un programma. Questi dati possono essere numeri, stringhe, strutture di dati o altri tipi di informazioni manipolate dalla funzione.

Le variabili di una funzione sono variabili locali in quanto sono definite all'interno della specifica funzione e non sono accessibili e utilizzabili da altre funzioni o parti del programma.

Le quattro variabili della funzione main del malware sono state individuate tramite l'**offset negativo**, ovvero la distanza o differenza rispetto ad un valore di riferimento, che può essere la base dello stack o un registro della CPU.

La sintassi delle variabili osservate è la seguente:

Nome variabile = <dimensione> ptr <offset negativo da un valore di riferimento > dove:

- **hmodule, data, var_8 e var_4**: sono i nomi delle variabili.
- **DWORD ptr e byte ptr**: indicano a che tipo di dato, e la sua dimensione, punta la variabile.
In particolare, DWORD indica un intero a 32 bit mentre byte indica un singolo byte di dati, ovvero 8 bit.
Quindi, DWORD ptr indica che l'area di memoria punta ad un dato di tipo double word, ovvero di 32 bit (4 byte) mentre byte ptr indica che l'area di memoria punta a un singolo byte di dati (8 bit).
- **- 11Ch, - 118h, - 8, - 4**: sono la distanza da un valore di riferimento, il frame pointer dello stack.

Quindi in relazione alla globalità dei componenti menzionati:

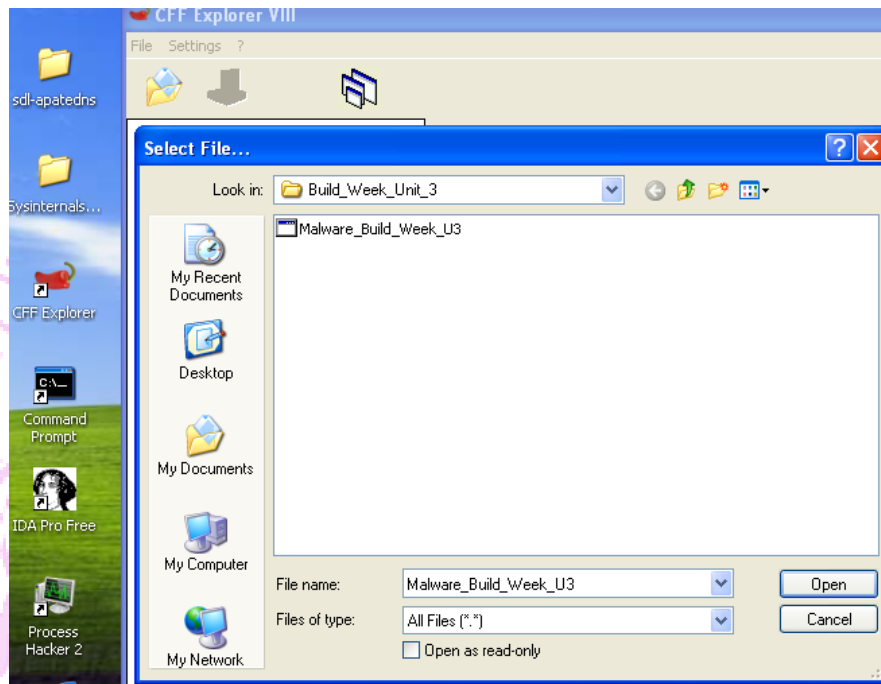
1. **hmodule = dword ptr -11Ch**: la variabile "hmodule" è un puntatore a un dato double word di 32 bit e dista - 284 byte (11CH esadecimale) rispetto al frame pointer, il puntatore di base dello stack.
2. **Data = byte ptr -118h**: la variabile "data" è un puntatore ad un byte con offset di 118h rispetto al frame pointer. Quindi, si trova a - 280 byte (o 118h esadecimale) di distanza rispetto al frame pointer, al puntatore della base dello stack.
3. **var_8 = dword ptr -4**: la variabile "var_8" è un puntatore ad una double word di 32 bit e dista 4 byte dal frame pointer, dal puntatore della base dello stack.
4. **var_4 = dword ptr -4**: la variabile "var_4" è un puntatore ad una double word di 32 bit e dista 4 byte dal frame pointer, dal puntatore della base dello stack.

Analisi statica con CFF Explorer

Per rispondere agli ultimi due quesiti della traccia si è analizzato staticamente il file eseguibile del malware tramite il tool **CFF Explorer**.

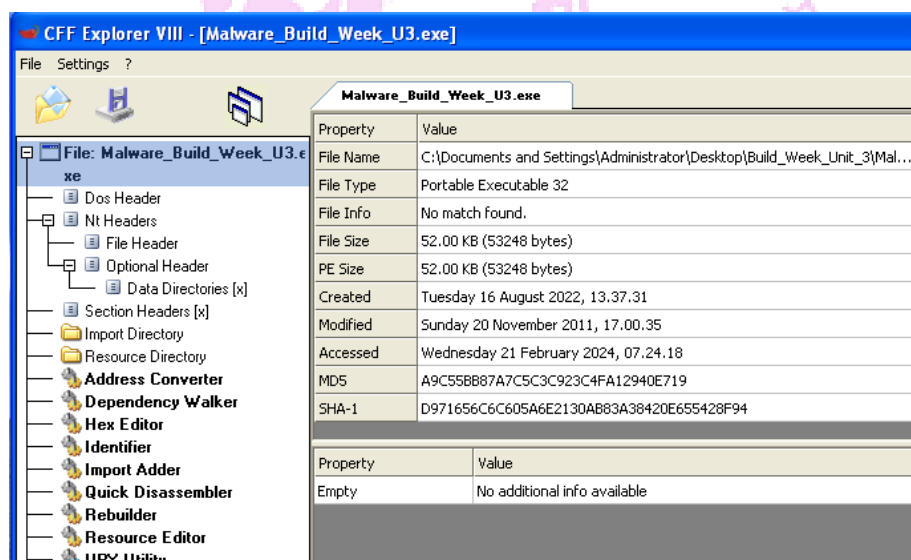
Una volta cliccato sull'icona rossa del tool nel desktop della VM, nella schermata che si apre, si ricerca il file eseguibile

di cui si vuole analizzare il contenuto: si seleziona l'icona  e, nel desktop, la cartella "Build_Week_Unit_3". In questo modo, con un semplice doppio click, si imposta il file "Malware_Build_Week_U3".



Con il doppio click, si apre la schermata iniziale di CFF Explorer relativa al file eseguibile selezionato.

Come si può vedere, sulla sinistra è presente un menù con varie opzioni per visualizzare e analizzare la struttura del file eseguibile, formato .exe.



Che cosa sono un file eseguibile e il formato PE?

Un **file eseguibile** è un tipo di file informatico che contiene istruzioni e dati codificati in linguaggio macchina, il formato comprensibile per il processore, che possono essere eseguiti direttamente dal sistema operativo o dall'interprete.

Questi file sono progettati per essere eseguiti dal sistema operativo o da un interprete e avviare un programma o un'applicazione o un servizio. Quindi, un file eseguibile può essere un programma, un'applicazione o un servizio che svolge una specifica funzione sul computer.

In altre parole, questo tipo di file, una volta avviato da un computer, esegue, tramite il suo contenuto di istruzioni e dati, un programma, un'applicazione o un servizio.

Il **formato Portable Executable (PE)** è uno dei formati più comuni utilizzati per i file eseguibili su sistemi operativi Windows. In particolare, supporta una vasta gamma di tipi di file, tra cui applicazioni eseguibili (.exe), librerie condivise (.dll), driver di dispositivo (.sys) e altro ancora.

Il **termine ".exe"** indica l'estensione di file comunemente associata ai file eseguibili su sistemi Windows.

Tuttavia, è possibile trovare altri tipi di file con estensione ".exe" su sistemi Windows, come i file di script eseguibili (ad esempio i file batch o i file di script PowerShell) o i file di installazione degli applicativi.

Il formato PE organizza i dati e le istruzioni all'interno del file eseguibile in una serie di sezioni, ciascuna delle quali ha un ruolo specifico nel funzionamento complessivo del programma.

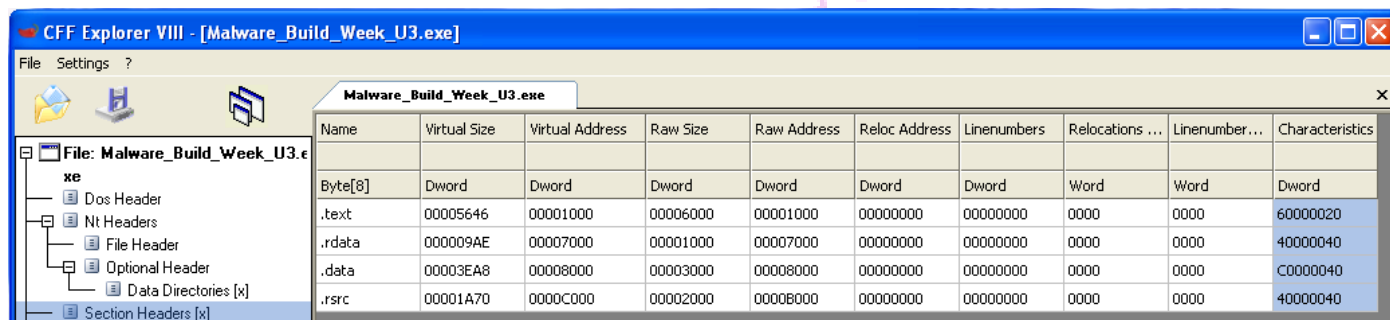
Sezioni

Le **sezioni** di un file eseguibile sono divisioni logiche del file che contengono dati, istruzioni specifiche o risorse ulteriori. Ogni sezione ha un ruolo specifico e contribuisce al funzionamento complessivo del file eseguibile. Le sezioni sono tipicamente presenti nei file eseguibili di vari formati, come ad esempio i file binari eseguibili (detti per semplicità eseguibili), come quelli di Windows, le librerie condivise (DLL) e gli oggetti compilati.

Da tale definizione si può comprendere come le sezioni di un eseguibile malware siano componenti cruciali per il suo funzionamento e la sua efficacia.

Infatti, il codice eseguibile, i dati e le risorse contenuti nelle varie sezioni sono sfruttate dal malware per eseguire funzioni dannose specifiche o per nascondere il proprio codice dannoso.

Tornando all'analisi statica con il tool **CFF Explorer**, selezioniamo l'opzione "**Section Headers [x]**":



Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations ...	Linenumber...	Characteristics
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	00005646	00001000	00006000	00001000	00000000	00000000	0000	0000	60000020
.rdata	000009AE	00007000	00001000	00007000	00000000	00000000	0000	0000	40000040
.data	00003EA8	00008000	00003000	00008000	00000000	00000000	0000	0000	C0000040
.rsrc	00001A70	0000C000	00002000	0000B000	00000000	00000000	0000	0000	40000040

All'interno del file eseguibile in questione, sono presenti diverse sezioni che svolgono ruoli specifici nell'esecuzione del programma, come possiamo notare nel riquadro verde presente nell'immagine vengono dichiarate quattro variabili:

1. **.text**: Questa sezione contiene il codice eseguibile del programma. Qui sono presenti le istruzioni assembler che vengono eseguite dal processore durante l'esecuzione del programma. È la parte "attiva" del programma che esegue le operazioni specificate dall'autore del software.
2. **.rdata**: In questa sezione sono contenuti dati di sola lettura, come costanti e stringhe di testo. È possibile che qui siano presenti anche riferimenti a funzioni importate da librerie esterne, sebbene siano solo dati e non codice eseguibile.
3. **.data**: La sezione .data contiene dati inizializzati che possono essere modificati durante l'esecuzione del programma. Questi dati potrebbero includere variabili globali, array o altre strutture di dati utilizzate dal programma.
4. **.rsrc**: Questa sezione contiene le risorse del programma, come icone, stringhe localizzate, immagini e altri dati non eseguibili utilizzati dall'applicazione. Queste risorse sono spesso utilizzate per l'interfaccia utente e altri scopi non direttamente legati alla logica del programma.

.text	Istruzioni da eseguire per la CPU al avvio del programma, spesso read-only per prevenire al programma di modificare accidentalmente le istruzioni.
.rdata	Read-only initialized data, include generalmente le informazioni circa le librerie e le funzioni importate ed esportate dall'eseguibile
.data	Initialized data (free format) include tipicamente i dati / le variabili globali del programma eseguibile
.rsrc	Include le risorse utilizzate dall'eseguibile come ad esempio icone, immagini, menù e stringhe che non sono parte dell'eseguibile stesso.

