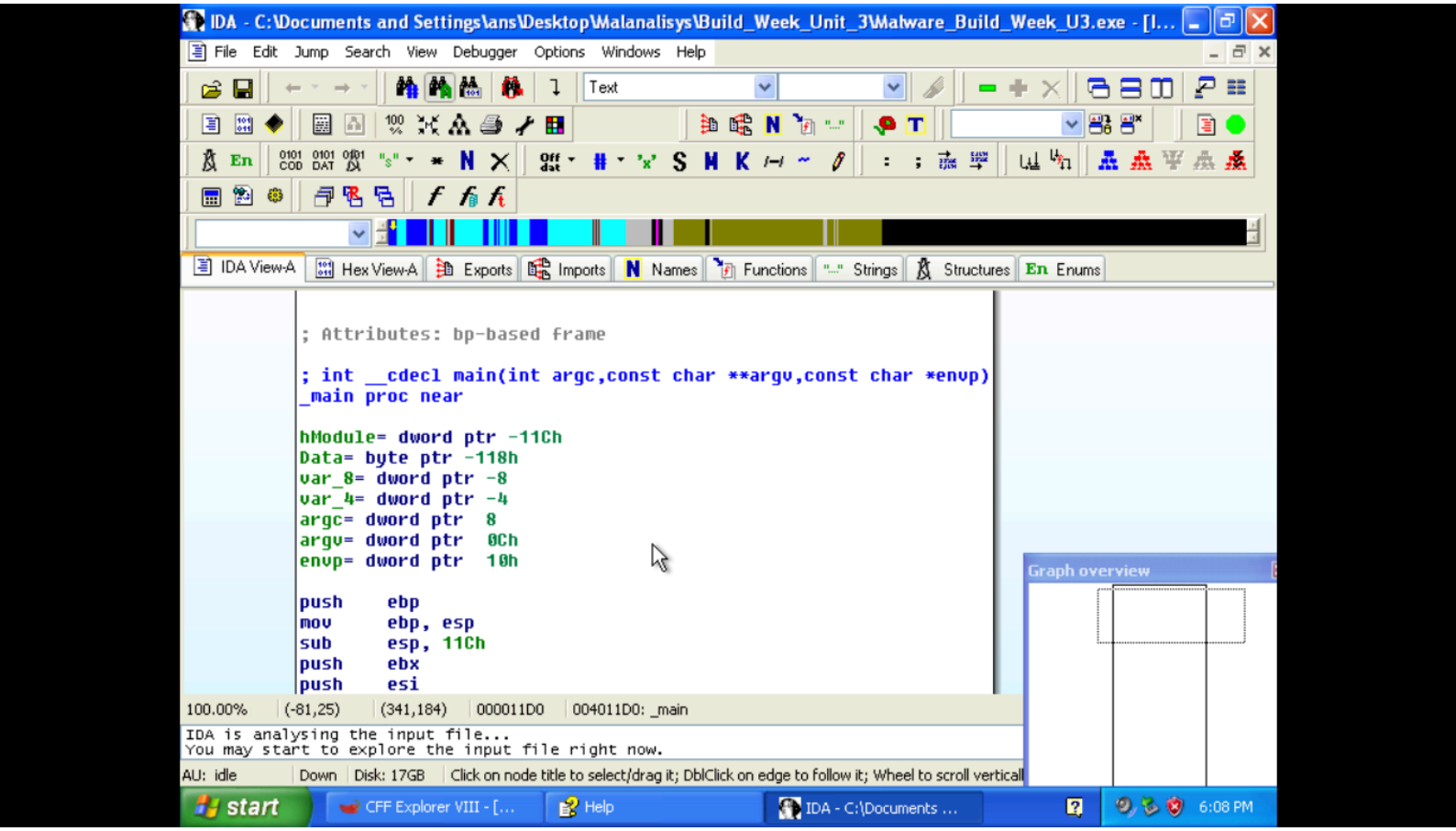
Malware Analysis

ANALISI STATICA



Come possiamo vedere dall'interfaccia del disassembler **IDA** abbiamo 3 parametri e 4 variabili per la funzione Main. Questi sono facilmente riconoscibili in quanto i parametri hanno un valore offset positivo rispetto al **EBP** (argc = 8) mentre le variabili negativo (hModule = -11Ch).

