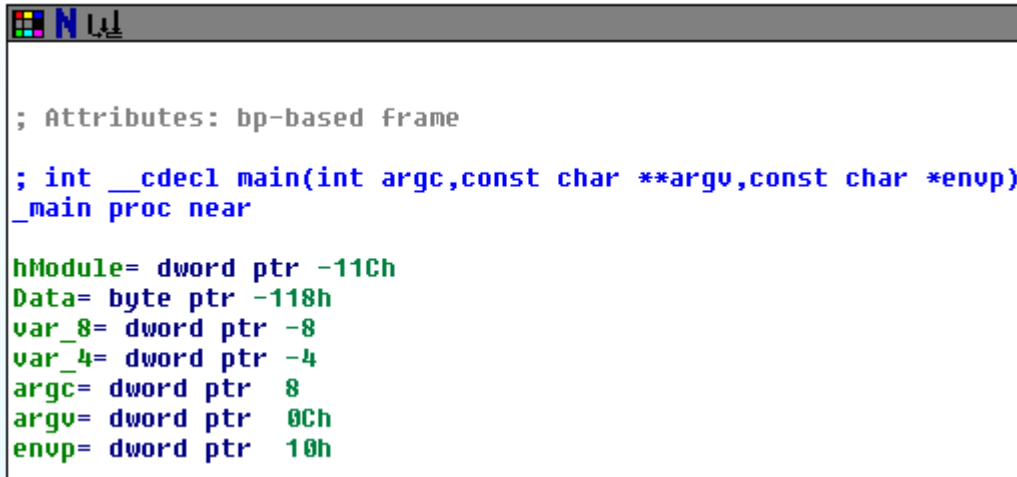


# Analisi Statica



```
; 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_8= dword ptr -8  
var_4= dword ptr -4  
argc= dword ptr 8  
argv= dword ptr 0Ch  
envp= dword ptr 10h
```

1) Nella funzione *main()* vengono passati tre parametri:

- **argc** (argument count): È un parametro in posizione 0 (primo parametro) e viene rappresentato da dword ptr 8, quindi occupa 4 byte.
- **argv** (argument vector): È un parametro in posizione 1 (secondo parametro) e viene rappresentato da dword ptr 0Ch, quindi occupa 4 byte.
- **envp** (environment pointer): È un parametro in posizione 2 (terzo parametro) e viene rappresentato da dword ptr 10h, quindi occupa 4 byte.

---

2) sono dichiarate 4 variabili nello stack frame della funzione:

- **hModule**: È una variabile di tipo dword (4 byte) che viene dichiarata come -11Ch rispetto all'indirizzo della base dello stack (ebp). Quindi occupa 4 byte di spazio nello stack.
  - **Data**: È una variabile di tipo byte (1 byte) che viene dichiarata come -118h rispetto all'indirizzo della base dello stack (ebp). Quindi occupa 1 byte di spazio nello stack.
  - **var\_8**: È una variabile di tipo dword (4 byte) che viene dichiarata come -8 rispetto all'indirizzo della base dello stack (ebp). Quindi occupa 4 byte di spazio nello stack.
  - **var\_4**: È una variabile di tipo dword (4 byte) che viene dichiarata come -4 rispetto all'indirizzo della base dello stack (ebp). Quindi occupa 4 byte di spazio nello stack.
-

3)

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	00003EAB	00008000	00003000	00008000	00000000	00000000	0000	0000	C0000040
.rsrc	00001A70	0000C000	00002000	0000B000	00000000	00000000	0000	0000	40000040

Sono presenti quattro sezioni, visibili con cff explorer

- **.rsrc**: include le risorse utilizzati dall'eseguibile che non vengono considerate parte di esso, come icone, immagini, menu e stringhe.
- **.rdata**: solitamente contiene le informazioni da importare ed esportare. Può inoltre salvare dai dati read-only (ossia che si può solo leggere) usati dal programma;

4) Vengono importate le librerie **Kernel32.dll** e **ADVAPI32.dll**

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

La libreria **Kernel32.dll** può far presupporre che il malware utilizzi funzioni per la gestione della memoria o funzioni per interagire con il sistema operativo

**ADVAPI32.dll** permette invece al malware di entrare nelle chiavi di registro

5)

00401003	51	PUSH ECX	pDisposition = NULL
00401004	6A 00	PUSH 0	pHandle
00401006	8D45 FC	LEA EAX, DWORD PTR SS:[EBP-4]	pSecurity = NULL
00401009	50	PUSH EAX	Access = KEY_ALL_ACCESS
0040100A	6A 00	PUSH 0	Options = REG_OPTION_NON_VOLATILE
0040100C	68 3F00F00	PUSH 0F003F	Class = NULL
00401011	6A 00	PUSH 0	Reserved = 0
00401013	6A 00	PUSH 0	Subkey = "SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon"
00401015	6A 00	PUSH 0	hKey = HKEY_LOCAL_MACHINE
00401017	68 54804000	PUSH Malware_.00408054	
0040101C	68 02000000	PUSH 80000002	
00401021	FF15 04704000	CALL DWORD PTR DS:[&ADVAPI32.RegCreateEx]	RegCreateKeyEx

Lo scopo della funzione all'indirizzo di memoria 00401021 è creare la chiave di registro