Quali librerie importa il malware?

Il formato PE al suo interno contiene delle informazioni necessarie al sistema operativo per capire come gestire il codice del file, come ad esempio le librerie.

Le **librerie** (anche chiamate moduli) sono insieme di funzioni predefinite pronte per l'uso, senza doverle riscrivere ogni volta.

Quando un programma ha bisogno di una funzione «chiama» una libreria al cui interno è definita la funzione necessaria.

Selezioniamo, nuovamente dal menu di CFF Explorer, l'opzione **"Import Directory"**:

Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.dll	51	00007534	00000000	00000000	0000769E	0000700C
ADVAPI32.dll	2	00007528	00000000	00000000	000076D0	00007000

Come possiamo vedere sono due le librerie importate dal Malware:

KERNEL32.dll: è una delle principali librerie di sistema di Windows utilizzata per effettuare operazioni di base del sistema operativo. Questa libreria offre funzioni di basso livello, come la gestione della memoria, operazioni di input/output, manipolazione dei file, creazione e gestione di processi e thread e molte altre.

ADVAPI32.dll: Fornisce funzionalità relative alla gestione della sicurezza, al registro di sistema e ai servizi di Windows e alle risorse di sistema.

Questa libreria è spesso utilizzata per eseguire operazioni che richiedono privilegi elevati, come la modifica delle impostazioni di sicurezza o l'accesso a parti sensibili del registro di sistema.

Ma cosa vuol dire .dll?

DLL è l'acronimo di "Dynamic Link Libraries" e qualifica KERNEL32 e ADVAPI32 come librerie di collegamento dinamico in ambienti Windows. Le DLL vengono utilizzate per fornire codice condiviso, risorse e funzionalità che possono essere richiamate da diversi programmi o processi all'interno del sistema operativo Windows.

In sostanza, queste librerie sono "contenitori" di codici, risorse e funzioni che sono condivise tra programmi Windows. Le DLL sono "dinamiche" perché vengono caricate in memoria solo quando necessario durante l'esecuzione di un programma. Ciò consente di risparmiare spazio su disco e memoria, poiché le DLL possono essere condivise tra più programmi. Inoltre, le DLL possono essere aggiornate indipendentemente dai programmi che le utilizzano, semplificando il processo di manutenzione del software.

Andiamo ora ad analizzare le funzioni contenute nelle librerie importate dal malware cliccando su ciascuna delle librerie individuate da CFF Explorer, che mostra un pannello nella parte inferiore della schermata, come in figura di seguito.

CFF Explorer VIII - [Malware_Build_Week_U3.exe]

File Settings ?

Malware_Build_Week_U3.exe

Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
0000769E	N/A	000074EC	000074F0	000074F4	000074F8	000074FC
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.dll	51	00007534	00000000	00000000	0000769E	0000700C
ADVAPI32.dll	2	00007528	00000000	00000000	000076D0	00007000

OFTs	FTs (IAT)	Hint	Name
Dword	Dword	Word	szAnsi
00007632	00007632	0295	SizeofResource
00007644	00007644	01D5	LockResource
00007654	00007654	01C7	LoadResource
00007622	00007622	02BB	VirtualAlloc
00007674	00007674	0124	GetModuleFileNameA
0000768A	0000768A	0126	GetModuleHandleA
00007612	00007612	00B6	FreeResource
00007664	00007664	00A3	FindResourceA
00007604	00007604	001B	CloseHandle

Per quanto riguarda **KERNEL32.dll**, le funzioni utilizzate dall'eseguibile sono **51** e riguardano:

1. Gestione delle risorse

Si tratta di funzioni che permettono di individuare (FindResource), accedere (LockResource) e ottenere informazioni (SizeofResource) sulle risorse incorporate all'interno di un eseguibile di Windows.

2. Gestione della memoria

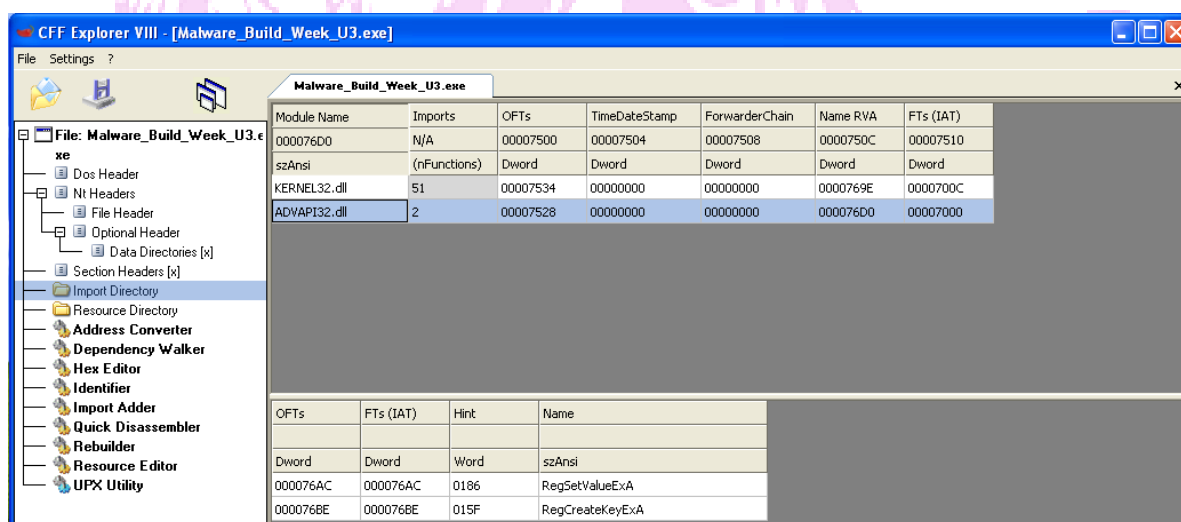
Le funzioni permettono di gestire la memoria, per esempio prevedendone l'allocazione (VirtualAlloc e Heap Alloc) e la liberazione (HeapDestroy, VirtualFree).

3. Gestione dei processi e thread

Le funzioni di questa tipologia servono a ottenere informazioni sui processi (GetModuleHandle, GetCurrentProcess) e sul sistema operativo (GetVersion), a controllare i processi (TerminateProcess), a manipolare file (CreateFileA, WriteFile, ReadFile) e ad effettuare il caricamento di librerie in maniera dinamica (GetProcAddress e LoadLibraryA).

4. Manipolazione delle stringhe

Infine, nell'eseguibile riscontriamo funzioni che si occupano della conversione di codifica (MultiByteToWideChar) e della mappatura e manipolazione delle stringhe (LCMapstringA e GetStringType).



Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
000076D0	N/A	00007500	00007504	00007508	0000750C	00007510
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.dll	51	00007534	00000000	00000000	0000769E	0000700C
ADVAPI32.dll	2	00007528	00000000	00000000	000076D0	00007000

OFTs	FTs (IAT)	Hint	Name
Dword	Dword	Word	szAnsi
000076AC	000076AC	0186	RegSetValueExA
000076BE	000076BE	015F	RegCreateKeyExA

Per quanto riguarda **ADVAPI32.DLL**, l'eseguibile utilizza due funzioni per l'interazione con il registro di sistema di Windows:

1. RegCreateKeyExA:

Questa funzione viene utilizzata per creare una nuova chiave di registro o aprire una chiave di registro esistente. Se la chiave specificata non esiste, questa funzione la crea. Se esiste, questa funzione semplicemente la apre. È inoltre possibile specificare varie opzioni durante la creazione della chiave, come le autorizzazioni di accesso e le opzioni di creazione.

2. RegSetValueExA:

Questa funzione viene utilizzata per impostare il valore di una chiave di registro esistente o creare un nuovo valore per una chiave di registro specificata. Inoltre, consente di specificare il tipo dei dati del valore, i dati stessi e la loro lunghezza. Se il valore specificato non esiste già, questa funzione lo crea. Se esiste, sovrascrive il valore esistente con i nuovi dati forniti.

Deduzioni sulle funzionalità implementate dal malware

Il malware, basandosi sulle funzioni delle librerie KERNEL32.dll e ADVAPI32.dll, dimostra di avere una complessa gamma di funzionalità che gli consentono di eseguire operazioni dannose e di manipolare il sistema operativo a proprio vantaggio.

Le funzioni della libreria KERNEL32.dll offrono al malware una vasta gamma di strumenti per manipolare il sistema operativo. Ad esempio, le funzioni per la gestione dei processi come CreateProcess e TerminateProcess consentono al malware di avviare e terminare processi all'interno del sistema, potenzialmente per eseguire azioni dannose o per influenzare il comportamento del sistema.

Allo stesso modo, le funzioni per la gestione della memoria come VirtualAlloc e VirtualFree consentono al malware di allocare e liberare spazio di memoria, che può essere utilizzato per caricare e eseguire codice dannoso o memorizzare dati sensibili. Inoltre, le funzioni per la manipolazione dei file come CreateFile e ReadFile permettono al malware di creare, leggere e modificare file sul sistema, aprendo la porta a possibili attacchi o manipolazioni. Queste funzionalità, insieme alla capacità di manipolare le stringhe di testo e altre operazioni di sistema, forniscono al malware gli strumenti necessari per eseguire una varietà di attività dannose, dalla persistenza nel sistema alla manipolazione dei dati sensibili.

D'altra parte, le funzioni della libreria ADVAPI32.dll, come RegSetValueExA e RegCreateKeyExA, dimostrano che il malware effettua la manipolazione del registro di sistema di Windows. Utilizzando queste funzioni, è in grado di modificare le impostazioni del sistema, creando nuove chiavi di registro o modificando quelle esistenti per garantire, per esempio, la sua persistenza nel sistema.

Infatti, potrebbe creare voci di registro per eseguirsi automaticamente all'avvio del sistema o per nascondere la propria presenza tra i processi di sistema. Inoltre, la manipolazione del registro di sistema può essere utilizzata per memorizzare informazioni di configurazione malevole o per influenzare il comportamento del sistema, permettendo al malware di eseguire azioni dannose in modo discreto e persistente.

Conclusioni

Si è visto come il file eseguibile sia classificabile come un malware per via delle funzionalità implementate che gli consentono di manipolare il sistema operativo a 360°, influenzandone gestione delle risorse, memoria, processi, file, stringhe e registro di sistema.

Di particolare rilevanza, per il tentativo di profilazione del malware, sono la presenza e l'utilizzo di tre funzioni: SizeofResource(), LoadResource() e FindResource().

Queste indicano che il malware potrebbe essere un dropper, un tipo di malware progettato per scaricare e installare altri componenti dannosi sul sistema compromesso. In particolare, queste funzioni sono utilizzate per accedere alle risorse incorporate nel file eseguibile, che di solito si trovano nella sezione .rsrc.

Ecco come potrebbero essere utilizzate queste funzioni da un dropper nel contesto di un attacco malware:

FindResource(): Questa funzione viene utilizzata per trovare la posizione di una risorsa specifica all'interno del file eseguibile. Nel contesto del malware, questa risorsa potrebbe essere un altro pezzo di codice dannoso o un payload che il dropper è destinato a installare sul computer della vittima. Ad esempio, il dropper potrebbe cercare una risorsa contenente un file eseguibile o un archivio compresso contenente il payload.

SizeofResource(): Dopo aver trovato la risorsa desiderata con FindResource(), il dropper utilizza SizeofResource() per determinare la dimensione della risorsa trovata. Questo è importante perché il dropper deve sapere quanto spazio allocare quando estrae la risorsa dal file eseguibile. Conoscere le dimensioni della risorsa consente al dropper di allocare la memoria necessaria per caricare e manipolare correttamente il payload senza rischio di sovrascrivere altre parti della memoria o causare errori nell'elaborazione dei dati.

In parole povere, la presenza di queste funzioni nel codice del malware suggerisce che il **programma potrebbe essere un dropper**, il cui scopo principale è quello di recuperare e installare componenti dannosi aggiuntivi sul sistema. Infatti, i dropper danno esecuzione al payload malevolo, il quale può essere un malware più sofisticato come un ransomware, un trojan o una backdoor.

Task Giorno 2

Con riferimento al Malware in analisi, spiegare:

1. Lo scopo della funzione chiamata alla locazione di memoria 00401021
2. Come vengono passati i parametri alla funzione alla locazione 00401021
3. Che oggetto rappresenta il parametro alla locazione 00401017
4. Il significato delle istruzioni comprese tra gli indirizzi 00401027 e 00401029
5. Con riferimento all'ultimo quesito, tradurre il codice Assembly nel corrispondente costruito C.
6. Valuta ora la chiamata alla locazione 00401047, qual è il valore del parametro "ValueName"?