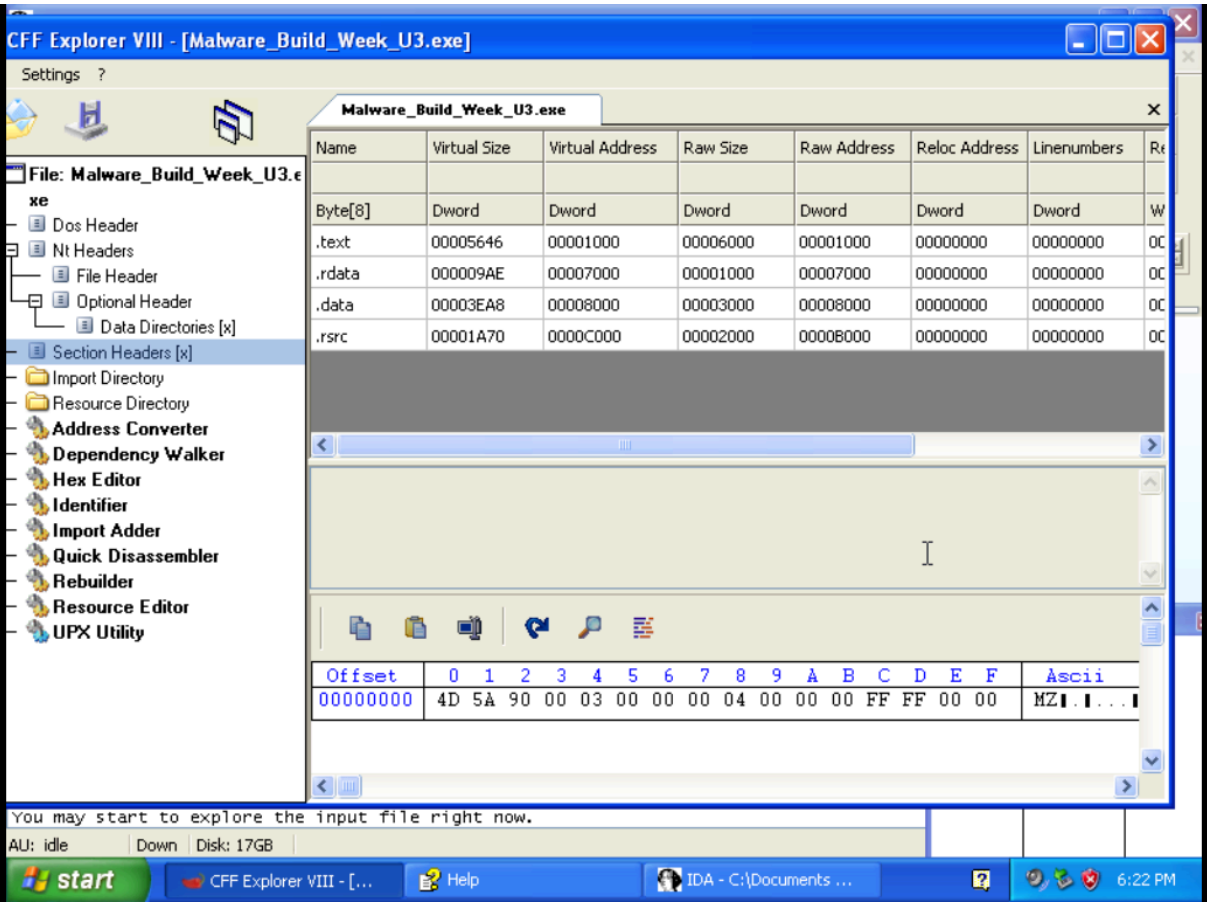
I **section headers** e le librerie importate invece possiamo trovarli tramite **CFF Explorer.** Le sezioni importate sono (nell'immagine successiva):

.text = contiene le righe di codice che saranno eseguite dalla CPU una volta avviato il malware.

