

PROJECT MODULE 6

FATIMA NARDA NICOLETT RODRIGUEZ ORTEGA

1) TRACCIA

Malware Analysis

Il Malware da analizzare è nella cartella Build_Week_Unit_3 presente sul desktop della macchina virtuale dedicata.

Analisi statica

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

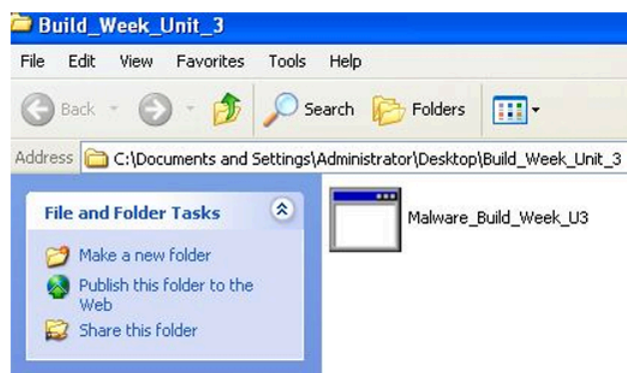
- Quanti parametri sono passati alla funzione Main()?
- Quante variabili sono dichiarate all'interno della funzione Main()?
- Quali sezioni sono presenti all'interno del file eseguibile? Descrivete brevemente almeno 2 di quelle identificate
- Quali librerie importa il Malware? Per ognuna delle librerie importate, fate delle ipotesi sulla base della sola analisi statica delle funzionalità che il Malware potrebbe implementare. Utilizzate le funzioni che sono richiamate all'interno delle librerie per supportare le vostre ipotesi.

Con riferimento al Malware in analisi, spiegare:

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

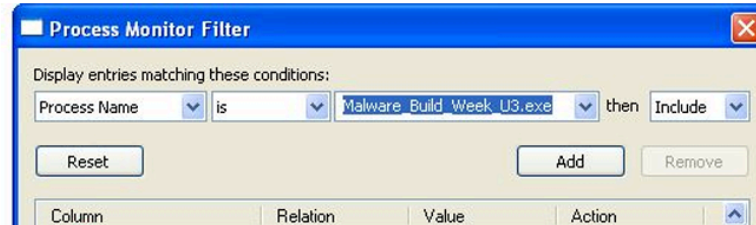
Analisi dinamica

Preparate l'ambiente ed i tool per l'esecuzione del Malware (suggerimento: avviate principalmente Process Monitor ed assicurate di eliminare ogni filtro cliccando sul tasto «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

Analizzate ora i risultati di Process Monitor (consiglio: utilizzate 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.



Filtrate includendo solamente l'attività sul registro di Windows.

- Quale chiave di registro viene creata?
- Quale valore viene associato alla chiave di registro creata?

Passate ora alla visualizzazione dell'attività sul file system.

- Quale chiamata di sistema ha modificato il contenuto della cartella dove è presente l'eseguibile del Malware?

Unite tutte le informazioni raccolte fin qui sia dall'analisi statica che dall'analisi dinamica per delineare il funzionamento del Malware.

2) SVOLGIMENTO

● ANALISI STATICA

Analizziamo il malware nel file Build_Week_Unit_3 su IDA Pro.

```
; Attributes: bp-based frame

; int __cdecl main(int argc, const char **argv, const char **envp)
_main proc near

hModule= dword ptr -11Ch
Data= byte ptr -118h
var_117= byte ptr -117h
var_8= dword ptr -8
var_4= dword ptr -4
argc= dword ptr 8
argv= dword ptr 0Ch
envp= dword ptr 10h
```

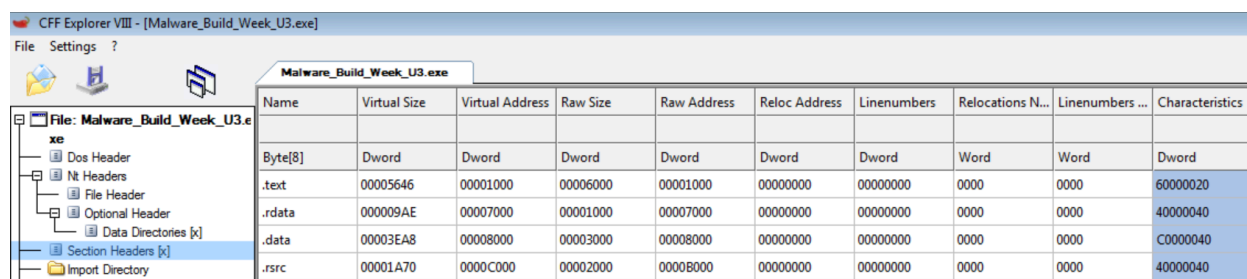
Nella definizione della funzione Main() troviamo i parametri:

- argc: ovvero "Argument count", il conteggio degli argomenti;
- argv: ovvero "Argument vector", un vettore di argomenti;
- envp: ovvero "Environment pointer", un puntatore dell'ambiente del sistema.

All'interno della funzione Main() vengono dichiarate quattro variabili:

- hModule, un puntatore a dword;
- Data, un puntatore a byte;
- var_8, un dword;
- var_4, un dword.