Nel complesso delle due funzionalità appena viste, spiega quale funzionalità sta implementando il Malware in questa sezione.

Come primo passaggio andiamo ad identificare la funzione chiamata alla locazione **00401021** dello stack di memoria.

<pre> * .text:00401015 * .text:00401017 * .text:0040101C * .text:00401021 * .text:00401027 * .text:00401029 * .text:0040102B </pre>	<pre> push 0 ; Reserved push offset SubKey ; "SOFTWARE\\Microsoft\\Win push 80000002h : hKey call ds:RegCreateKeyExA test eax, eax jz short loc_401032 mov eax, 1 </pre>
---	---

Come possiamo vedere nel riquadro rosso in figura alla locazione **00401021** troviamo la funzione **RegCreateKeyExA**

<pre> * .text:00401027 * .text:00401029 * .text:0040102B * .text:00401030 </pre>	<pre> test eax, eax jz short loc_401032 mov eax, 1 jmp short loc_40107B </pre>
--	--

```

* .text:0040103C
* .text:0040103E
* .text:00401043
* .text:00401046
* .text:00401047

```

```

push    0 ; Reserved
push    offset ValueName ; "GinaDLL"
mov     eax, [ebp+hObject]
push    eax ; hKey
call    ds:RegSetValueExA

```

Qual è lo scopo della funzione **RegCreateKeyExA**?

Premessa

L'invocazione della funzione **RegCreateKeyExA** in questo contesto riveste un ruolo fondamentale nelle attività nefaste del software dannoso. Tale interfaccia di programmazione delle applicazioni di Windows non solo consente la creazione di nuove voci di registro, ma anche l'accesso a voci esistenti con specifici livelli di autorizzazione. Costituisce un mezzo diretto attraverso il quale il software dannoso può stabilire un punto di ancoraggio nel sistema compromesso, permettendo potenzialmente al malware di auto-avviarsi ad ogni riavvio del sistema, modificare il comportamento del sistema stesso o occultare altre sottochiavi e valori maligni. Questo potrebbe altresì essere impiegato per sovrascrivere le impostazioni di sicurezza, di rete o altre configurazioni, agevolando ulteriori azioni dannose del malware o garantendo la sua persistenza e occultamento nell'ambiente compromesso.

Analisi approfondita ed eventuali casistiche:

Persistenza: Il malware può creare chiavi di registro auto-avvianti (AUTORUN) per garantirsi la persistenza nel sistema, resistendo anche alla rimozione manuale o alla reinstallazione del sistema operativo.

Evasione: La modifica di chiavi specifiche del Registro di sistema può ostacolare l'analisi e la rimozione del malware, rendendolo invisibile ai software antivirus e agli strumenti di analisi forense.

Modifica del comportamento del sistema: Il malware può modificare le impostazioni di sistema, disattivare le funzionalità di sicurezza o compromettere la configurazione di rete per ottenere un accesso non autorizzato o per recuperare dati sensibili.

Elevazione dei privilegi: Il malware può sfruttare vulnerabilità del Registro di sistema per ottenere privilegi elevati all'interno del sistema, acquisendo il controllo completo del dispositivo.

Esempi di possibili comportamenti del malware:

Botnet: Il malware può creare chiavi di registro per aggiungere il computer infetto a una botnet, una rete di computer controllati da remoto utilizzata per attività dannose come l'invio di spam o l'esecuzione di attacchi DDoS.

Ransomware: Il malware può crittografare i file dell'utente e creare una chiave di registro che contiene la chiave di decrittografia, utilizzata per estorcere denaro alla vittima in cambio del ripristino dei dati.

Keylogger: Il malware può installare un keylogger, un software che registra le sequenze di tasti premuti dall'utente, creando una chiave di registro per archiviare le informazioni rubate.

Misure di protezione:

AntiVirus e AntiMalware: L'utilizzo di software di sicurezza aggiornato può aiutare a prevenire l'infezione da malware e a bloccare l'accesso alle chiavi di registro dannose.

Analisi del Registro di sistema: Strumenti specializzati possono essere utilizzati per scansionare il Registro di sistema alla ricerca di chiavi e valori sospetti, identificando potenziali minacce.

Aggiornamenti di sicurezza: L'installazione tempestiva di aggiornamenti software di sicurezza rilasciate da Microsoft può correggere vulnerabilità note del Registro di sistema sfruttate dai malware.

Conclusione:

La funzione RegCreateKeyExA rappresenta un bersaglio primario per il malware, offrendo un punto di accesso per la compromissione del sistema e l'esecuzione di attività dannose. L'adozione di misure di sicurezza adeguate, l'utilizzo di software AntiVirus e AntiMalware e l'installazione di aggiornamenti di sicurezza sono fondamentali per proteggere il sistema da minacce informatiche.

Come vengono passati i parametri alla locazione indicata?

I parametri vengono passati sullo stack tramite la chiamata push, seguendo il principio LIFO (Last In, First Out). Questo è uno standard comune per le chiamate di funzione in linguaggi come il C e consente di trasferire un numero variabile di parametri in modo ordinato. Inoltre, si ipotizza che il puntatore a una variabile `phkResult` riceva l'handle della chiave aperta, presumibilmente una chiave di registro predefinita. Tuttavia, è rilevante notare che non viene richiamata la funzione `RegCloseKey` per liberare l'handle dopo il suo utilizzo. Questa mancanza può portare a perdite di risorse nel sistema e potenziali problemi di sicurezza, specialmente se il malware opera in modo persistente e crea un gran numero di chiavi senza liberarle correttamente.

Vantaggi:

Flessibilità: Permette di trasferire un numero variabile di parametri senza la necessità di modificarne la struttura.

Efficienza: Facilita l'accesso ai parametri all'interno della funzione, semplificando il processo di elaborazione.

Ordine logico: L'ordine LIFO garantisce che l'ultimo parametro inserito sia il primo ad essere utilizzato, utile in determinate situazioni.

Svantaggi:

Perdite di memoria: La mancata chiusura dell'handle può causare perdite di memoria nel sistema. Ogni handle aperto occupa una porzione di memoria, e se non rilasciato correttamente, può accumularsi nel tempo, causando un consumo eccessivo di risorse e un rallentamento del sistema.

Problemi di sicurezza: La mancata chiusura dell'handle può creare potenziali vulnerabilità di sicurezza. Un handle aperto può essere sfruttato da malware o hacker per ottenere accesso non autorizzato al sistema o per compromettere la chiave di registro associata.

Mitigazione dei rischi:

Gestione responsabile degli handle: È fondamentale richiamare la funzione **RegCloseKey** per ogni handle aperto, liberando la memoria associata e prevenendo perdite di risorse.

Verifica e controllo: È consigliabile implementare un sistema di verifica per assicurarsi che tutti gli handle vengano correttamente chiusi, evitando potenziali perdite di memoria o vulnerabilità di sicurezza.

Analisi approfondita

Locazione specifica: I parametri vengono passati alla locazione indicata dalla funzione chiamante. Questo offre un maggiore controllo sul posizionamento dei dati in memoria e può essere utile per ottimizzare l'accesso e l'elaborazione.

Gestione dell'handle: Il puntatore `phkResult` riceve l'handle della chiave aperta, che dovrebbe essere una chiave di registro predefinita. È fondamentale notare che la funzione **RegCloseKey** non viene mai richiamata per liberare l'handle dopo il suo utilizzo.

Qual è l'oggetto rappresentato alla locazione indicata?

```
.data:00408054 SubKey          db 'SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon',0
.data:00408054                ; DATA XREF: sub_401000+1770
.data:0040808A                align 4
```

Il parametro trasferito all'indirizzo 00401017 è un riferimento a una chiave di registro ampiamente conosciuta, utilizzata per avviare automaticamente le applicazioni durante l'avvio del sistema. Il malware sfrutta astutamente questa caratteristica per garantire l'esecuzione del suo codice senza richiedere l'interazione dell'utente, garantendo così la massima persistenza nel sistema. Sebbene questa tecnica sia comunemente impiegata in software legittimi per scopi come gli aggiornamenti automatici, nel contesto del malware rappresenta inequivocabilmente un segno di attività maligna.

Qual è il significato delle istruzioni comprese tra gli indirizzi 00401027 e 00401029

Questo codice assembly x86 esegue un controllo di uguaglianza tra il valore contenuto nelle registrazioni EAX e EAX

.text:00401027	test	eax, eax	
.text:00401029	jz	short loc_401032	
.text:0040102B	mov	eax, 1	
.text:00401030	jmp	short loc_40107B	
.text:00401032	; -----		
.text:00401032	loc_401032:		
.text:00401032			; CODE XREF: sub_401000+29↑j
.text:00401032	mov	ecx, [ebp+cbData]	

Queste istruzioni sono parte di un intricato processo di verifica dell'integrità in seguito a un'azione cruciale eseguita dal malware. Mediante un test sui bit del registro EAX, il software dannoso analizza se la precedente chiamata alla funzione **RegCreateKeyExA** ha avuto esito positivo. Questo controllo riveste un'importanza vitale nella logica di gestione del flusso del malware, poiché influenzerà le decisioni successive del programma. Nel caso in cui l'operazione non abbia avuto successo, il malware potrebbe intraprendere un percorso alternativo, tentare di rieseguire l'operazione o persino interrompere il proprio funzionamento per evitare la rilevazione da parte delle misure di sicurezza del sistema. Tale livello di adattabilità e reattività nel comportamento del malware evidenzia la sofisticatezza delle sue strategie per rimanere operativo e persistente nell'ambiente infetto.

L'istruzione "test" esegue una operazione bit a bit "and" tra i due valori e imposta i flag del processore in base al risultato. L'istruzione "jz" (jump if zero) esegue un salto condizionale all'indirizzo specificato "loc_401032" solo se lo Zero Flag è impostato a 1, il che significa che i due valori sono uguali. Quindi, questo codice sta confrontando il valore contenuto nella registrazione EAX con quello contenuto in EAX. Se i due valori sono uguali, il codice esegue un salto all'indirizzo "loc_401032", altrimenti continua eseguendo il codice successivo.

Con riferimento all'ultimo quesito, tradurre il codice Assembly nel corrispondente costruito

La traduzione del codice assembly fornito in un costruito C fornisce un equivalente ad alto livello più comprensibile per i programmatori e gli analisti di malware. Questa rappresentazione mostra come il malware sfrutti le istruzioni condizionali per prendere decisioni basate sul successo o fallimento delle sue operazioni, un concetto cruciale nella programmazione. La capacità del malware di adattarsi dinamicamente in base alle condizioni del sistema è evidenziata attraverso l'uso di costrutti condizionali nel codice C, riflettendo la sua natura reattiva e adattabile nell'ambiente infetto.

```
if (eax == 0) {  
    // Salta a loc_401032 se la chiamata a RegCreateKeyExA() è riuscita.  
}  
else {  
    eax = 1; // Imposta il valore di eax a 1 se la chiamata non è riuscita.  
}
```

Valutazione della chiamata alla locazione 00401047, qual è il valore del parametro "ValueName"?

Nella chiamata alla locazione 00401047, si osserva un'integrazione della funzione RegSetValueExA, dove il parametro "ValueName" è codificato come "GinaDLL". Questo suggerisce un tentativo del software dannoso di manipolare i punti di accesso critici del sistema operativo Windows, focalizzandosi sulla sovrascrittura della chiave di registro associata alla DLL di autenticazione grafica. Tale azione, se riuscita, potrebbe avere conseguenze gravi, permettendo al malware di interferire con il processo di autenticazione, acquisire le credenziali d'accesso, o aggirare i controlli di sicurezza. L'intento malizioso diventa evidente nella manipolazione della chiave di registro "SOFTWARE\Microsoft\Windows\CurrentVersion\Run", mirando a garantire l'avvio automatico del malware ad ogni riavvio del sistema. Questo approccio, comune tra i malware, consente una persistenza insidiosa e una sfida aggiuntiva per la sua individuazione e rimozione. In sintesi, l'analisi rivela un attacco sofisticato e stratificato che mina l'integrità del sistema operativo Windows, evidenziando la necessità di difese avanzate e una comprensione approfondita delle minacce informatiche.

Task Giorno 3

Riprendete l'analisi del codice, analizzando le routine tra le locazioni di memoria 00401080 e 00401128:

1. Qual è il valore del parametro "**ResourceName**" passato alla funzione **FindResourceA()**;
2. Il susseguirsi delle chiamate di funzione che effettua il Malware in questa sezione di codice l'abbiamo visto durante le lezioni teoriche. Che funzionalità sta implementando il Malware
3. È possibile identificare questa funzionalità utilizzando l'analisi statica basica?
4. In caso di risposta affermativa, elencare le evidenze a supporto.

