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:

1. Quanti parametri sono passati alla funzione Main()?
2. Quante variabili sono dichiarate all'interno della funzione Main()?
3. Quali sezioni sono presenti all'interno del file eseguibile? Descrivete brevemente almeno 2 di quelle identificate
4. 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:

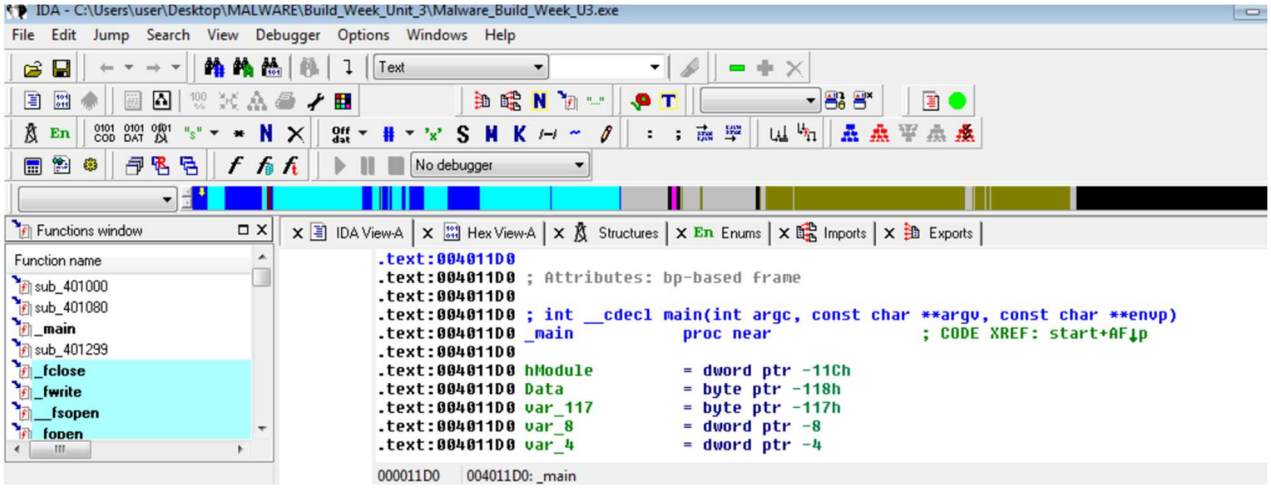
1. Lo scopo della funzione chiamata alla locazione di memoria 00401021
2. Come vengono passati i parametri alla funzione alla locazione 00401021;
3. Che oggetto rappresenta il parametro alla locazione 00401017
4. Il significato delle istruzioni comprese tra gli indirizzi 00401027 e 00401029.
5. Con riferimento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

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.

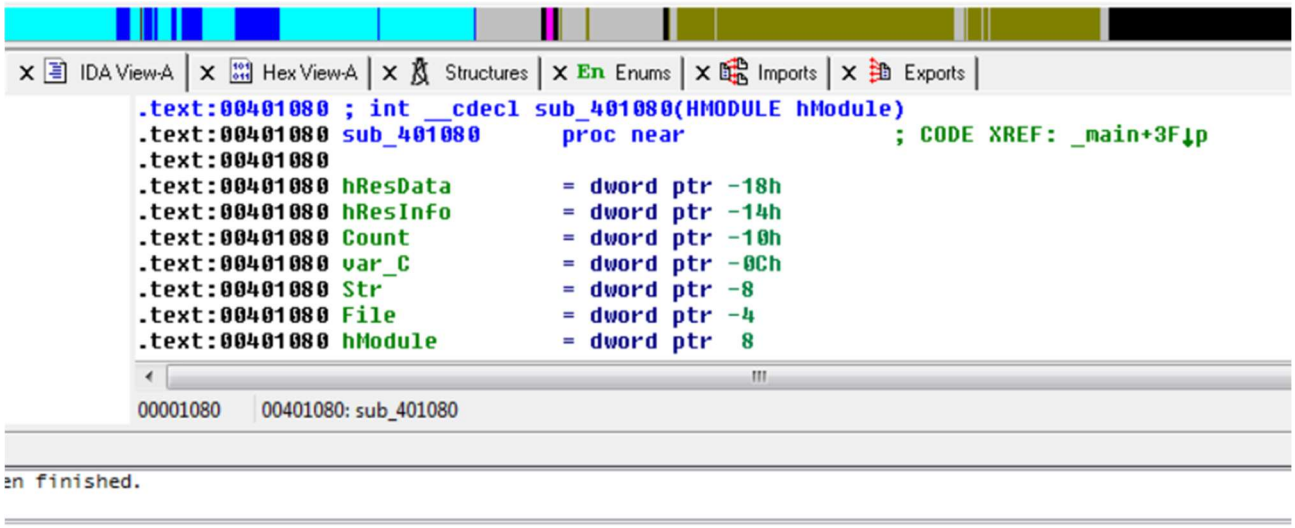
Filtrate includendo solamente l'attività sul registro di Windows:

1. Quale chiave di registro viene creata?
2. Quale valore viene associato alla chiave di registro creata? Passate ora alla visualizzazione dell'attività sul file system.
3. Quale chiamata di sistema ha modificato il contenuto della cartella dove è presente l'eseguibile del Malware?
4. Unite tutte le informazioni raccolte fin qui sia dall'analisi statica che dall'analisi dinamica per delineare il funzionamento del Malware.