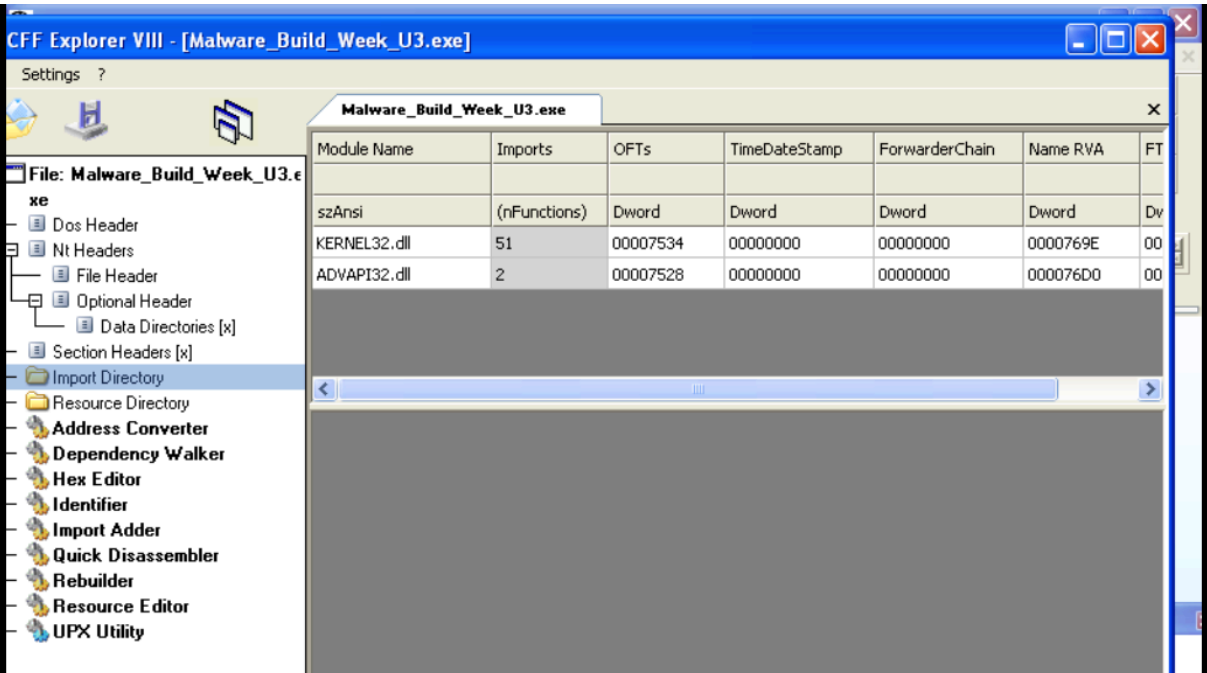
.rdata = sezione che contiene informazioni generali sulle librerie importate.

.data = al interno troviamo i dati e le variabili globali dell'eseguibile.

.rsrc = contiene risorse utilizzate dall'eseguibile che non sono parte