Entrambe le funzionalità principali del Malware viste finora sono richiamate all'interno della funzione Main().

Disegnare un diagramma di flusso (inserite all'interno dei box solo le informazioni circa le funzionalità principale) che comprende le 3 funzioni.

Valore del parametro "ResourceName" passato alla funzione FindResourceA

```
.text:00401088 ; -----
.text:00401088
.text:00401088 loc_401088:      ; CODE XREF: sub_401080+2F1j
.text:00401088      mov     eax, lpType
.text:0040108D      push    eax           ; lpType
.text:0040108E      mov     ecx, lpName
.text:004010C4      push    ecx           ; lpName
.text:004010C5      mov     edx, [ebp+hModule]
.text:004010C8      push    edx           ; hModule
.text:004010C9      call    ds:FindResourceA
.text:004010CF      mov     [ebp+hResInfo], eax
.text:004010D2      cmp     [ebp+hResInfo], 0
.text:004010D6      jnz     short loc_4010DF
.text:004010D8      xor     eax, eax
.text:004010DA      jmp     loc_4011BF
.text:004010DF ; -----
```

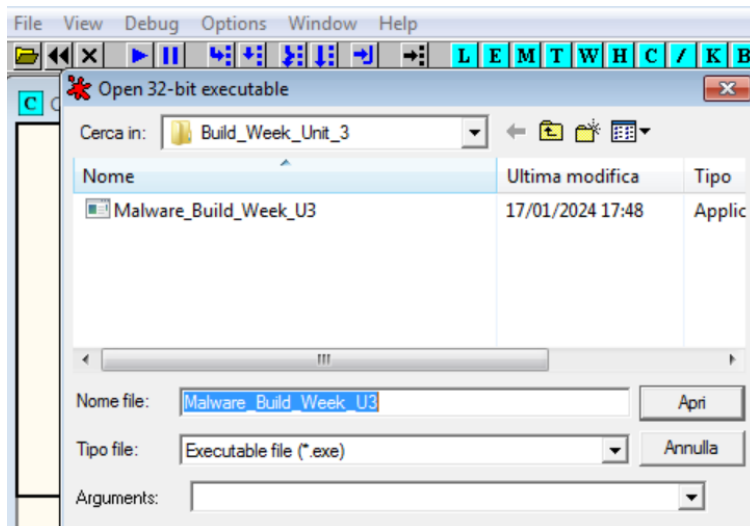
Nel contesto dell'analisi del malware, va sottolineata l'importanza cruciale del parametro "**ResourceName**" utilizzato nella chiamata alla funzione **FindResourceA**. Questo particolare parametro agisce come un identificatore per una risorsa incorporata all'interno del file eseguibile del malware. Nel caso specifico preso in esame, è stato individuato con il nome "TGAD". Tale informazione rivela un dettaglio fondamentale circa le risorse interne al malware,

apportando significativa comprensione nel suo funzionamento e nelle potenziali azioni malevoli che potrebbe intraprendere.

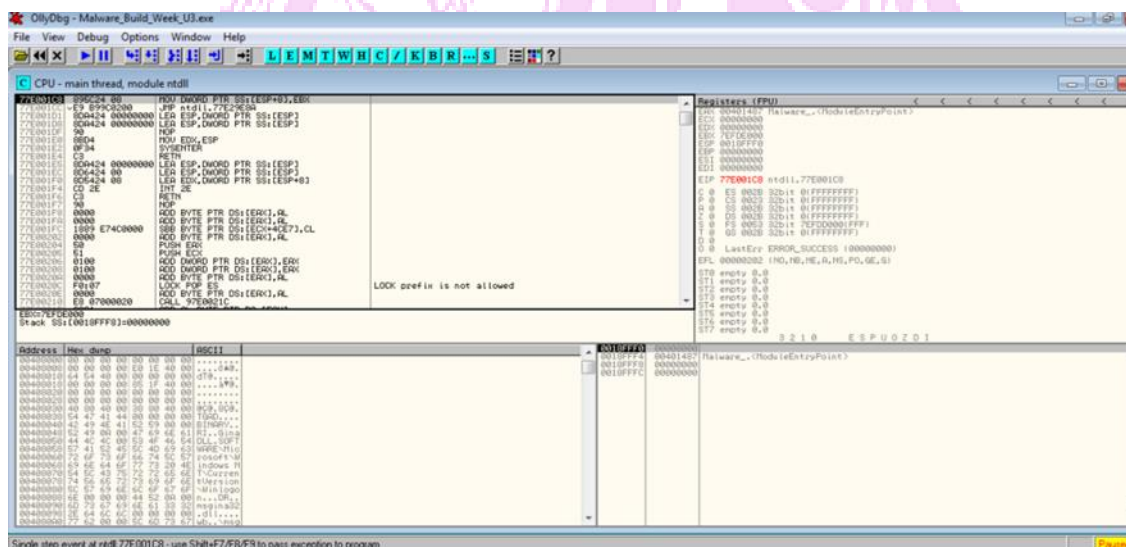
OllyDBG:

Come possiamo notare nell'immagine successiva, utilizzando OllyDBG avremo la conferma di ciò che abbiamo dedotto.

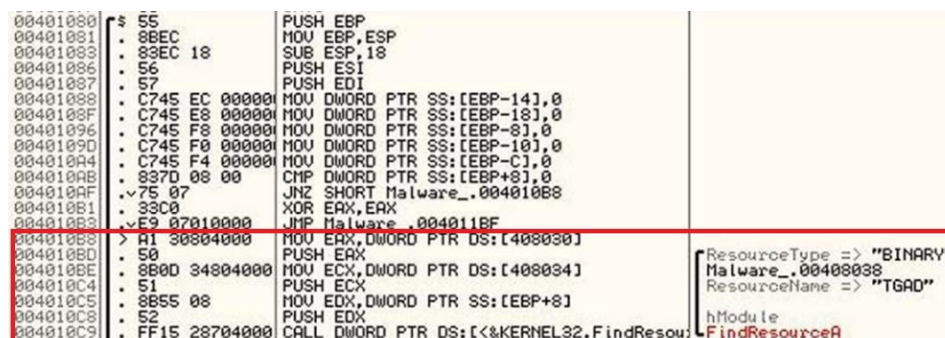
1. Avviare il tool e selezionare l'eseguibile malware da analizzare.



2. Schermata iniziale dopo apertura del file

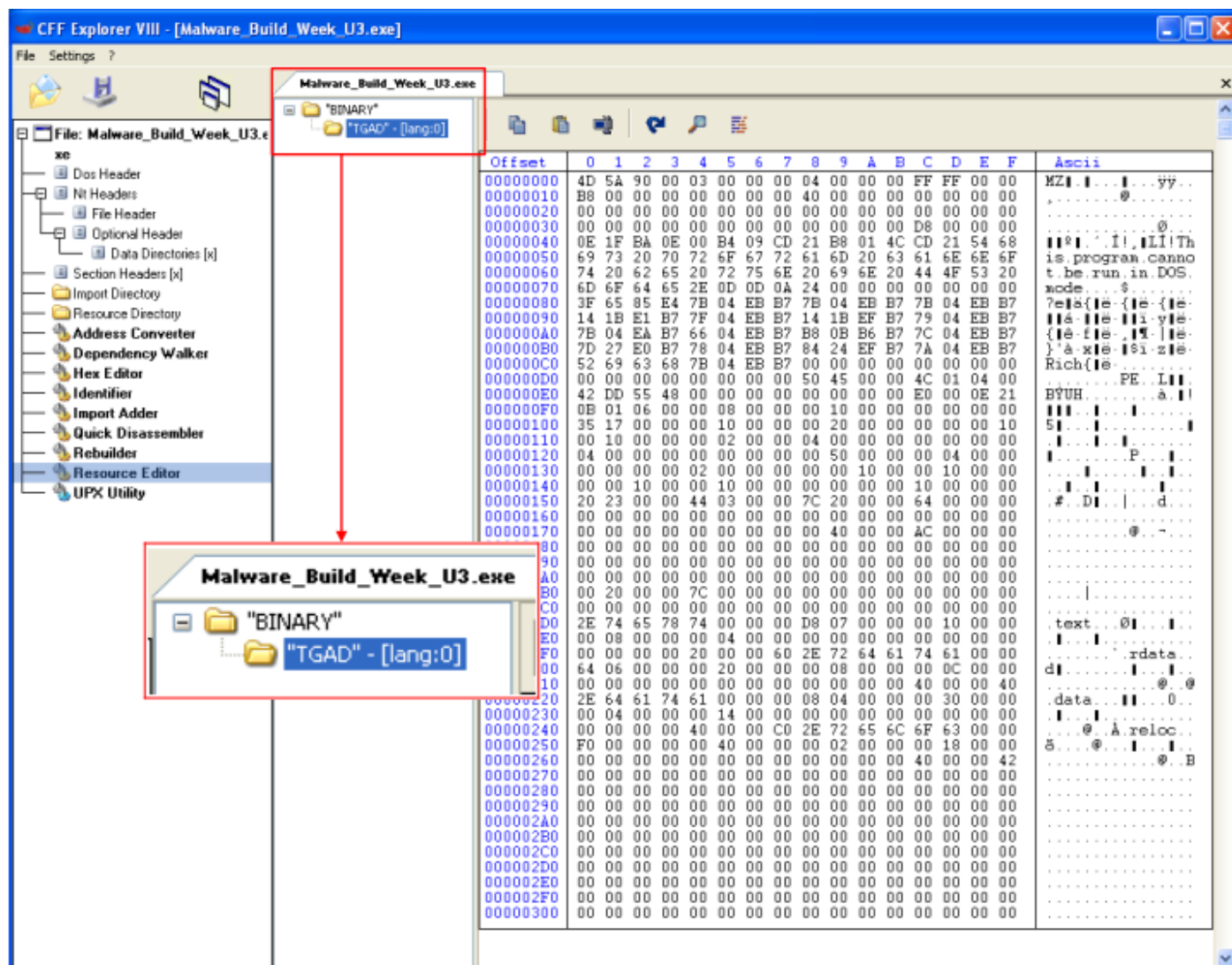


3. Individuiamo la funzione FindResourceA, ricercandola.



CFF Explorer:

Utilizzando CFF Explorer potremo inoltre controllare se l'eseguibile è presente nella cartella del Malware stesso, purtroppo non potremo reperire altre informazioni a riguardo



Funzionalità implementate dal Malware

Dall'analisi delle istruzioni presenti nelle locazioni di memoria indicate, emerge chiaramente che il malware sta seguendo una sequenza operativa tipica dei dropper. Tale sequenza include le seguenti chiamate alle API di Windows: FindResourceA, LoadResource e SizeOfResource.

La chiamata a FindResourceA è utilizzata per individuare una risorsa incorporata all'interno dell'eseguibile, identificata nel caso in questione con il nome "TGAD". Questo suggerisce che il malware sta cercando un componente specifico che è stato deliberatamente nascosto all'interno del proprio codice eseguibile, forse un secondo stadio di un attacco o un payload aggiuntivo.

Successivamente, la funzione LoadResource viene utilizzata per caricare il contenuto della risorsa identificata in memoria. Questo è un comportamento comune dei dropper, che spesso necessitano di caricare componenti dannosi in memoria per eseguirli o per ulteriori manipolazioni.

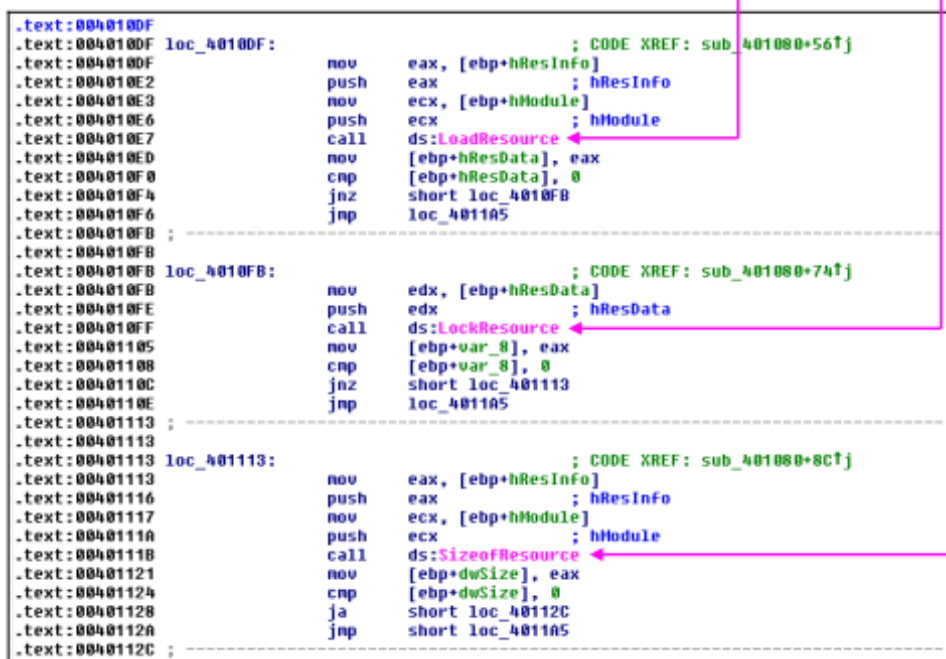
Infine, `SizeOfResource` viene invocata per assicurare che il componente malevolo sia stato caricato correttamente e per ottenere la dimensione della risorsa, necessaria per la corretta gestione della memoria durante l'estrazione e l'eventuale esecuzione del payload.