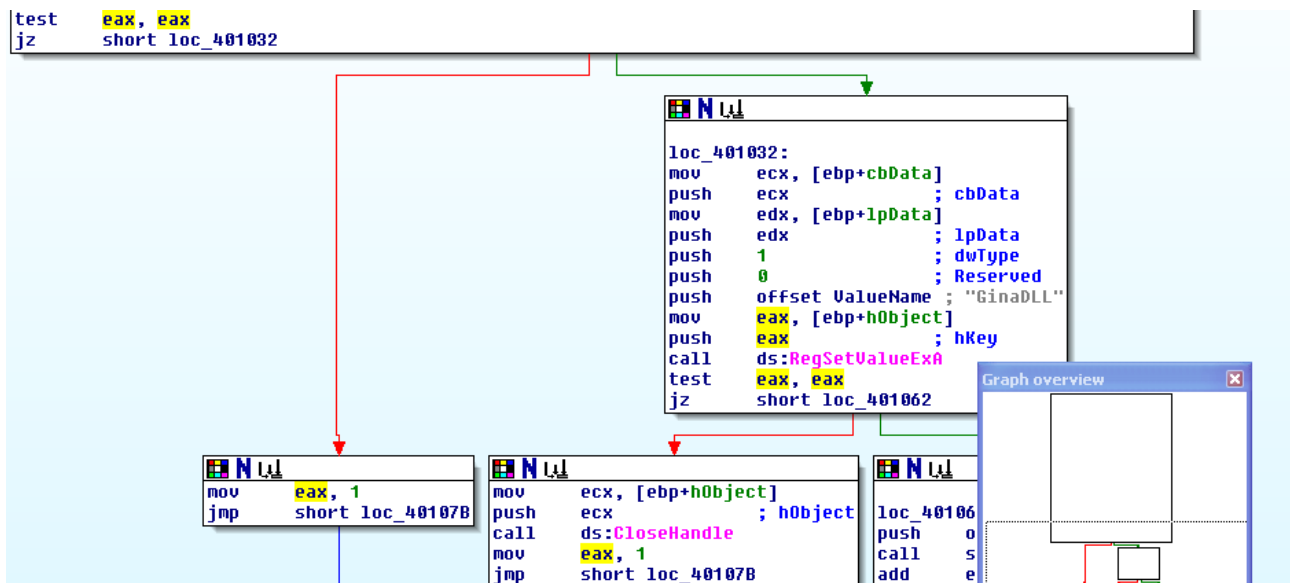
"SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon" e viene passata tramite i push che caricano i parametri

**.text:00401017**      **push**      **offset SubKey**      ; "SOFTWARE\Microsoft\Windows NT\CurrentVe"...

All'indirizzo 00401017 troviamo la chiave di registro che porta all'avvio automatico la dll fallata.

**.text:00401027**      **test**      **eax, eax**  
**.text:00401029**      **jz**      **short loc\_401032**

Il malware qui verifica se è già stato avviato. In caso negativo fa il salto all'indirizzo **loc\_401032** altrimenti si chiude, come visibile nel diagramma di flusso a seguito



Di seguito la traduzione del costrutto if in C

```

if (eax == 0){
    funct_401032();
}
else{
    eax = 1;
    funct_40107B();
}

```

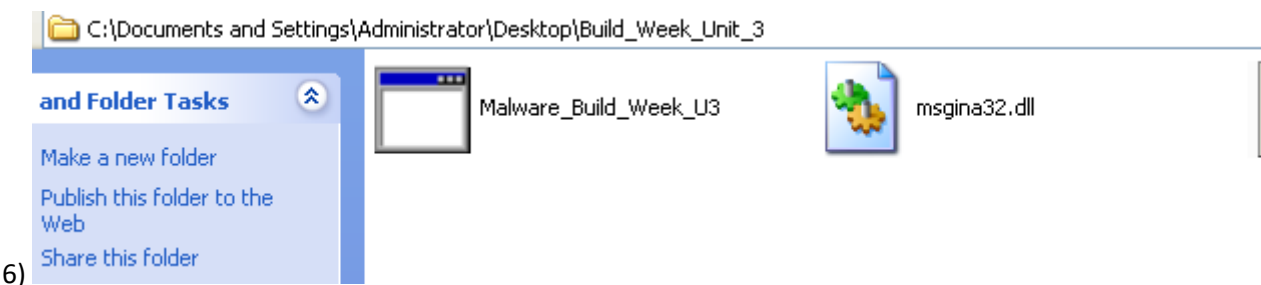
```

.text:0040103E      push    offset ValueName ; "GinaDLL"
.text:00401043      mov     eax, [ebp+hObject]
.text:00401046      push    eax                ; hKey
.text:00401047      call    ds:RegSetValueExA

```

All'indirizzo di memoria 00401047 il parametro ValueName ha valore "GinaDLL"

# Analisi Dinamica



all'interno della cartella dove era situato inizialmente il malware si è creato il file msgina32.dll, che è probabilmente una versione corrotta della DLL GINA la quale, citando le dispense di Microsoft: "The purpose of a *GINA DLL* is to provide customizable user identification and authentication procedures." Ovvero: "Ha lo scopo di fornire procedure di identificazione e autenticazione dell'utente personalizzabili"

7)

RegOpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\kernel32.dll	NAME NOT FOUND	Desired Access: Read
RegCreateKey	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon	SUCCESS	Desired Access: All Access
RegSetValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\GinaDLL	SUCCESS	Type: REG_SZ, Length: 520, Data

Viene creata la chiave di registro Winlogon, e gli viene dato il valore msgina32.dll che abbiamo trovato nella cartella del malware

Malware_Build_Week_U3.exe	1856	CreateFile	C:\Documents and Settings\Administrator\Local Settings\Temp\AAA4_AHP\UUMPA1.IX1
Malware_Build_Week_U3.exe	1856	CreateFile	C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\MSGINA32.DLL
Malware_Build_Week_U3.exe	1856	CreateFile	C:\WINDOWS\AppPatch\sysmain.sdb

La chiamata che modifica il contenuto della cartella è la CreateFile

**Conclusioni:** Il malware è probabilmente un **dropper** in quanto utilizza la sezione .rsrc e contiene al suo interno un logger che copia le credenziali di accesso