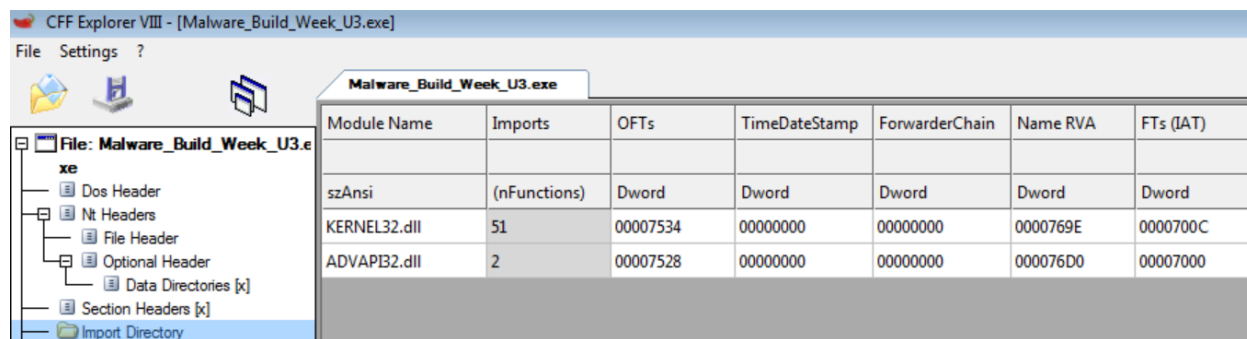
Analizziamo il malware anche con CFF Explorer.



Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations N...	Linenumbers ...	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

Possiamo vedere che nel file eseguibile ci sono 4 sezioni, di cui:

- .text: sezione contenente il codice eseguibile del programma,
- .data: sezione contenente dati inizializzati esplicitamente che il programma può leggere e modificare durante l'esecuzione.



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

Notiamo anche che il malware importa 2 librerie:

- KERNEL32.dll: fornisce funzioni necessarie per interagire con il sistema operativo, ad esempio la gestione dei file e della memoria;
- ADVAPI32.dll: fornisce funzioni avanzate per la gestione dei servizi e della sicurezza del sistema, come l'accesso ai registri di sistema.

```
.text:00401021          call     ds:RegCreateKeyExA
```

Nell'indirizzo 00401021, con l'istruzione *call* è stata chiamata la funzione [RegCreateKeyExA che ha lo scopo di creare una chiave di registro nel sistema.

```
.text:00401017          push     offset SubKey ; "SOFTWARE\Microsoft\Windows NT\CurrentVe"...  
'SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon'
```

Nell'indirizzo 00401017, troviamo una chiave di registro che contiene le informazioni di configurazione per il processo di accesso al sistema(Winlogon).

```
.text:00401027          test     eax, eax  
.text:00401029          jz       short loc_401032
```

Agli indirizzi 00401027 e 00401029, troviamo 2 istruzioni collegate tra loro:

- *test*, effettuando un'operazione AND, controlla se il registro *eax* è equivalente a zero;
- *jz*, ovvero *jump if zero*, effettua un salto all'indirizzo 00401032 se il test precedente risulta in zero.

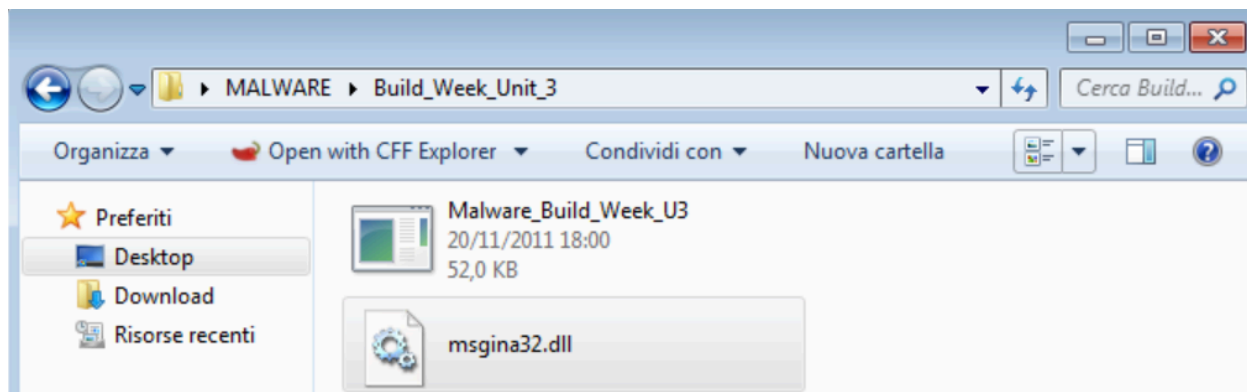
Il corrispondente costruito in linguaggio C è:

```
if ( eax == 0 ) {  
    goto loc_401032;  
}
```