Le funzionalità che il malware sta implementando con questa sequenza di chiamate includono la ricerca, il caricamento e il dimensionamento di un componente occultato. Questa attività è comunemente utilizzata dai malware per eseguire codice dannoso in modo discreto, poiché la risorsa non viene eseguita direttamente dal disco, ma viene invece caricata e processata in memoria.

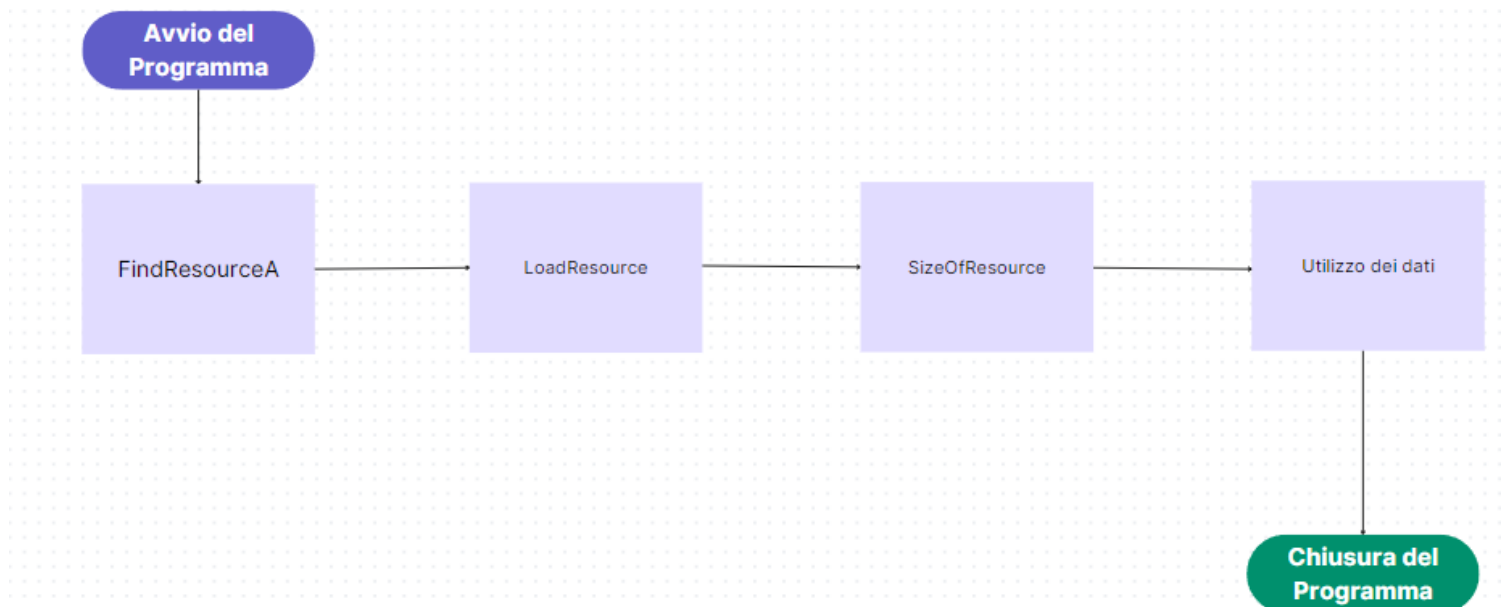
L'uso di risorse binarie nascoste rappresenta un metodo astuto per eludere la rilevazione basata su firma, in quanto il payload non appare come un file eseguibile separato sul disco rigido del sistema infetto. Inoltre, l'identificazione della risorsa attraverso un nome come "TGAD" potrebbe indicare l'impiego di tecniche di steganografia o compressione per nascondere ulteriormente la sua natura.

In sintesi, le chiamate sequenziali a `FindResourceA`, `LoadResource` e `SizeOfResource` indicano che il malware sta svolgendo le operazioni di un dropper, distribuendo componenti dannosi all'interno del proprio corpo eseguibile. Questo comportamento riflette tecniche avanzate di evasione e persistenza, consentendo al malware di rimanere attivo e latente nel sistema compromesso, eseguendo operazioni dannose senza essere facilmente individuato da strumenti di sicurezza tradizionali.

Andando ad analizzare qualche riga seguente all'indirizzo **00104080**, possiamo vedere come nella stessa routine siano invocate le funzioni: **LoadResource**, **LockResource** e **SizeofResource**



```
.text:004010DF
.text:004010DF loc_4010DF:                ; CODE XREF: sub_401080+56↑j
.text:004010DF      mov     eax, [ebp+hResInfo] ; hResInfo
.text:004010E2      push    eax
.text:004010E3      mov     ecx, [ebp+hModule] ; hModule
.text:004010E6      push    ecx
.text:004010E7      call   ds:LoadResource
.text:004010ED      mov     [ebp+hResData], eax
.text:004010F0      cmp     [ebp+hResData], 0
.text:004010F4      jnz     short loc_4010FB
.text:004010F6      jmp     loc_4011A5
.text:004010FB ;
.text:004010FB loc_4010FB:                ; CODE XREF: sub_401080+74↑j
.text:004010FB      mov     edx, [ebp+hResData] ; hResData
.text:004010FE      push    edx
.text:004010FF      call   ds:LockResource
.text:00401105      mov     [ebp+var_8], eax
.text:00401108      cmp     [ebp+var_8], 0
.text:0040110C      jnz     short loc_401113
.text:0040110E      jmp     loc_4011A5
.text:00401113 ;
.text:00401113 loc_401113:                ; CODE XREF: sub_401080+8C↑j
.text:00401113      mov     eax, [ebp+hResInfo] ; hResInfo
.text:00401116      push    eax
.text:00401117      mov     ecx, [ebp+hModule] ; hModule
.text:0040111A      push    ecx
.text:0040111B      call   ds:SizeOfResource
.text:00401121      mov     [ebp+dwSize], eax
.text:00401124      cmp     [ebp+dwSize], 0
.text:00401128      ja      short loc_40112C
.text:0040112A      jmp     short loc_4011A5
.text:0040112C ;
```

Per utilizzo dei dati, si intendono i dati della risorsa caricata che possono essere utilizzati per scopi diversi all'interno del programma.

Descrizione API di Windows presenti nel grafico a flusso

FindResourceA:

FindResourceA è una funzione utilizzata per trovare una risorsa all'interno di un modulo eseguibile o di una libreria a collegamento dinamico (DLL).

Questa funzione prende in input il manico (handle) del modulo (eseguibile o DLL) che contiene la risorsa, il nome della risorsa e il tipo della risorsa.

Restituisce un puntatore al descrittore della risorsa (HRSRC) se la risorsa viene trovata, altrimenti restituisce NULL.

LoadResource:

. Successivamente nel grafico troviamo la funzione LoadResource utilizzata per caricare una risorsa identificata dal suo descrittore (HRSRC) restituito da FindResourceA.

. Prende in input il modulo contenente la risorsa e il descrittore della risorsa.

. Restituisce un puntatore al dato (risorsa) caricato in memoria se il caricamento ha successo, altrimenti restituisce NULL.

SizeOfResource:

SizeOfResource è una funzione utilizzata per ottenere la dimensione, in byte, di una risorsa, prende in input il modulo contenente la risorsa e il descrittore della risorsa.

Restituisce la dimensione della risorsa se l'operazione ha successo, altrimenti restituisce zero.

È possibile identificare questa funzionalità utilizzando l'analisi statica basica? Nel caso elencare le evidenze a supporto.

Tramite l'analisi statica del malware in questione, emergono segni rivelatori che consentono agli esperti di individuarne le funzionalità principali:

1. **Chiamate API Sospette:** Le chiamate a funzioni API specifiche di Windows, come FindResourceA, LoadResource e SizeOfResource, quando si verificano in una sequenza ordinata, possono essere indicative di un comportamento sospetto. Queste API sono spesso impiegate in combinazione per manipolare risorse interne a un eseguibile.
2. **Nomi di Risorse Insoliti:** Il parametro "ResourceName" passato alla funzione FindResourceA è un identificatore di risorsa non convenzionale ("TGAD"), che non corrisponde a risorse comunemente associate ad applicazioni legittime. Questo potrebbe suggerire un utilizzo atipico o celato di tali risorse.
3. **Pattern di Comportamento:** Il pattern sequenziale delle chiamate API osservate segue una logica tipica di un dropper o downloader, finalizzato a individuare, caricare e determinare la dimensione di una risorsa nascosta per ulteriori azioni malevole.
4. **Sezioni di Risorse Atipiche:** L'esame della sezione.rsrc del file eseguibile, contenente le risorse incorporate, può rivelare anomalie o risorse occultate non solitamente presenti in software legittimo.
5. **Flusso di Controllo:** Il flusso di controllo che segue le chiamate API può indicare la preparazione del terreno per eseguire o decodificare il payload nascosto dopo il suo caricamento in memoria.
6. **Stringhe e Costanti:** Le stringhe e le costanti presenti nel codice, come nomi di risorse o chiavi di registro specifiche, possono fornire indizi sulle intenzioni del malware. L'analisi statica di tali elementi può rivelare correlazioni con database di minacce note, identificando comportamenti o tecniche di attacco già noti.
7. **Metadati e Simboli:** I metadati e i simboli all'interno dell'eseguibile possono rivelare informazioni su strumenti di sviluppo utilizzati, autori o gruppi di minacce, e versioni di librerie correlate a vulnerabilità note.
8. **Firme di Malware:** Confrontando il codice con firme di malware conosciute, è possibile identificare eventuali corrispondenze. Sebbene la risorsa in questione sia nascosta e possa eludere la rilevazione basata sulle firme, le tecniche utilizzate per accedervi potrebbero essere già state osservate in precedenza e quindi essere note.
9. **Decompilazione del Codice:** Utilizzando decompilatori, è possibile trasformare il codice macchina in un linguaggio di alto livello, semplificando l'analisi della logica del malware.

L'analisi statica è fondamentale per comprendere le funzionalità e le minacce potenziali di un malware senza doverlo eseguire in un ambiente live. Questo metodo si basa sull'ispezione del codice sorgente o, più comunemente nel caso dei malware, del codice eseguibile disassemblato o de-compilato.

Conclusioni:

Dalla descrizione delle funzionalità principali del Malware, possiamo trarre le seguenti conclusioni:

Valore del parametro "ResourceName" passato alla funzione FindResourceA: Il parametro "ResourceName" è identificato come "TGAD", suggerendo un particolare tipo di risorsa incorporata all'interno dell'eseguibile del malware. Questo fornisce informazioni cruciali sulle risorse interne utilizzate dal malware e potrebbe indicare l'esistenza di componenti dannosi nascosti.

Funzionalità implementate dal Malware: Il malware segue una sequenza operativa tipica dei dropper, coinvolgendo le chiamate alle API di Windows FindResourceA, LoadResource e SizeOfResource. Queste chiamate indicano un comportamento sospetto mirato alla ricerca, al caricamento e alla gestione di risorse nascoste all'interno dell'eseguibile. Questo suggerisce che il malware stia distribuendo componenti dannosi all'interno del proprio corpo eseguibile, con l'intento di eseguire codice malevolo in modo discreto e persistente nel sistema compromesso.

Identificazione della funzionalità tramite analisi statica basica: L'analisi statica del malware fornisce diverse evidenze a supporto dell'identificazione delle sue funzionalità principali. Queste includono chiamate API sospette, nomi di risorse insoliti, pattern di comportamento tipici dei dropper, sezioni di risorse atipiche, flusso di controllo delle chiamate API, stringhe e costanti rivelatori, metadati e simboli nel codice, firme di malware e potenziale compilazione del codice.