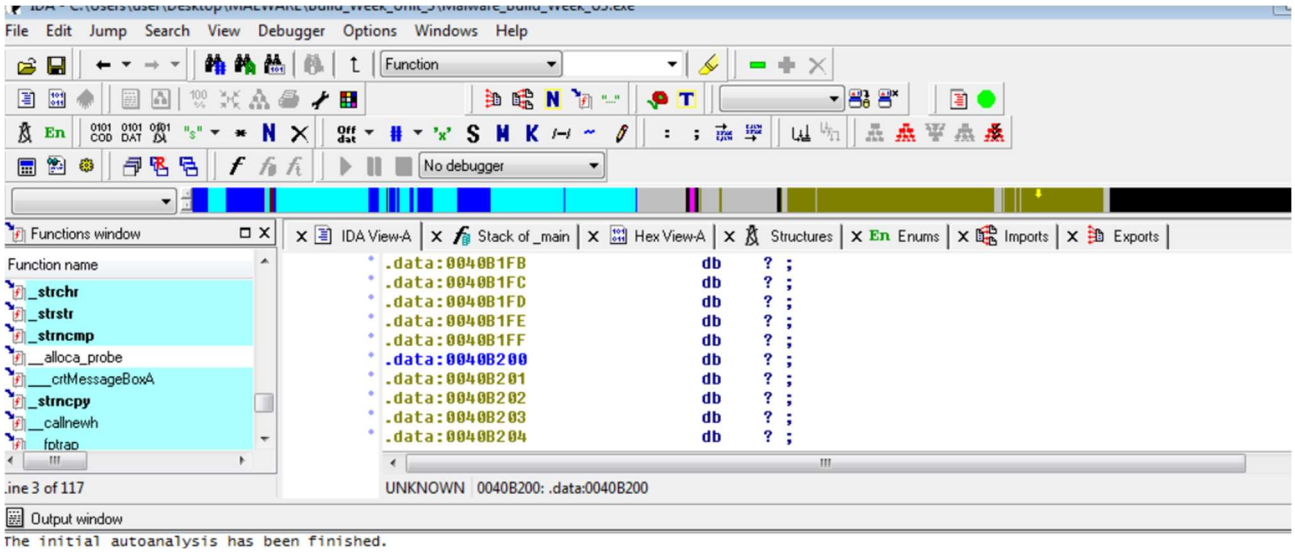
1.



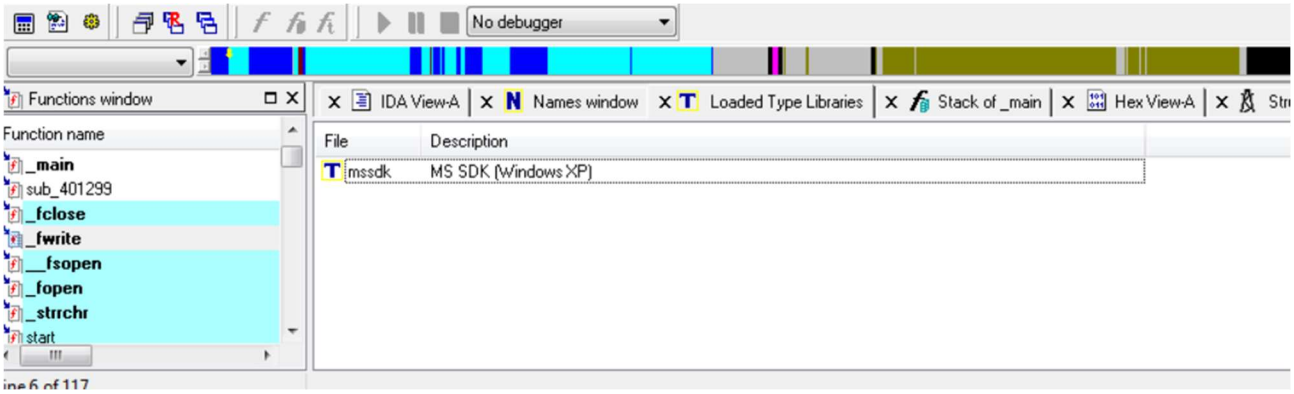
2.



3. .TEXT : contiene il codice eseguibile.
.DATA: contiene dati globali accessibili da ogni punto del programma.

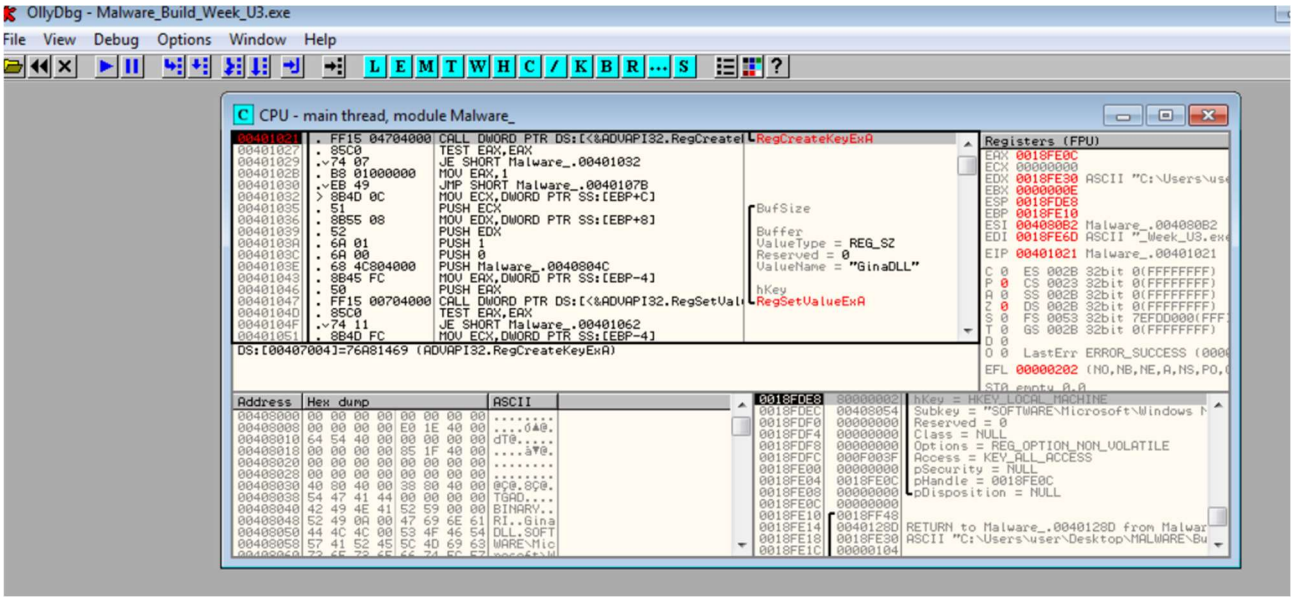


4.