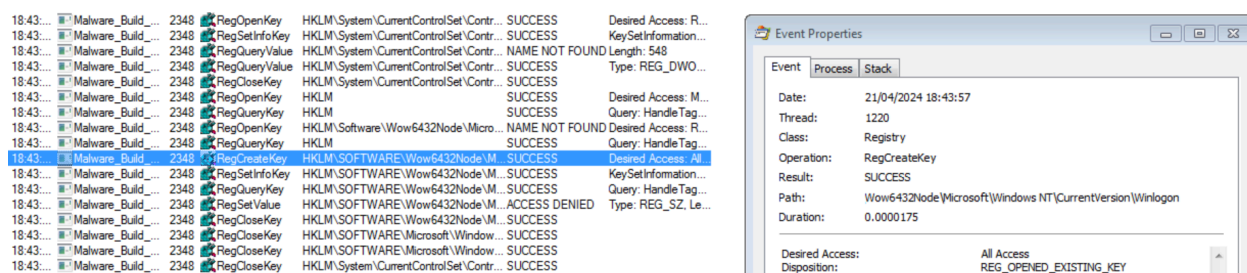
All'indirizzo 00401047, la chiamata della funzione *RegSetValueExA* usa il parametro *ValueName*, che è "GinaDLL".

```
loc_401032:  
mov     ecx, [ebp+cbData]  
push    ecx ; cbData  
mov     edx, [ebp+lpData]  
push    edx ; lpData  
push    1 ; dwType  
push    0 ; Reserved  
push    offset ValueName ; "GinaDLL"  
mov     eax, [ebp+hObject]  
push    eax ; hKey  
call    ds:RegSetValueExA
```

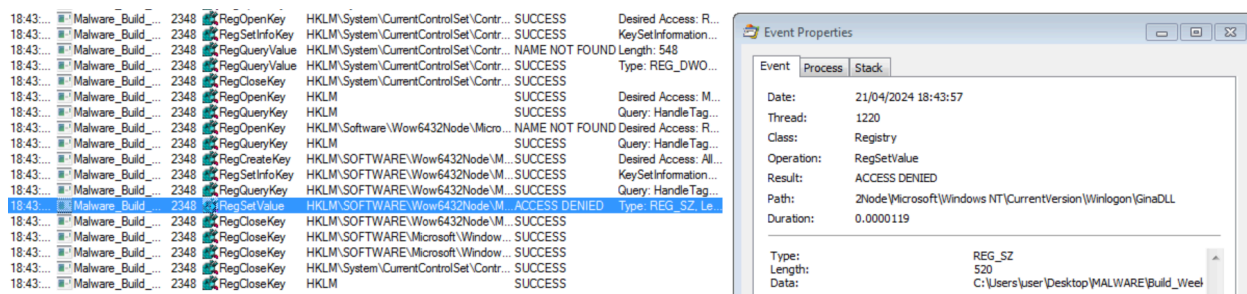
● ANALISI DINAMICA



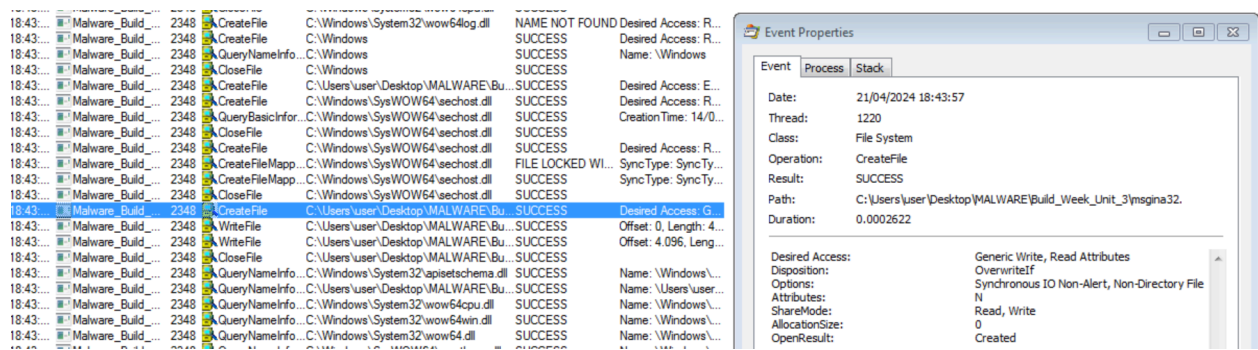
Dopo aver eseguito il malware, è stato creato il file *msgina32.dll*. Esso è la versione corrotta della libreria *GinaDLL*, una componente importante nel sistema, infatti essa gestisce il processo di autenticazione degli utenti.



La chiave di registro creata dal malware è *Winlogon*.



Il valore assegnato è "*GinaDLL*".



La chiamata di sistema che ha modificato il contenuto del file “Build_Week_Unit_3” è *CreateFile*, creando il file *msgina32.dll*.

● CONCLUSIONE

Il malware è un dropper, perchè, iniziata la sua esecuzione, ha estratto il file malevolo *msgina32.dll* che immagazzina dati sul processo di autenticazione degli utenti. Inoltre, per estrarre il file malevolo sono stati usati APIs come:

00007632	00007632	0295	SizeofResource
00007644	00007644	01D5	LockResource
00007654	00007654	01C7	LoadResource