dell'eseguibile stesso, come ad esempio l'icona o immagini.



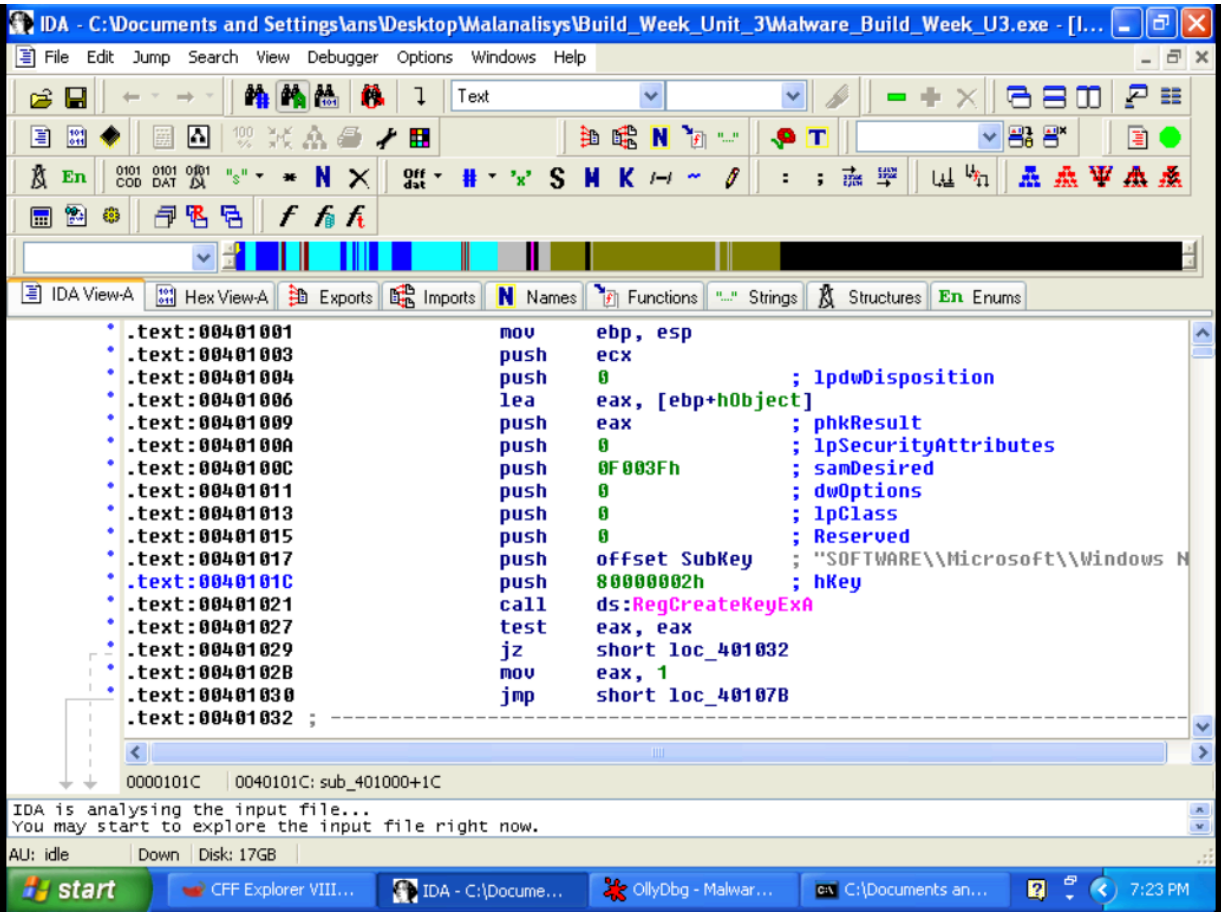
Qui sotto invece possiamo vedere le librerie importate sempre grazie a CFF Explorer. Queste sono **KERNEL32.DLL** e **ADVAPI32.DLL**. La prima contiene le