Task Giorno 4

Preparare l'ambiente ed i tools per l'esecuzione del Malware (suggerimento: avviate principalmente Process Monitor ed assicurate di eliminare ogni filtro cliccando sul "reset" quando richiesto in fase di avvio). Eseguite il Malware, facendo doppio click sull'icona dell'eseguibile.

Cosa notate all'interno della cartella dove è situato l'eseguibile del Malware? Spiegate cosa è avvenuto, unendo le evidenze che avete raccolto finora per rispondere alla domanda

Analizzare ora i risultati di Process Monitor (consiglio: utilizzare il filtro come in figura sotto per estrarre solo le modifiche apportate al sistema da parte del Malware). Fate click su "ADD" poi su "Apply" come abbiamo visto nella lezione teorica.

Tool utilizzati

Process Monitor: Si tratta di uno strumento, approfondito precedentemente, che ci permette di analizzare i processi in esecuzione sul pc, nel nostro caso il malware.

Regedit: È lo strumento che verrà usato per verificare cosa viene cambiato da parte del malware nel registro.

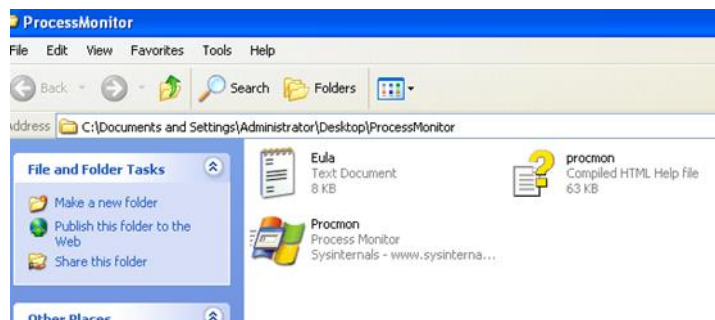
Analisi dinamica con Process Monitor

Per avviare un'analisi con process monitor dovremo eseguire pochi step:

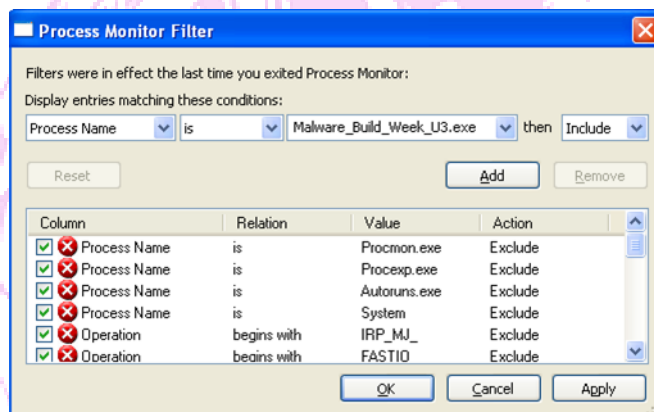
1. Apriamo la cartella sul Desktop "Process Monitor" la quale contiene l'eseguibile per avviare il programma.



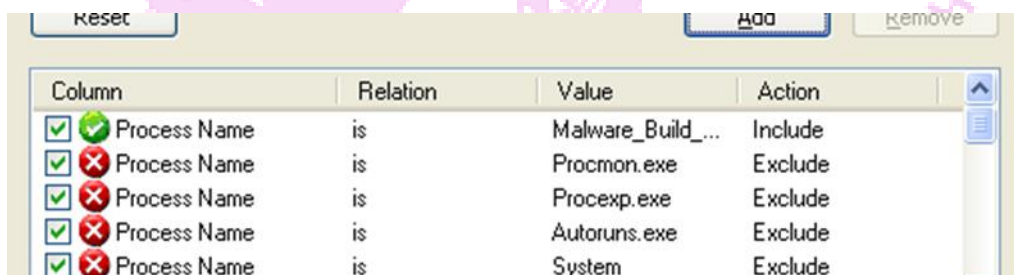
2. Clicchiamo sull'icona denominata "Procmon" in quanto eseguibile dell'applicazione.



3. Una volta che il programma viene avviato ci troveremo davanti alla schermata di seguito, la quale ci chiederà se vogliamo inserire un filtro. Per farlo selezioniamo nel riquadro più a sinistra "Process Name" questo permetterà al programma di filtrare i processi attraverso il nome dello stesso. Il nome, invece, verrà inserito nel penultimo riquadro a destra, nel nostro caso inseriremo "Malware_Build_Week_U3.exe".



4. Premiamo su "add" e poi "apply" così da aggiungere il filtro ed evitare di incappare in altri processi oltre a quello di nostro interesse.



Una volta che l'applicazione è stata avviata ed abbiamo impostato il filtro, possiamo iniziare la cattura dei processi del malware, questo avverrà automaticamente una volta cliccato sull'eseguibile dello stesso tenendo Process Monitor aperto. Ci ritroveremo davanti ad una schermata simile a questa:

Time of Day	Process Name	PID	Operation	Path	Result	Detail
11:52:03.6460	Malware_Build_Week_U3...	340	Process Start		SUCCESS	Parent PID: 1504, Command line: "C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\Malware_Build_Week_U3.exe"
11:52:03.6460	Malware_Build_Week_U3...	340	Thread Create		SUCCESS	Thread ID: 852
11:52:03.6468	Malware_Build_Week_U3...	340	QueryNameInformationFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\Malware_Build_Week_U3.exe	SUCCESS	Name: \Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\Malware_Build_Week_U3.exe
11:52:03.6469	Malware_Build_Week_U3...	340	Load Image	C:\WINDOWS\system32\ntdll.dll	SUCCESS	Image Base: 0x000000, Image Size: 0x0000
11:52:03.6470	Malware_Build_Week_U3...	340	QueryNameInformationFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\Malware_Build_Week_U3.exe	SUCCESS	Name: \Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\Malware_Build_Week_U3.exe
11:52:03.6471	Malware_Build_Week_U3...	340	CreateFile	C:\WINDOWS\system32\MALWARE_BUILD_WEEK_U3\EXE-0E171D0F.pl	SUCCESS	Desired Access: Generic Read, Disposition: Open, Options: Synchronous IO Non-Alert, Attributes: n/a, ShareMode: None, AllocationSize: n
11:52:03.6486	Malware_Build_Week_U3...	340	QueryStandardInformationFile	C:\WINDOWS\system32\MALWARE_BUILD_WEEK_U3\EXE-0E171D0F.pl	SUCCESS	AllocationSize: 12,288, EndOfFile: 10,856, NumberLinks: 1, DeletePending: False, Directory: False
11:52:03.6487	Malware_Build_Week_U3...	340	ReadFile	C:\WINDOWS\system32\MALWARE_BUILD_WEEK_U3\EXE-0E171D0F.pl	SUCCESS	Offset: 0, Length: 10,856
11:52:03.6491	Malware_Build_Week_U3...	340	CloseFile	C:\WINDOWS\system32\MALWARE_BUILD_WEEK_U3\EXE-0E171D0F.pl	SUCCESS	Offset: 0, Length: 10,856, I/O Flag: Non-cached, Paging I/O, Synchronous Paging I/O
11:52:03.6494	Malware_Build_Week_U3...	340	CreateFile	C:\	SUCCESS	Desired Access: Read Attributes, Write Attributes, Synchronize, Disposition: Open, Options: Synchronous IO Non-Alert, Attributes: n/a, Share
11:52:03.6494	Malware_Build_Week_U3...	340	QueryInformationVolume	C:\	SUCCESS	VolumeCreationTime: 3/20/2017 9:34:16 PM, VolumeSerialNumber: D8BA-8021, SupportsObjects: True, VolumeLabel
11:52:03.6502	Malware_Build_Week_U3...	340	FileSystemControl	C:\	SUCCESS	Control: FSCTL_FILE_PREFETCH
11:52:03.6502	Malware_Build_Week_U3...	340	CreateFile	C:\	SUCCESS	Desired Access: Read Data/List Directory, Synchronize, Disposition: Open, Options: Directory, Synchronous IO Non-Alert, Open For Backup
11:52:03.6505	Malware_Build_Week_U3...	340	QueryDirectory	C:\	SUCCESS	0, 1, ..., FileInformationClass: FileNamesInformation, 3 All Users, 4 Default User, 5 LocalService, 6 NetworkService
11:52:03.6505	Malware_Build_Week_U3...	340	CloseFile	C:\	NO MORE FILES	
11:52:03.6506	Malware_Build_Week_U3...	340	CreateFile	C:\DOCUMENTS AND SETTINGS	SUCCESS	Desired Access: Read Data/List Directory, Synchronize, Disposition: Open, Options: Directory, Synchronous IO Non-Alert, Open For Backup
11:52:03.6510	Malware_Build_Week_U3...	340	QueryDirectory	C:\Documents and Settings	SUCCESS	0, 1, ..., FileInformationClass: FileNamesInformation, 3 Cookies, 4 Desktop, 5 Favorites, 6 Local Settings, 7 My Documents, 8 NetHood,
11:52:03.6511	Malware_Build_Week_U3...	340	CloseFile	C:\Documents and Settings	SUCCESS	
11:52:03.6514	Malware_Build_Week_U3...	340	CreateFile	C:\Documents and Settings\ADMINISTRATOR	SUCCESS	Desired Access: Read Data/List Directory, Synchronize, Disposition: Open, Options: Directory, Synchronous IO Non-Alert, Open For Backup
11:52:03.6515	Malware_Build_Week_U3...	340	QueryDirectory	C:\Documents and Settings\Administrator	SUCCESS	0, 1, ..., FileInformationClass: FileNamesInformation, 3 Cookies, 4 Desktop, 5 Favorites, 6 Local Settings, 7 My Documents, 8 NetHood,
11:52:03.6515	Malware_Build_Week_U3...	340	QueryDirectory	C:\Documents and Settings\Administrator	NO MORE FILES	
11:52:03.6516	Malware_Build_Week_U3...	340	CreateFile	C:\Documents and Settings\Administrator\Desktop	SUCCESS	Desired Access: Read Data/List Directory, Synchronize, Disposition: Open, Options: Directory, Synchronous IO Non-Alert, Open For Backup
11:52:03.6516	Malware_Build_Week_U3...	340	QueryDirectory	C:\Documents and Settings\Administrator\Desktop	SUCCESS	0, 1, ..., FileInformationClass: FileNamesInformation, 3 CFF Explorer Ink, 4 Command Prompt Ink, 5 Esercizio_Pratico_U2_W2_L1, 6 Eser
11:52:03.6517	Malware_Build_Week_U3...	340	QueryDirectory	C:\Documents and Settings\Administrator\Desktop	NO MORE FILES	
11:52:03.6517	Malware_Build_Week_U3...	340	CloseFile	C:\Documents and Settings\Administrator\Desktop	SUCCESS	
11:52:03.6518	Malware_Build_Week_U3...	340	CreateFile	C:\Documents and Settings\Administrator\Desktop\BUILD_WEEK_UNIT_3	SUCCESS	Desired Access: Read Data/List Directory, Synchronize, Disposition: Open, Options: Directory, Synchronous IO Non-Alert, Open For Backup
11:52:03.6519	Malware_Build_Week_U3...	340	QueryDirectory	C:\Documents and Settings\Administrator\Desktop\BUILD_WEEK_UNIT_3	SUCCESS	0, 1, ..., FileInformationClass: FileNamesInformation
11:52:03.6519	Malware_Build_Week_U3...	340	QueryDirectory	C:\Documents and Settings\Administrator\Desktop\BUILD_WEEK_UNIT_3	NO MORE FILES	
11:52:03.6520	Malware_Build_Week_U3...	340	CloseFile	C:\Documents and Settings\Administrator\Desktop\BUILD_WEEK_UNIT_3	SUCCESS	
11:52:03.6520	Malware_Build_Week_U3...	340	CreateFile	C:\WINDOWS	SUCCESS	Desired Access: Read Data/List Directory, Synchronize, Disposition: Open, Options: Directory, Synchronous IO Non-Alert, Open For Backup
11:52:03.6521	Malware_Build_Week_U3...	340	QueryDirectory	C:\WINDOWS	SUCCESS	0, 1, ..., FileInformationClass: FileNamesInformation, 3 0 log, 4 addn, 5 AppPatch, 6 assembly, 7 Blue Lace 16.bmp, 8 bootstat.dat, 9
11:52:03.6522	Malware_Build_Week_U3...	340	QueryDirectory	C:\WINDOWS	NO MORE FILES	
11:52:03.6522	Malware_Build_Week_U3...	340	CloseFile	C:\WINDOWS	SUCCESS	

Andremo dunque a ricercare, scorrendo le informazioni a noi utili ovvero, la creazione di una chiave di registro (**RegCreateKey**) e del nome che le viene dato, come possiamo vedere nell'immagine sottostante.