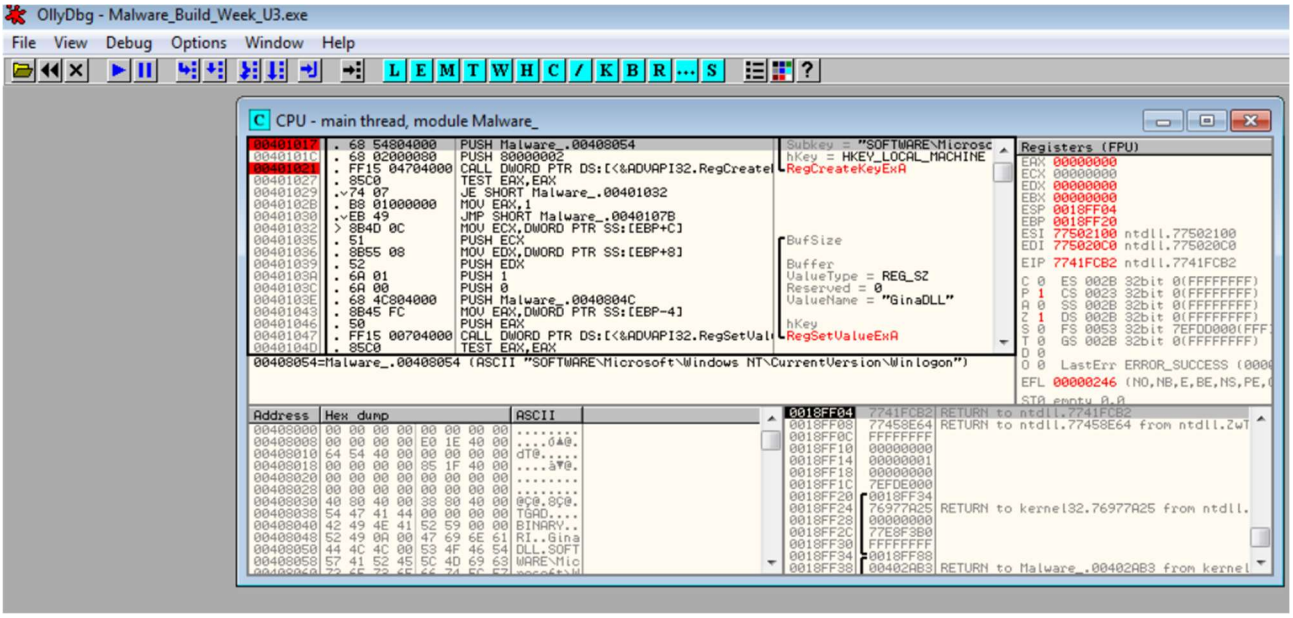


- **Advapi32.dll**: questa libreria fornisce funzioni per la gestione dell'accesso al registro di sistema, dei servizi, della crittografia e delle autorizzazioni di sistema. I malware possono importare funzioni da questa libreria per modificare le autorizzazioni di file o registro, creare servizi maligni o crittografare/decrittografare dati.

1.



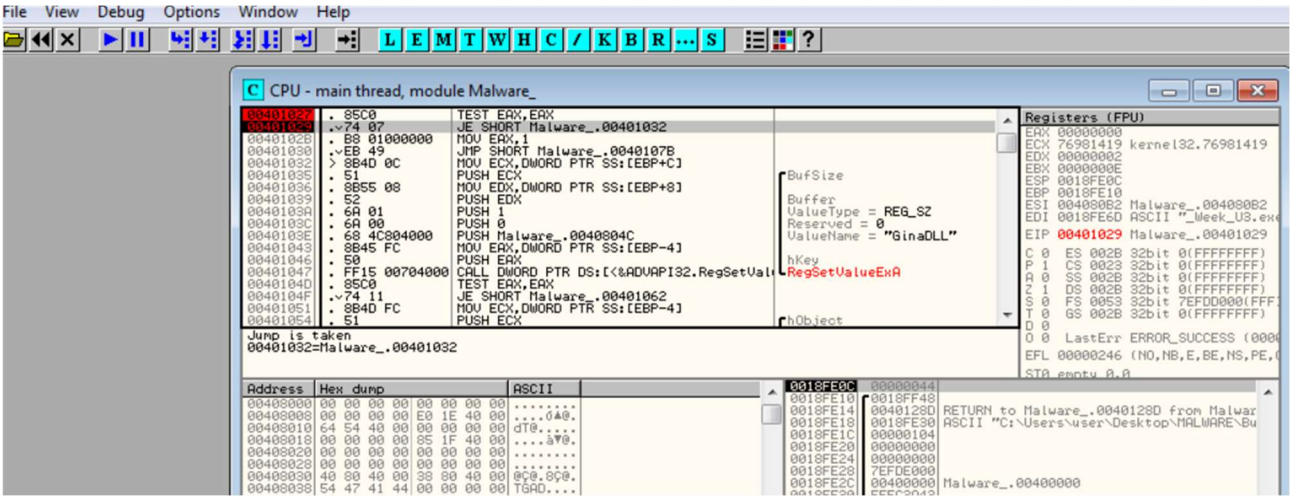
RegCreateKeyExA: che fa parte della libreria di sistema Windows ADVAPI32.dll. Questa funzione viene utilizzata per creare o aprire una chiave di registro in base al percorso specificato. Il suo scopo è di creare o aprire una chiave di registro nel registro di sistema Windows, che può essere utilizzata per memorizzare informazioni di configurazione, impostazioni o altre informazioni utili al malware. Ad esempio, il malware potrebbe utilizzare questa funzione per creare una chiave di registro per memorizzare le proprie impostazioni o per garantire la persistenza nel sistema operativo, creando una voce di avvio automatico nel registro di sistema.