principali funzioni per interagire con il sistema operativo mentre la seconda ha le funzioni per interagire con servizi e i registri di sistema Windows.

Su queste poche informazioni possiamo già intuire che il malware potrebbe essere in grado di ottenere persistenza sul sistema operativo grazie alle funzioni **RegCreateKeyExA** e **RegSetValueExA** della libreria ADVAPI32.dll. Infatti queste permettono al malware di creare e modificare le chiavi di registro in modo da avviarsi automaticamente ad ogni avvio. Come ad esempio **Software\\Microsoft\\Windows\\CurrentVersion\\Run**.

Possiamo vedere inoltre che dentro a KERNEL32.dll ci sono funzioni tipiche di un dropper come **FindResource** o **LoadResource,** che permettono al malware di andare a cercare un altro programma malevolo contenuto al suo interno. Il programma potrebbe anche caricare dinamicamente altre funzioni che non vediamo grazie a **LoadLibrary** e **GetProcAddress**.
Ovviamente queste supposizioni andranno verificate con l'analisi dinamica.



Assembly x86 Analysis

- Alla locazione di memoria **00401021** abbiamo la funzione RegCreateKeyExA che crea una chiave di registro. Se la chiave già esiste allora andrà ad aprirla.

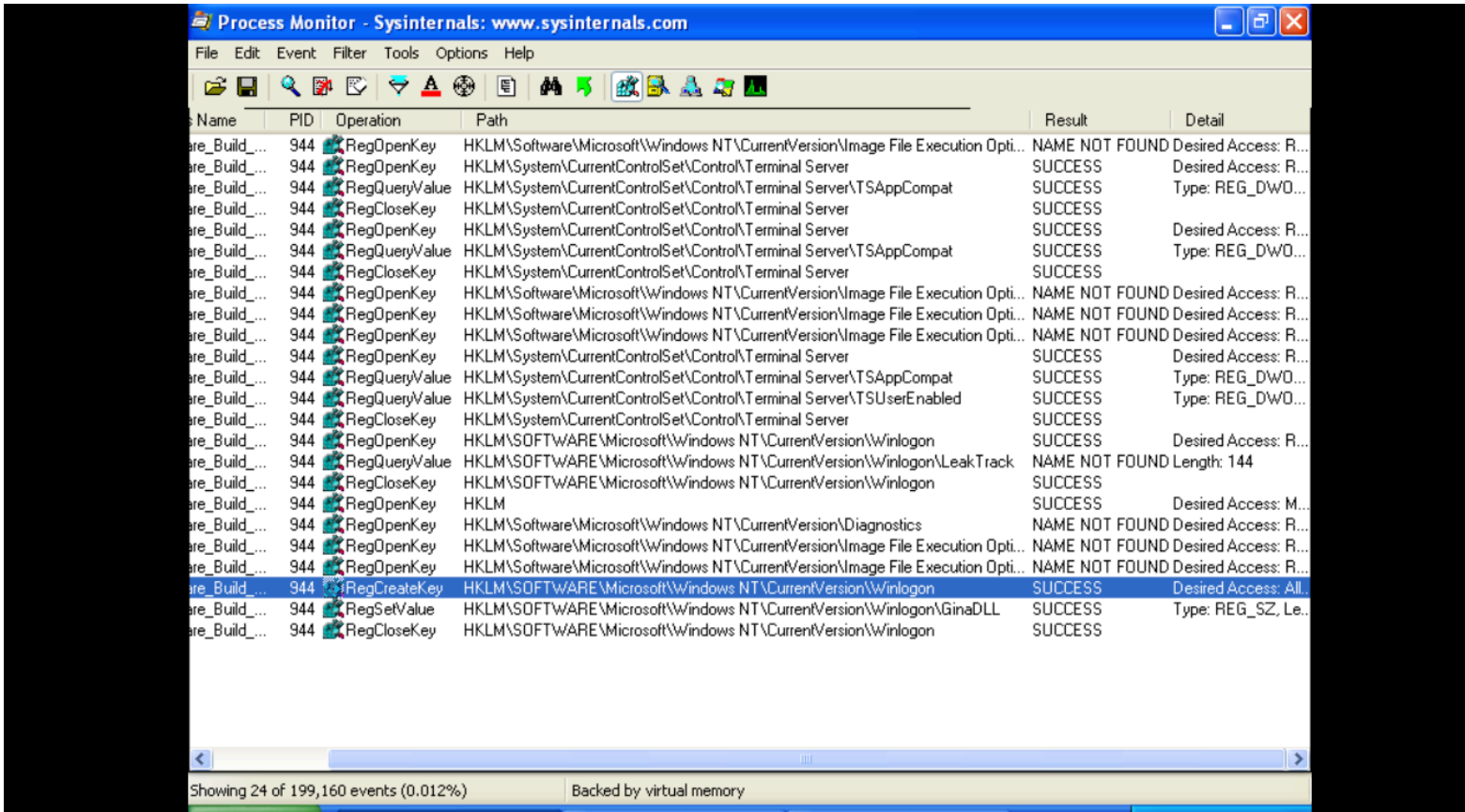
- I parametri passano alla funzione tramite i **push** nelle precedenti istruzioni (da **00401004** a **0040101C**). Questi sono, fra gli altri, **lpdwDisposition** (un puntatore che stabilisce se la chiave esiste o è stata creata), **samDesired** (una maschera che stabilisce i diritti d'accesso per creare la chiave), e così via.

- L'oggetto all'istruzione **00401017** rappresenta la subkey, cioè la sottochiave che la funzione andrà ad aprire o creare. Nel nostro caso si tratta di **HKLM\ \Software\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon** che serve per la configurazione del servizio Winlogon per tutti gli utenti.

- Fra le istruzioni **00401027** e **00401029** abbiamo un jump condizionale. L'istruzione **test** farà un AND su EAX per controllare se è 0. Se l'AND è 0 il **ZF** sarà 1 e quindi il salto sarà effettuato. Il costrutto in C potrebbe essere scritto così:

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

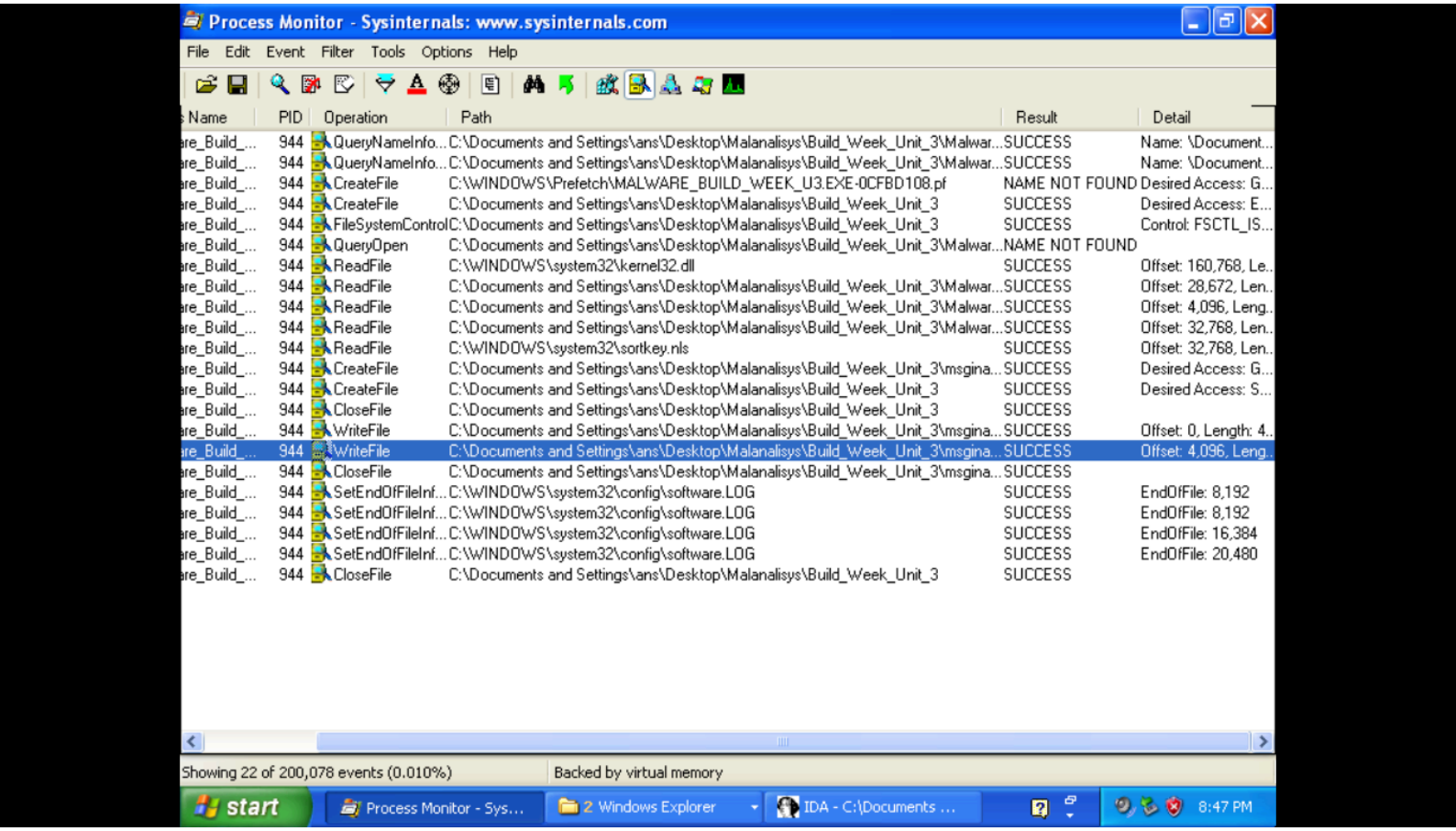
- Alla locazione **00401047** possiamo trovare la funzione RegSetValueExA che scriverà su una chiave un determinato valore. Il valore dell'oggetto ValueName sarà **GinaDLL**.



ANALISI DINAMICA

Dopo aver avviato il malware possiamo notare che, dentro alla cartella dove è contenuto, è stato creato un nuovo file chiamato **msgina32.dll.**

La chiave di registro che il malware apre è **HKLM\\Software\\Microsoft\ \Windows NT\\CurrentVersion\\Winlogon,** come abbiamo constatato nell'analisi



statica. A questa chiave assegna la sottochiave **GinaDLL.** Facendo doppio click vediamo che il percorso associato a questa sottochiave è proprio ~\msgina32.dll.

La funzione che crea il file msgina32.dll è **CreateFile** della libreria KERNEL32.dll.

A questo punto possiamo constatare che il malware in questione appartenga alla categoria dei **dropper**. Infatti sono presenti nel codice le funzioni **FindResource, LoadResource, LockResource** e **SizeOfResource,** tipiche di un malware che andrà ad estrarne un altro per salvarlo sul disco.
Il file "droppato" in questione è appunto **msgina32.dll**. Lo scopo di una DLL GINA è fornire procedure personalizzabili di identificazione e autenticazione dell'utente. Quindi il malware **ha creato persistenza** tramite la GinaDLL.

Riavviando il computer è possibile notare già una differente schermata di login rispetto al solito. Questo conferma la persistenza del malware tramite il servizio Winlogon.