10:36:24.9471	Malware_Build_Week_U3...	1820	CloseFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\msgina32.dll	SUCCESS	
10:36:24.9474	Malware_Build_Week_U3...	1820	RegCreateKey	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon	SUCCESS	Desired Access: All Access
10:36:24.9474	Malware_Build_Week_U3...	1820	RegSetValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\GinaDLL	SUCCESS	Type: REG_SZ, Length: 520, Data: C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\msgina32.dll
10:36:24.9478	Malware_Build_Week_U3...	1820	Thread Exit		SUCCESS	
10:36:24.9479	Malware_Build_Week_U3...	1820	Process Exit		SUCCESS	Thread ID: 1492, User Time: 0.0000000, Kernel Time: 0.0000000
10:36:24.9479	Malware_Build_Week_U3...	1820	CloseFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3	SUCCESS	Exit Status: 0, User Time: 0.0156250 seconds, Kernel Time: 0.0000000 seconds, Private Bytes: 212,992, Peak Private Bytes:

Possiamo notare come il malware crei una chiave di registro usando l'operazione "RegCreateKey" al percorso HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\winlogon.

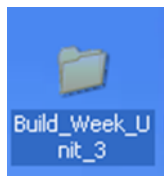
Mentre con l'operazione seguente, ovvero, "RegSetValue", viene dato il nome alla suddetta chiave di registro di GinaDLL.

Winlogon: È un processo fondamentale nei sistemi Windows, in quanto, gestisce il processo di accesso e logout degli utenti. In caso di login lo fa autenticando le credenziali dell'utente ed iniziando i processi del desktop, mentre, per quanto riguarda il logout, questo processo eseguibile si occupa della chiusura di tutti i processi e del salvataggio delle informazioni di sessione. Ricontrare problemi con l'eseguibile potrebbe portare a difficoltà di accesso e gestione delle sessioni.

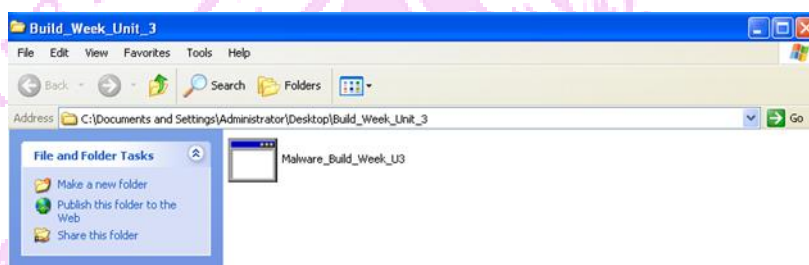
GinaDLL: tentativo del malware di camuffarsi come un processo noto a windows, ovvero, "msgina" abbreviazione di "Microsoft Graphical Identification and Authentication", questa libreria dinamica è responsabile dell'interfaccia grafica utilizzata per l'identificazione e l'autenticazione degli utenti durante il processo di accesso al sistema. È un componente cruciale per il processo di accesso e sicurezza del sistema, qualsiasi problema con questa libreria può influenzare l'efficienza e la sicurezza del sistema.

Cosa avviene all'interno della cartella dove è situato il Malware?

Per iniziare l'analisi del malware a noi fornito andiamo ad aprire la cartella presente sul desktop chiamata "Build_Week_Unit_3".

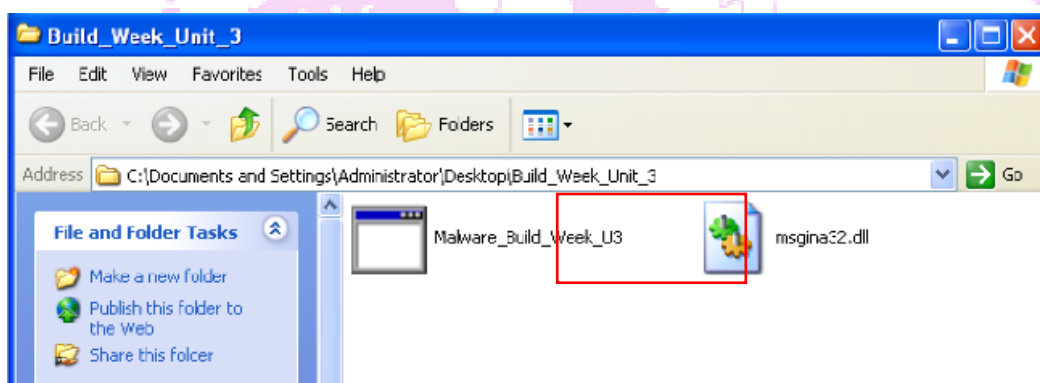


Una volta fatto doppio click su di essa ci si aprirà la cartella con all'interno l'eseguibile contenente il malware, per avviarlo eseguiamo un doppio click su di esso.



Come possiamo notare nella cartella dove è situato l'eseguibile del Malware viene creato un nuovo file, ovvero **msgina.dll**.

Fatto ciò si potrà notare apparire il file "msgina32.dll" creato dal malware.



Cosa avviene quando il malware viene avviato?

Una volta avviato l'eseguibile, il malware cerca e estrae una DLL contenente le vere istruzioni e funzioni necessarie per portare a termine il suo intento malevolo. Attraverso un processo di "injecting", viene inserita una libreria malevola tra "winlogon.exe" e "msgina32.dll", al fine di intercettare le credenziali di accesso degli utenti.

Modifiche apportate dal malware al sistema:

1. **Creazione di File Non Autorizzati:** All'avvio del processo malevolo, viene rilevata la generazione di un file denominato "msgina32.dll" all'interno della directory dell'applicazione. Questa DLL, estranea alle librerie standard di Windows, denota un'azione malevola. L'impiego del nome "msgina32.dll" suggerisce un tentativo di mimetizzarsi come un componente legittimo di sistema, utilizzando una denominazione familiare per mascherare le sue intenzioni. Questo comportamento è tipico dei dropper, i quali estraggono elementi dannosi dal loro corpo principale per perpetrare azioni malevole all'interno del sistema ospite.
2. **Manipolazione del Registro di Sistema:** L'analisi tramite Process Monitor rileva modifiche significative alle chiavi di registro. L'inserimento del valore "GinaDLL" nella chiave HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon suscita particolare preoccupazione. Tale modifica potrebbe indicare un tentativo di compromettere o manipolare il processo di autenticazione di Windows, un attacco che potrebbe causare gravi problemi di sicurezza, come il furto di credenziali.
3. **Interazione con il File System:** Le chiamate alle funzioni CreateFile() e WriteFile() evidenziano un'attiva manipolazione dei file da parte del malware. Queste azioni suggeriscono l'intento di stabilire la persistenza del malware nel sistema, un passaggio fondamentale per garantire operazioni prolungate e resistere ai tentativi di rimozione.

Ipotesi sul funzionamento del malware

Analizzando sia l'aspetto statico che dinamico del malware, possiamo delineare il suo funzionamento:

1. **Fase di Estrazione o Decompressione:** Il malware avvia un'operazione di estrazione o decompressione per un payload nascosto, identificato come "msgina32.dll", nella stessa directory dell'eseguibile. Questo suggerisce l'azione di un dropper, preparando il terreno per ulteriori attività malevole.
2. **Manipolazione dell'Autenticazione di Windows:** Attraverso la modifica del registro di sistema, il malware cerca di indirizzare Windows a caricare una DLL malevola ("msgina32.dll") durante il processo di autenticazione. Questo potrebbe consentire al malware di rubare credenziali o eseguire altre operazioni dannose durante il processo di login.
3. **Persistenza:** Creando una nuova chiave di registro e associandovi un valore, il malware garantisce che il suo payload venga eseguito ad ogni avvio del processo di login di Windows, garantendo la persistenza nel sistema.
4. **Esecuzione di Codice Dannoso:** Con il valore "GinaDLL" modificato, ogni tentativo di accesso degli utenti avvia il caricamento della DLL malevola invece della libreria legittima, permettendo al malware di eseguire azioni dannose senza il consenso dell'utente.
5. **Attività Post-Login:** Dopo l'accesso, il malware potrebbe eseguire ulteriori attività dannose, sfruttando il contesto di autenticazione per mascherare la sua presenza e le sue azioni.
6. **Obiettivi a Lungo Termine:** Il malware potrebbe avere vari obiettivi a lungo termine, come la raccolta di informazioni sensibili, la creazione di una botnet, il criptaggio dei file per il ransomware o l'utilizzo del sistema compromesso per ulteriori attacchi.
7. **Strategie di Mitigazione:** Per contrastare il malware, è cruciale utilizzare software antivirus aggiornato, applicare regolarmente patch di sicurezza e adottare una politica di minimo privilegio per gli account utente.
8. **Risposta agli Incidenti:** Nel caso di compromissione, è fondamentale avere un piano di risposta agli incidenti per contenere la diffusione del malware, recuperare i dati e ripristinare le operazioni normali del sistema.

Analizzando tutti i dati forniti e vedendo il comportamento del malware possiamo ammettere che si tratti di un dropper

In informatica, il termine "**dropper**" si riferisce a un tipo di malware o software dannoso progettato per scaricare e installare ulteriori componenti dannosi sul sistema colpito, ovvero, agisce come un veicolo per consegnare e eseguire altri malware sul computer dell'utente senza il loro consenso.

Il nome "dropper" deriva dalla sua funzione principale: rilasciare o, appunto, "droppare" il payload dannoso sul sistema bersaglio. Questo payload può includere virus, worm, trojan o altri tipi di malware, a seconda degli obiettivi dell'attaccante.

I dropper sono spesso progettati per eludere le misure di sicurezza e diffondersi in modo silenzioso, sfruttando vulnerabilità nel sistema o ingannando l'utente attraverso tecniche di social engineering.

Regedit

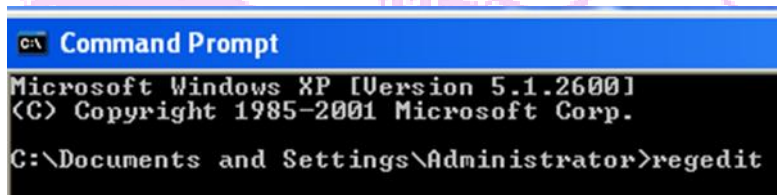
Per confermare ulteriormente quanto affermato, è possibile accedere al "Registro di sistema" di Windows utilizzando l'applicazione Regedit. Dopo aver aperto Regedit, navigando nella cartella Winlogon, sarà evidente che prima dell'esecuzione del malware non sono presenti né il file "Ginadll" né una chiave ad esso associata. Tuttavia, dopo l'esecuzione del malware, sarà possibile individuare la chiave creata dal malware stesso, contenente il percorso del file associato, confermando così l'attività dannosa del malware.

Accedervi è semplicissimo:

1. Avviamo Command Prompt presente sul desktop o premendo sul pulsante start e poi selezionando l'icona sottostante.



2. Inseriamo il comando "regedit".



3. Infine premiamo invio, una volta fatto saremo all'interno del programma.

Per verificare cosa viene cambiato dal malware avremo una foto che mostra le chiavi di registro presenti in winlogon (cartella in cui viene inserita la chiave di registro da parte del malware) prima e dopo l'esecuzione del malware

PRIMA



related.desc	DefaultUserName	REG_SZ	Administrator
SeCEdit	ForceUnlockLogon	REG_DWORD	0x00000000 (0)
Setup	HibernationPreviouslyEnabled	REG_DWORD	0x00000001 (1)
SvcHost	LegalNoticeCaption	REG_SZ	
SystemRestore	LegalNoticeText	REG_SZ	
Terminal Server	LogonType	REG_DWORD	0x00000001 (1)
Time Zones	PasswordExpiryWarning	REG_DWORD	0x0000000e (14)
Tracing	PowerdownAfterShutdown	REG_SZ	0
Type 1 Installer	ReportBootOk	REG_SZ	1
Userinstallable.drivers	ScreenMoveOption	REG_SZ	0
Windows	SFCDisable	REG_DWORD	0x00000000 (0)
Winlogon	SfcQuota	REG_DWORD	0xffffffff (4294967295)
Credentials	Shell	REG_SZ	Explorer.exe
GPEExtensions	ShowLogonOptions	REG_DWORD	0x00000000 (0)
Notify	ShutdownWithoutLogon	REG_SZ	0
SpecialAccounts			

DOPO



Tracing	DefaultUserName	REG_SZ	Administrator
Type 1 Installer	ForceUnlockLogon	REG_DWORD	0x00000000 (0)
Userinstallable.drivers	GinaDLL	REG_SZ	C:\Documents and Settings\Administrator\Desktop\Bull...
Windows	HibernationPreviouslyEnabled	REG_DWORD	0x00000001 (1)
Winlogon	Key	REG_BINARY	98 33 07 01
Credentials			
GPEExtensions			

Possiamo vedere come le voci GinaDLL (nome dato dal malware alla chiave di registro creata) e Key prima assenti siano ora invece presenti nel sistema, camuffandosi come una qualsiasi altra chiave di registro.