2. La chiamata alla funzione avviene indirettamente attraverso il puntatore alla funzione situato all'indirizzo di memoria `00401021`. Per passare i parametri alla funzione `RegCreateKeyExA`, i valori dei parametri devono essere precedentemente preparati e memorizzati nello stack. Nell'assembly x86, i parametri delle funzioni sono generalmente passati attraverso lo stack, e vengono letti dalla funzione chiamata nell'ordine inverso rispetto a come sono stati inseriti nello stack. Pertanto, prima della chiamata alla funzione `RegCreateKeyExA`, ci si aspetta che i parametri siano stati precedentemente caricati nello stack. Solitamente, questi parametri includono il puntatore alla chiave di registro padre, il nome della sottochiave da creare o aprire, l'offset di sicurezza e altri parametri opzionali.
3. 

Il parametro all'indirizzo di memoria 00401017 è rappresentato dal valore 54804000, che viene inserito nello stack tramite l'istruzione `PUSH`. Questo valore rappresenta l'indirizzo di memoria in cui è memorizzato il nome della sottochiave del registro da creare o aprire.

Il nome della sottochiave del registro è "SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon" e viene utilizzato come parametro per la funzione chiamata successivamente. Questo indica che il malware sta cercando di creare o aprire una chiave di registro con il nome specificato, probabilmente per scopi di persistenza o per effettuare modifiche nel registro di sistema.

4.



Le istruzioni assembly fornite sono:

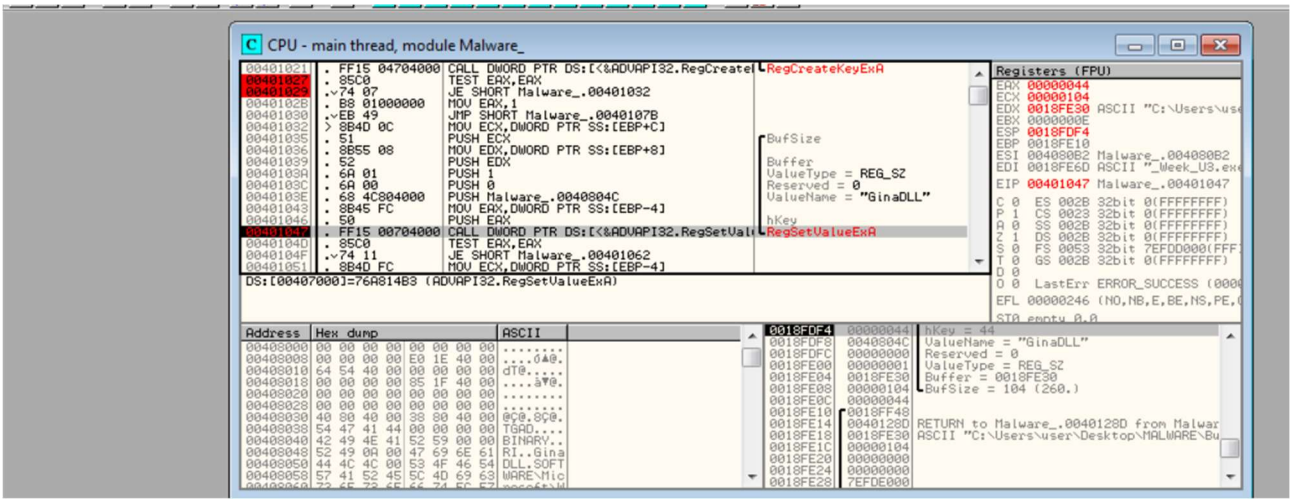
1. `TEST EAX, EAX`: Questa istruzione esegue un'operazione di AND bit a bit tra il registro `EAX` e se stesso. È comunemente utilizzata per verificare se un registro è uguale a zero. Se `EAX` è zero, il flag ZF (Zero Flag) sarà impostato a 1, altrimenti sarà a 0.

2. `JE SHORT Malware_.00401032`: Questa istruzione è una condizione di salto condizionato. Se il flag ZF (Zero Flag) è impostato a 1 (indicando che `EAX` è zero), il controllo salterà all'indirizzo specificato, in questo caso Malware_.00401032, altrimenti proseguirà con l'esecuzione delle istruzioni successive.

Quindi, nel complesso, queste istruzioni controllano se il registro `EAX` è zero. Se lo è, il controllo salterà all'indirizzo Malware_.00401032, altrimenti continuerà con le istruzioni successive. Questo è spesso utilizzato per gestire condizioni e decisioni nel flusso di esecuzione del programma.

