Giorno 2:

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

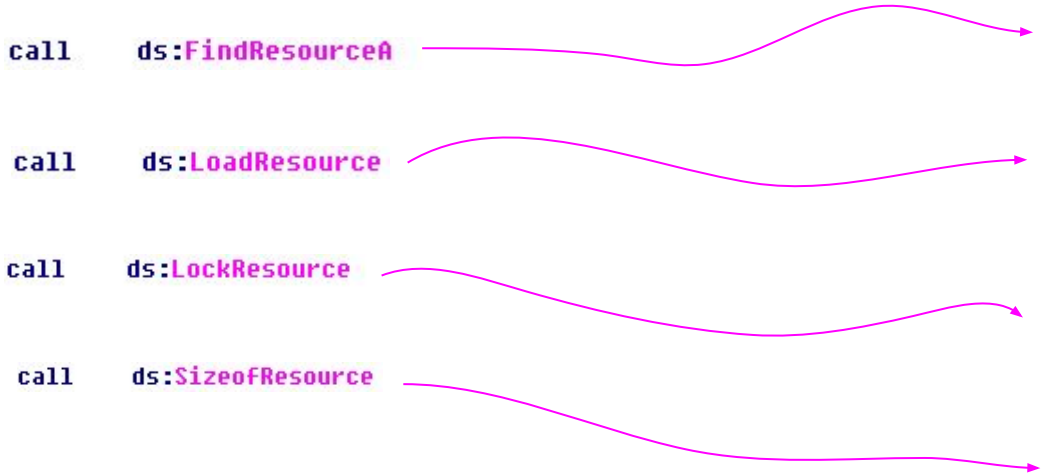
- Qual è il valore del parametro «ResourceName» passato alla funzione FindResourceA();

```
.text:00401088 ; ---------------------------------------------------------------------------
.text:00401088
.text:00401088 loc_401088:                             ; CODE XREF: sub_401080+2F↑j
.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; LPCSTR lpName
.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

lpName          dd offset aTgad         ; DATA XREF: sub_401080+3E↑r
                                        ; "TGAD"
```

"**lpName**" rappresenta l'identificatore della risorsa **ResourceName**. In questo caso il suo valore è "**TGAD**". "**DATA XREF**" è una notazione che mostra quale indirizzo di memoria ha usato questo parametro, nel nostro caso una **subroutine all'indirizzo 401080**, con **3E** che indica uno spostamento dopo l'indirizzo che corrisponde a quell'esadecimale **(in bytes)**, mentre i simboli "**↑r**" indicano un riferimento non assoluto all'etichetta **sub_401080**.

- 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?



FindResourceA: cerca una risorsa specifica all'interno di un modulo o di un file eseguibile

LoadResource: utilizzata per caricare una risorsa specifica identificata dall'handle del modulo e dall'handle della risorsa restituiti dalla funzione FindResourceA

LockResource: utilizzata per ottenere un puntatore ai dati di una risorsa caricata in memoria mediante la funzione LoadResource

SizeofResource: utilizzata per ottenere la dimensione, in byte, di una risorsa specifica all'interno di un modulo o di un file eseguibile.

Le seguenti **APIs** permettono di localizzare il malware e caricarlo in memoria per l'esecuzione. Da questa analisi possiamo presumere che sia un **Dropper** ovvero un malware che ha come scopo quello di installare altri tipi di malware, estraendoli dal proprio codice.
Una volta estratto il dropper generalmente propone 2 variabili ovvero:

1) Creazione di un processo dove utilizzerà le Apis precedentemente descritte
2) Salvataggio del malware per un utilizzo futuro. In questo caso utilizzerà **Createfile** e **WriteFile** presi dalla libreria **Kernel32.dll**

- È possibile identificare questa funzionalità utilizzando l'analisi statica basica?

Sì, perché andando ad utilizzare **CFF Explorer** è possibile identificare le librerie e le funzioni da esse importate, andando ad analizzare con più attenzione la **Resource Directory** riusciamo a vedere in in elenco la risorsa richiamata dalla funzione **FindResourceA()** la prima presente nella sezione di codice analizzata.

Resource Directory

- In caso di risposta affermativa, elencare le evidenze a supporto.

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

Resource Directory

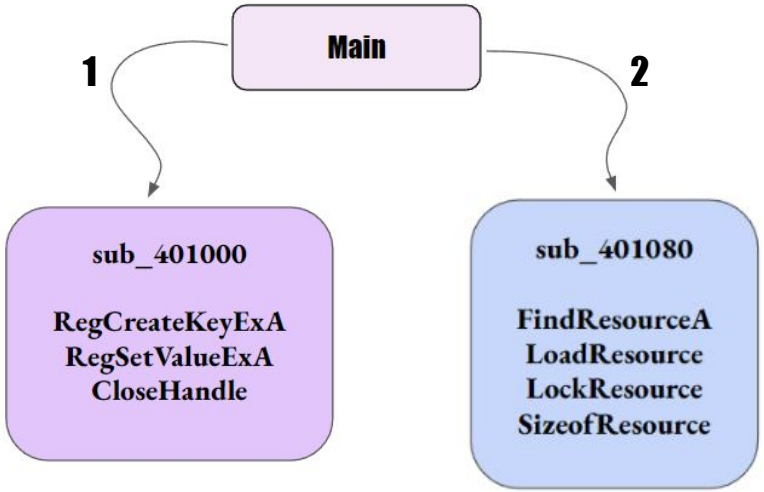
Chiamate alle funzioni: “**SizeofResource**”, “**LockResource**”, “**LoadResource**”, e “**FindResourceA**” suggeriscono l'interazione con le **API** di gestione delle risorse di Windows.

Riferimento a “KERNEL32.dll”: Indica l'uso di funzioni essenziali per l'accesso alle risorse del sistema operativo Windows.

Stringa "**Resource Directory Entry 1, Name: TGAD**": (Trusted Group Access Directory) È una directory service che consente agli utenti di accedere in modo sicuro alle risorse di rete. Utilizza un modello di autorizzazione basato su gruppi, che consente agli amministratori di assegnare facilmente i privilegi agli utenti.

Se queste chiamate e riferimenti sono all'interno di una parte del codice che gestisce risorse grafiche o file di risorse, suggerisce l'uso di risorse di sistema o incorporate nel file eseguibile stesso.

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 principali) che comprenda le 3 funzioni.



RegCreateKeyExA: Crea o apre una chiave nel registro di sistema di Windows.

RegSetValueExA: Imposta il valore di una voce nel registro di sistema.

CloseHandle: Questa funzione viene utilizzata per chiudere un handle di un oggetto aperto precedentemente. Gli handle vengono utilizzati per fare riferimento a risorse di sistema come file, porte, processi, ecc. Chiudere un handle indica che non è più necessario utilizzare quella risorsa.