Conclusioni

Questo tipo di software malevolo ha molti metodi di diffusione, tra i quali abbiamo, email di phishing, link per il download compromessi, usb infette e molti altri. Un utente poco preparato potrebbe cadere in trappola e inavvertitamente scaricarlo, ciò provocherebbe seri danni all'azienda se non preso per tempo in quanto potrebbe infettare gran parte della rete ed ottenere le credenziali di diversi utenti. Per evitare o limitare che ciò avvenga è consigliabile:

1. Educare gli utenti su pratiche di sicurezza informatica, riconoscendo e evitando minacce come e-mail di phishing.
2. Mantenere sempre aggiornato il sistema operativo e le applicazioni, poiché i dropper spesso sfruttano vulnerabilità note per diffondersi.
3. Limitare l'accesso ed il permesso di esecuzione da dispositivi rimovibili per prevenire la diffusione attraverso supporti infetti.
4. Implementare politiche di sicurezza aziendale che limitino l'installazione di software non autorizzati e promuovano pratiche di sicurezza informatica.
5. Mantenere backup regolari dei dati critici per ridurre l'impatto in caso di attacco, consentendo un rapido ripristino in caso di compromissione.

Task Giorno 5

“GINA (Graphical identification & authentication) è un componente lecito di Windows che permette l’autenticazione degli utenti tramite interfaccia grafica – ovvero permette agli utenti di inserire **username** e **password** nel classico riquadro Windows, come quello in figura a destra che usate anche voi per accedere alla macchina virtuale.

Cosa può succedere se il file .dll lecito viene sostituito con un file .dll malevolo, che intercetta i dati inseriti?”

Svolgimento e spiegazione del Task Giorno 5:

Sostituzione del File .dll

La sostituzione di un file .dll legittimo relativo al componente GINA di Windows con una variante malevola introduce significative vulnerabilità di sicurezza che possono essere sfruttate in diversi modi per compromettere sia l'integrità del sistema che la privacy degli utenti. Di seguito, viene approfondita e dettagliata ulteriormente l'analisi dei rischi associati/amplificati.

Rischi di Sicurezza Associati/Amplificati:

Intercezione delle Credenziali:

1. Sostituzione di un .dll malevolo:

- 1.1. Questo tipo di attacco consente agli aggressori di sostituire un file di libreria dinamica (.dll) legittimo con una versione dannosa. Una volta sostituito, il file malevolo può intercettare e catturare le credenziali degli utenti durante le operazioni di autenticazione, mettendo a repentaglio la sicurezza e la privacy delle informazioni sensibili.

2. Esecuzione di Codice Arbitrario:

- 2.1. La presenza di un .dll dannoso apre la porta all'esecuzione di codice malevolo sul sistema. Ciò significa che gli attaccanti possono introdurre e eseguire programmi dannosi sul computer, aprendo la strada a attacchi sofisticati, inclusi quelli che consentono il controllo remoto del sistema da parte degli aggressori.

3. Elevazione dei Privilegi:

- 3.1. Gli aggressori possono sfruttare il .dll malevolo per ottenere privilegi elevati. Questo significa che possono acquisire un accesso avanzato e prendere il controllo completo del sistema, compromettendo la sicurezza complessiva. Un attacco di elevazione dei privilegi può consentire agli aggressori di superare le restrizioni di sicurezza imposte dal sistema operativo.

4. Movimenti Lateral:

- 4.1. Una volta che un sistema è stato compromesso, gli aggressori possono utilizzarlo come trampolino di lancio per accedere ad altre risorse di rete. Questo movimento laterale consente loro di estendere l'entità dell'attacco, esplorando e compromettendo ulteriori sistemi all'interno dell'ambiente di rete.

5. Manipolazione dell'Autenticazione:

- 5.1. La sostituzione del .dll offre agli aggressori la possibilità di manipolare il processo di autenticazione. Questo potrebbe tradursi nella creazione di account nascosti, nella modifica delle politiche di sicurezza o nell'alterazione dei meccanismi di controllo degli accessi, aprendo la strada a ulteriori azioni malevole.

6. Fuga di Dati:

- 6.1. La compromissione dei file .dll può portare alla fuga di dati sensibili. Gli aggressori possono accedere, modificare o esfiltrare informazioni sensibili, causando danni finanziari e danneggiando l'immagine e la reputazione dell'organizzazione colpita. La fuga di dati è spesso associata a violazioni della privacy e a possibili conseguenze legali.

Conseguenze Aggiuntive:

1. Implicazioni Legal e Normative:

- 1.1. Le organizzazioni sono soggette a sanzioni legali (ad esempio, multe) e finanziarie in caso di violazione dei regolamenti sulla protezione dei dati.
- 1.2. Violare normative come il GDPR o altre leggi sulla privacy può comportare conseguenze legali significative, mettendo a rischio la reputazione e la sostenibilità finanziaria dell'organizzazione.
- 1.3. Le implicazioni legali possono variare a seconda della natura e della gravità della violazione, includendo azioni legali da parte degli individui interessati.

2. Costi di Risposta agli Incidenti:

- 2.1. Le spese per rispondere agli incidenti possono essere considerevoli e comprendono costi diretti e indiretti.
- 2.2. I costi diretti potrebbero includere l'assunzione di esperti forensi per indagare sull'incidente, il ripristino dei sistemi compromessi e la notifica degli individui colpiti.
- 2.3. I costi indiretti possono derivare dalla perdita di produttività, danni alla reputazione e perdita di clienti, contribuendo a un impatto finanziario a lungo termine.
- 2.4. Investire in misure preventive e in un piano di risposta agli incidenti può aiutare a mitigare i costi complessivi e a preparare l'organizzazione per affrontare situazioni di emergenza.

Misure di Mitigazione e Difesa:

1. Implementazione di Sistemi di Rilevamento e Prevenzione delle Intrusioni (IDS e IPS):

- 1.1. Adozione di soluzioni avanzate per individuare e prevenire intrusioni nel sistema.
- 1.2. Monitoraggio costante delle attività anomale per identificare potenziali minacce in tempo reale.
- 1.3. Configurazione di allarmi e notifiche per rispondere prontamente a eventi di sicurezza.

2. Hardening del Sistema:

- 2.1. Aggiornamento regolare del software e applicazione di patch di sicurezza per colmare le vulnerabilità.
- 2.2. Implementazione di procedure di hardening per configurare il sistema in modo sicuro.
- 2.3. Monitoraggio costante delle nuove minacce e aggiornamento delle politiche di sicurezza di conseguenza.

3. Autenticazione Multifattore (MFA):

- 3.1. Utilizzo di metodi di autenticazione aggiuntivi oltre alle password tradizionali.
- 3.2. Incremento della sicurezza richiedendo più di un elemento di verifica per l'accesso.
- 3.3. Protezione aggiuntiva contro accessi non autorizzati anche in caso di compromissione delle credenziali.

4. Formazione e Consapevolezza:

- 4.1. Sensibilizzazione degli utenti sui rischi di sicurezza e sulle pratiche migliori.
- 4.2. Offerta di formazione regolare per riconoscere e evitare minacce come phishing e malware.
- 4.3. Coinvolgimento degli utenti nel mantenere un ambiente informatico sicuro.

5. Gestione delle Patch e delle Vulnerabilità:

- 5.1. Implementazione di un processo strutturato per gestire l'applicazione di patch.
- 5.2. Valutazione regolare delle vulnerabilità per identificare e mitigare potenziali rischi.
- 5.3. Riduzione delle esposizioni attraverso una gestione proattiva delle patch.

6. Backup e Ripristino:

- 6.1. Mantenimento di backup regolari dei dati critici.
- 6.2. Test periodici dei piani di ripristino per garantire un recupero rapido in caso di perdita o danneggiamento dei dati.
- 6.3. Considerazione di soluzioni di backup off-site per proteggere i dati in caso di disastro.

7. Risposta agli Incidenti e Piano di Recupero:

- 7.1. Sviluppo di un piano dettagliato per gestire gli incidenti di sicurezza.
- 7.2. Identificazione di ruoli e responsabilità durante una risposta agli incidenti.
- 7.3. Valutazione e miglioramento continuo del piano in base all'evoluzione delle minacce.

8. Isolamento e Segmentazione della Rete:

- 8.1. Utilizzo di tecnologie per limitare la propagazione di un attacco all'interno della rete.
- 8.2. Segmentazione della rete per separare le risorse critiche e ridurre la superficie di attacco.
- 8.3. Isolamento di sistemi compromessi per prevenire la diffusione di minacce.

1. Implementazione di Sistemi di Monitoraggio:

- 1.1. Adozione di soluzioni avanzate di monitoraggio per sorvegliare costantemente l'ambiente informatico.
- 1.2. Utilizzo di strumenti specializzati per rilevare attività sospette e comportamenti anomali.
- 1.3. Configurazione di allarmi e notifiche per avvertire tempestivamente gli amministratori in caso di potenziali minacce.

2. Rilevamento di Attività Sospette:

- 2.1. Analisi approfondita dei log di sistema, delle registrazioni di rete e delle attività utente per individuare comportamenti non conformi.
- 2.2. Implementazione di algoritmi e regole personalizzate per identificare pattern indicativi di attività malevole.
- 2.3. Monitoraggio costante dei flussi di dati per rilevare variazioni insolite o accessi non autorizzati.

3. Interruzione Tempestiva:

- 3.1. Sviluppo di procedure e protocolli per rispondere rapidamente alle attività sospette rilevate.
- 3.2. Implementazione di contromisure automatiche o interventi manuali per interrompere le attività malevole.
- 3.3. Collaborazione con sistemi di risposta agli incidenti per un intervento rapido e coordinato.

4. Analisi Forense:

- 4.1. Conservazione accurata dei dati di monitoraggio per condurre analisi forensi in caso di incidenti.
- 4.2. Utilizzo di strumenti e metodologie forensi per comprendere l'entità e la natura di un'eventuale violazione.
- 4.3. Documentazione dettagliata delle azioni intraprese durante la risposta agli incidenti per migliorare la preparazione futura.

5. Integrazione con Sistemi di Prevenzione:

- 5.1. Coordinamento dei sistemi di monitoraggio con soluzioni di prevenzione delle intrusioni.
- 5.2. Attuazione di politiche automatiche di prevenzione in risposta alle minacce identificate.
- 5.3. Monitoraggio continuo dell'efficacia delle contromisure adottate e aggiornamenti periodici in base all'evoluzione delle minacce.

6. Formazione del Personale:

- 6.1. Sensibilizzazione degli utenti sull'importanza della collaborazione nel segnalare attività sospette.
- 6.2. Offerta di formazione regolare per il personale per riconoscere segnali di minacce potenziali.
- 6.3. Creazione di una cultura aziendale orientata alla sicurezza informatica attraverso la consapevolezza degli utenti.

7. Audit e Revisione Continua:

- 7.1. Conduzione di audit regolari per valutare l'efficacia dei sistemi di monitoraggio.
- 7.2. Revisione continua delle politiche di monitoraggio per adattarsi alle nuove minacce e alle mutevoli condizioni operative.
- 7.3. Collaborazione con esperti di sicurezza esterni per una valutazione imparziale della robustezza del sistema di monitoraggio.

Conclusioni e Grafico Malware:

La sostituzione di un file .dll legittimo con una variante dannosa presenta rischi significativi che richiedono un approccio globale alla sicurezza.

Questa minaccia comporta potenziali violazioni della sicurezza, rischi legali, impatti sull'integrità del sistema e possibili fughe di informazioni sensibili. Affrontare efficacemente questa sfida implica l'implementazione di difese multistrato, comprese soluzioni avanzate di rilevamento delle intrusioni, autenticazione multi fattore e gestione proattiva delle vulnerabilità. Il monitoraggio continuo e l'adattamento delle strategie di sicurezza sono essenziali per fronteggiare la natura in evoluzione delle minacce informatiche.

In definitiva, la prevenzione, la rilevazione tempestiva e una risposta rapida sono fondamentali per preservare l'integrità e la sicurezza dei sistemi informativi aziendali.