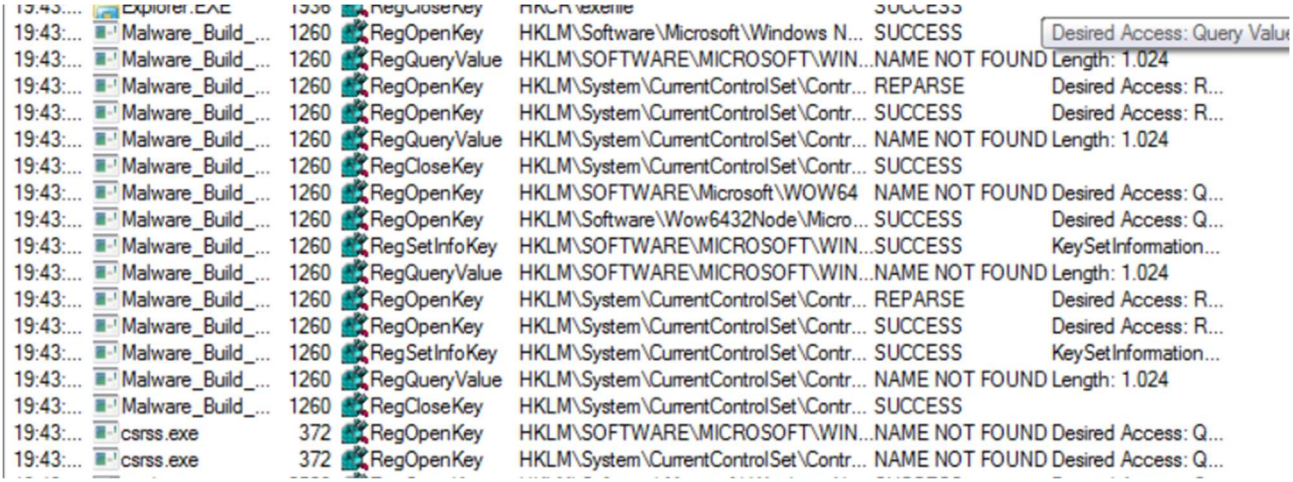
5.
```
// Test se il registro EAX è uguale a zero
if (EAX == 0) {
 // Salta a Malware_.00401032 se EAX è zero
} else {
 // Continua con le istruzioni successive se EAX non è zero
}
```

6.

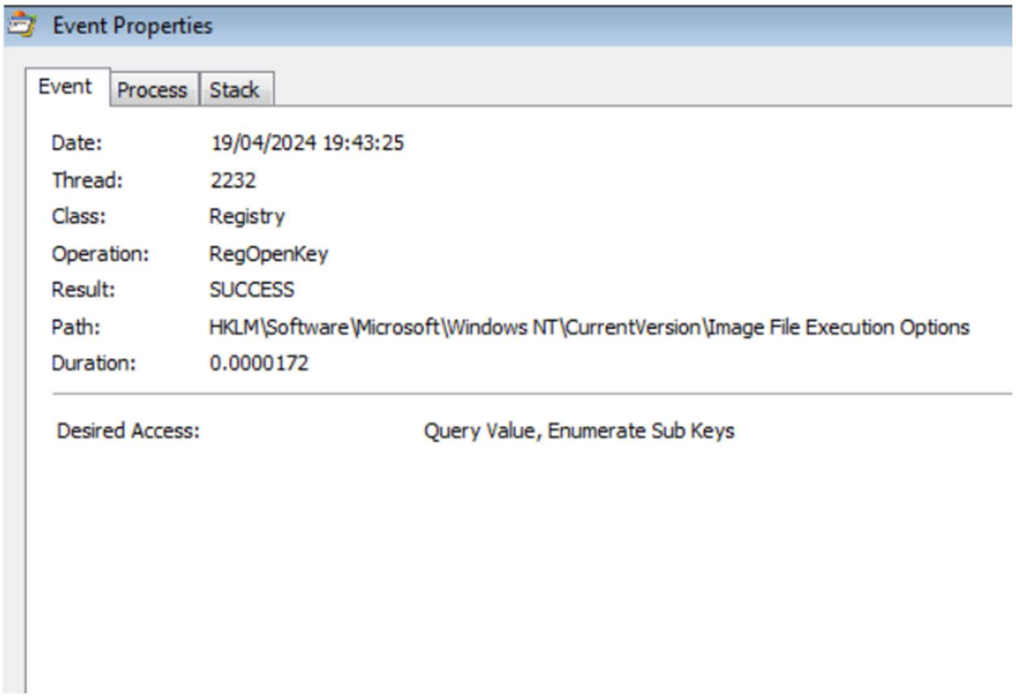


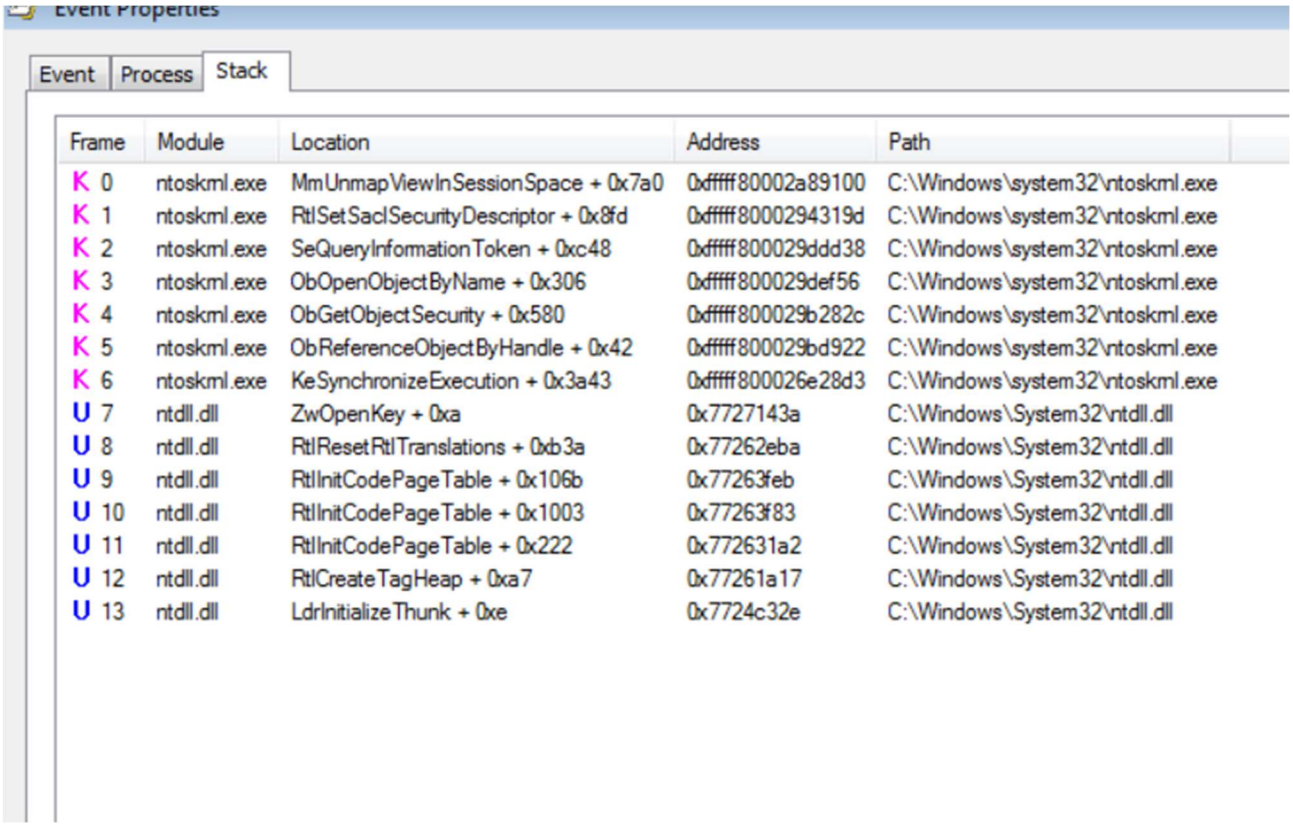
La funzione `RegSetValueExA` viene utilizzata per impostare il valore di un'entry del registro. Di solito, il parametro `valuename` rappresenta il nome dell'entry del registro cui si desidera impostare il valore. Quindi, il valore del parametro `valuename` potrebbe essere il nome dell'entry del registro che il malware sta cercando di modificare. Potrebbe essere un nome di una chiave del registro, una sottochiave o un valore specifico all'interno di una chiave del registro.

1.

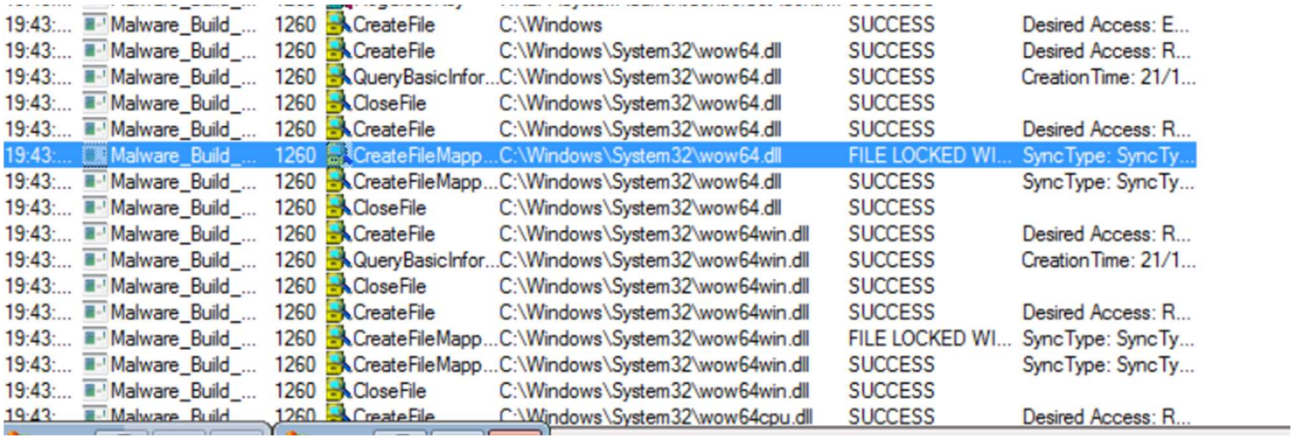


2.

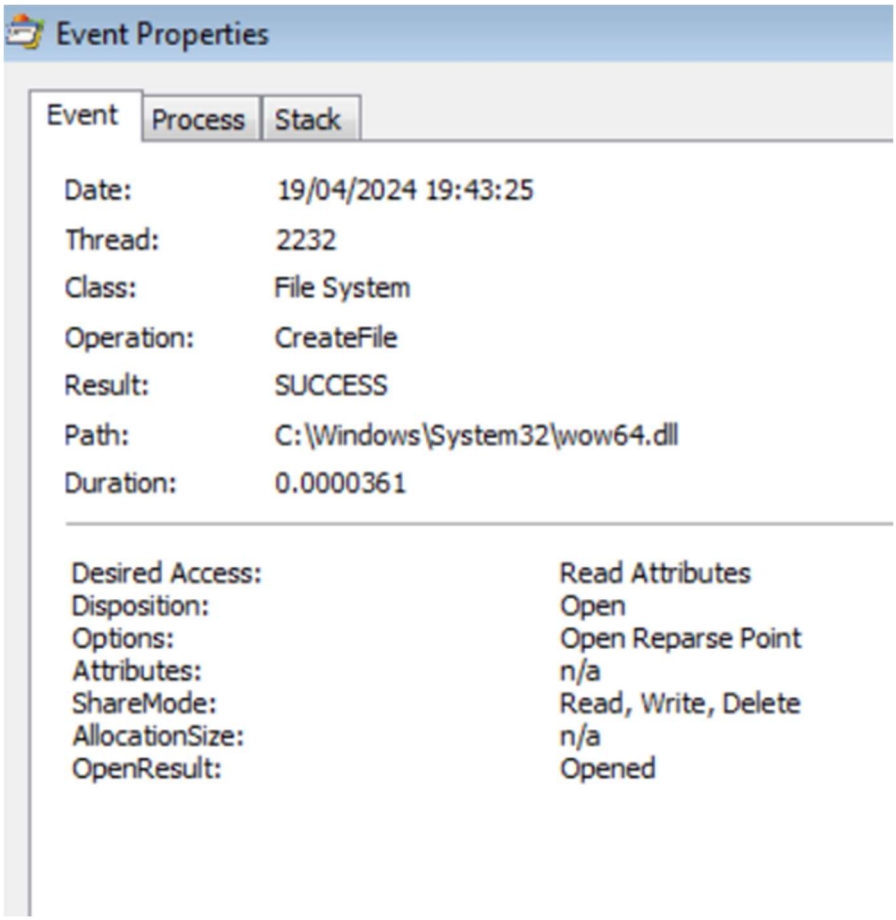

Event Properties
Event
Process
Stack
Date: 19/04/2024 19:43:25
Thread: 2232
Class: Registry
Operation: RegOpenKey
Result: SUCCESS
Path: HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options
Duration: 0.0000172
Desired Access: Query Value, Enumerate Sub Keys


Event Properties
Event
Process
Stack
Frame Module Location Address Path
K 0 ntoskrnl.exe MmUnmapViewInSessionSpace + 0x7a0 0xfffff80002a89100 C:\Windows\system32\ntoskrnl.exe
K 1 ntoskrnl.exe RtlSetSaclSecurityDescriptor + 0x8fd 0xfffff8000294319d C:\Windows\system32\ntoskrnl.exe
K 2 ntoskrnl.exe SeQueryInformationToken + 0xc48 0xfffff800029ddd38 C:\Windows\system32\ntoskrnl.exe
K 3 ntoskrnl.exe ObOpenObjectByName + 0x306 0xfffff800029def56 C:\Windows\system32\ntoskrnl.exe
K 4 ntoskrnl.exe ObGetObjectSecurity + 0x580 0xfffff800029b282c C:\Windows\system32\ntoskrnl.exe
K 5 ntoskrnl.exe ObReferenceObjectByHandle + 0x42 0xfffff800029bd922 C:\Windows\system32\ntoskrnl.exe
K 6 ntoskrnl.exe KeSynchronizeExecution + 0x3a43 0xfffff800026e28d3 C:\Windows\system32\ntoskrnl.exe
U 7 ntdll.dll ZwOpenKey + 0xa 0x7727143a C:\Windows\System32\ntdll.dll
U 8 ntdll.dll RtlResetRtlTranslations + 0xb3a 0x77262eba C:\Windows\System32\ntdll.dll
U 9 ntdll.dll RtlInitCodePageTable + 0x106b 0x77263feb C:\Windows\System32\ntdll.dll
U 10 ntdll.dll RtlInitCodePageTable + 0x1003 0x77263f83 C:\Windows\System32\ntdll.dll
U 11 ntdll.dll RtlInitCodePageTable + 0x222 0x772631a2 C:\Windows\System32\ntdll.dll
U 12 ntdll.dll RtlCreateTagHeap + 0xa7 0x77261a17 C:\Windows\System32\ntdll.dll
U 13 ntdll.dll LdrInitializeThunk + 0xe 0x7724c32e C:\Windows\System32\ntdll.dll

3.



19:43:...	Malware_Build_...	1260	CreateFile	C:\Windows	SUCCESS	Desired Access: E...
19:43:...	Malware_Build_...	1260	CreateFile	C:\Windows\System32\wow64.dll	SUCCESS	Desired Access: R...
19:43:...	Malware_Build_...	1260	QueryBasicInfor...	C:\Windows\System32\wow64.dll	SUCCESS	CreationTime: 21/1...
19:43:...	Malware_Build_...	1260	CloseFile	C:\Windows\System32\wow64.dll	SUCCESS	
19:43:...	Malware_Build_...	1260	CreateFile	C:\Windows\System32\wow64.dll	SUCCESS	Desired Access: R...
19:43:...	Malware_Build_...	1260	CreateFileMapp...	C:\Windows\System32\wow64.dll	FILE LOCKED WI...	SyncType: SyncTy...
19:43:...	Malware_Build_...	1260	CreateFileMapp...	C:\Windows\System32\wow64.dll	SUCCESS	SyncType: SyncTy...
19:43:...	Malware_Build_...	1260	CloseFile	C:\Windows\System32\wow64.dll	SUCCESS	
19:43:...	Malware_Build_...	1260	CreateFile	C:\Windows\System32\wow64win.dll	SUCCESS	Desired Access: R...
19:43:...	Malware_Build_...	1260	QueryBasicInfor...	C:\Windows\System32\wow64win.dll	SUCCESS	CreationTime: 21/1...
19:43:...	Malware_Build_...	1260	CloseFile	C:\Windows\System32\wow64win.dll	SUCCESS	
19:43:...	Malware_Build_...	1260	CreateFile	C:\Windows\System32\wow64win.dll	SUCCESS	Desired Access: R...
19:43:...	Malware_Build_...	1260	CreateFileMapp...	C:\Windows\System32\wow64win.dll	FILE LOCKED WI...	SyncType: SyncTy...
19:43:...	Malware_Build_...	1260	CreateFileMapp...	C:\Windows\System32\wow64win.dll	SUCCESS	SyncType: SyncTy...
19:43:...	Malware_Build_...	1260	CloseFile	C:\Windows\System32\wow64win.dll	SUCCESS	
19:43:...	Malware_Build_...	1260	CreateFile	C:\Windows\System32\wow64cpu.dll	SUCCESS	Desired Access: R...

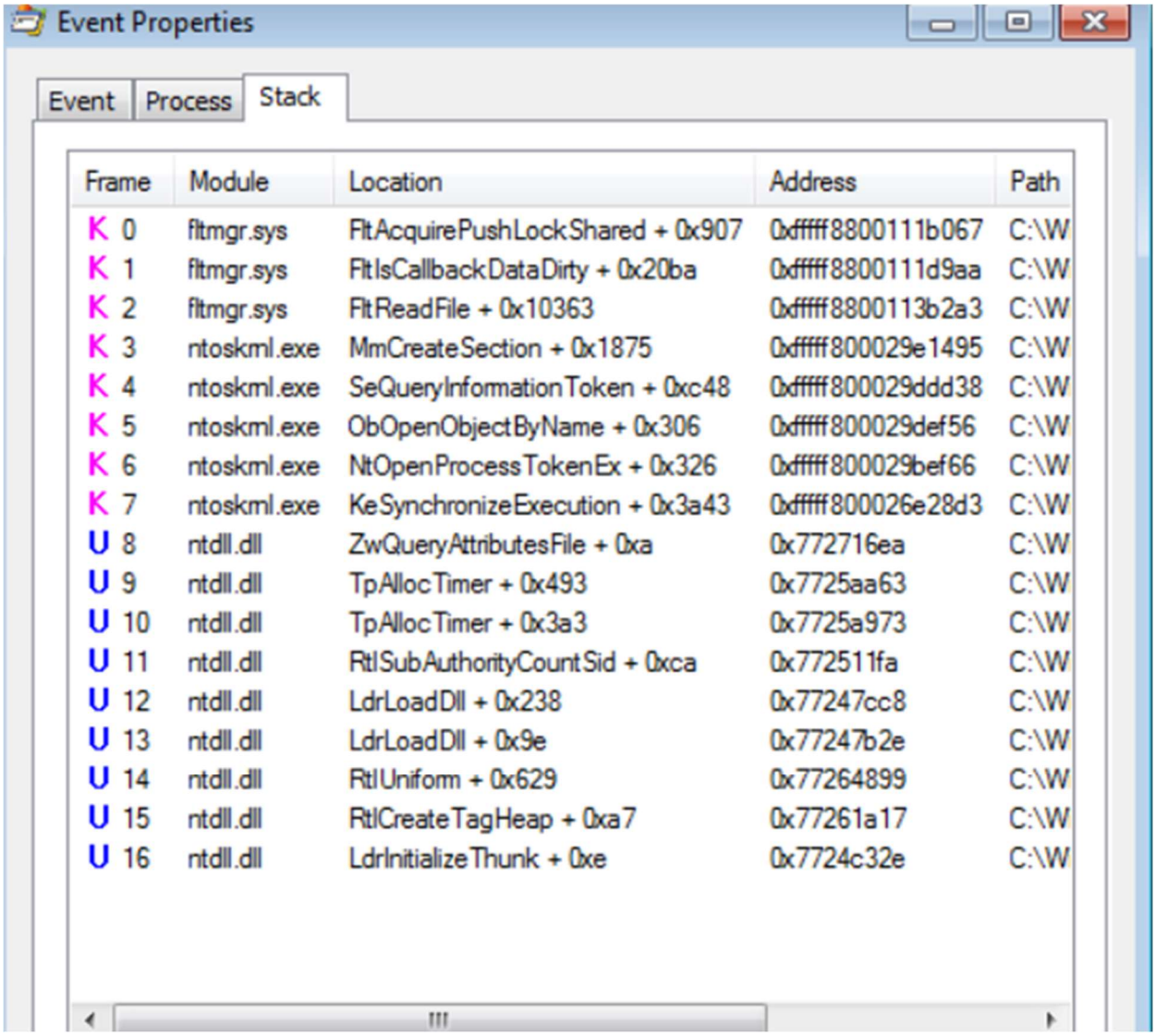


Event Properties

Event | Process | Stack

Date:	19/04/2024 19:43:25
Thread:	2232
Class:	File System
Operation:	CreateFile
Result:	SUCCESS
Path:	C:\Windows\System32\wow64.dll
Duration:	0.0000361

Desired Access:	Read Attributes
Disposition:	Open
Options:	Open Reparse Point
Attributes:	n/a
ShareMode:	Read, Write, Delete
AllocationSize:	n/a
OpenResult:	Opened


Event Properties
Event
Process
Stack
Frame Module Location Address Path
K 0 fltmgr.sys FltAcquirePushLockShared + 0x907 0xfffff8800111b067 C:\W
K 1 fltmgr.sys FltIsCallbackDataDirty + 0x20ba 0xfffff8800111d9aa C:\W
K 2 fltmgr.sys FltReadFile + 0x10363 0xfffff8800113b2a3 C:\W
K 3 ntoskrnl.exe MmCreateSection + 0x1875 0xfffff800029e1495 C:\W
K 4 ntoskrnl.exe SeQueryInformationToken + 0xc48 0xfffff800029ddd38 C:\W
K 5 ntoskrnl.exe ObOpenObjectByName + 0x306 0xfffff800029def56 C:\W
K 6 ntoskrnl.exe NtOpenProcessTokenEx + 0x326 0xfffff800029bef66 C:\W
K 7 ntoskrnl.exe KeSynchronizeExecution + 0x3a43 0xfffff800026e28d3 C:\W
U 8 ntdll.dll ZwQueryAttributesFile + 0xa 0x772716ea C:\W
U 9 ntdll.dll TpAllocTimer + 0x493 0x7725aa63 C:\W
U 10 ntdll.dll TpAllocTimer + 0x3a3 0x7725a973 C:\W
U 11 ntdll.dll RtlSubAuthorityCountSid + 0xca 0x772511fa C:\W
U 12 ntdll.dll LdrLoadDll + 0x238 0x77247cc8 C:\W
U 13 ntdll.dll LdrLoadDll + 0x9e 0x77247b2e C:\W
U 14 ntdll.dll RtlUniform + 0x629 0x77264899 C:\W
U 15 ntdll.dll RtlCreateTagHeap + 0xa7 0x77261a17 C:\W
U 16 ntdll.dll LdrInitializeThunk + 0xe 0x7724c32e C:\W