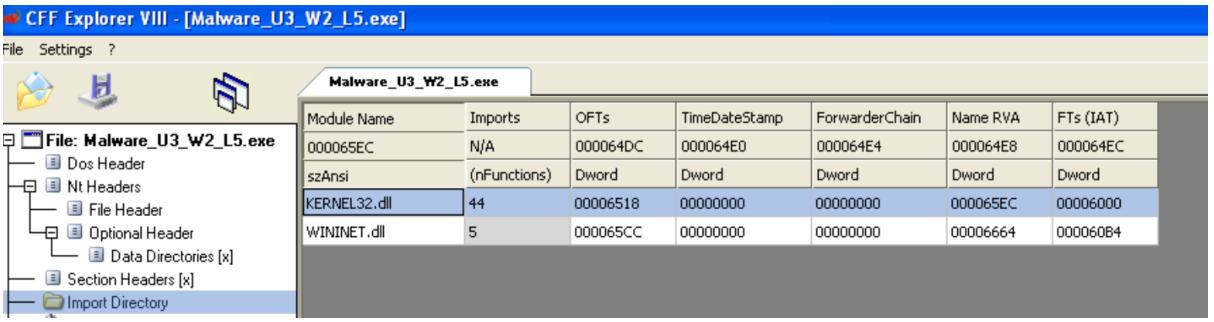
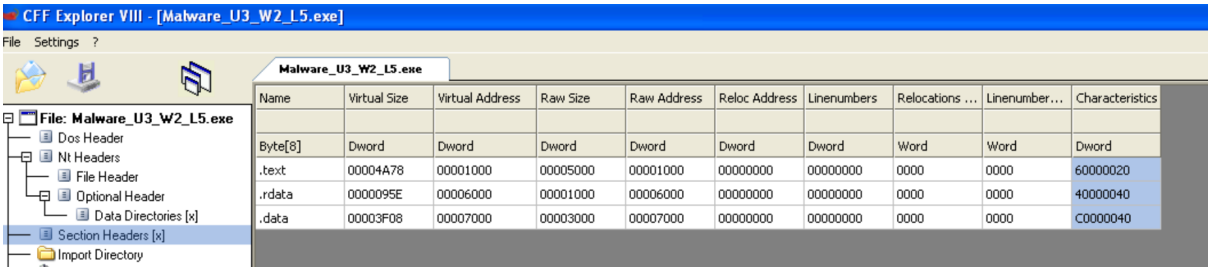
1) Librerie del Malware



Notiamo che il programma carica 2 librerie:
KERNEL32.dll con 44 funzioni, necessaria a comunicare con il sistema operativo
WININET.dll con 5 funzioni utilizzate principalmente per accedere a servizi di rete
quindi ci aspettiamo che il programma possa comunicare.

2)Sezioni



3)Costrutti

```
push    ebp
mov     ebp, esp
push    ecx
push    0                   ; dwReserved
push    0                   ; lpdwFlags
call    ds:InternetGetConnectedState
mov     [ebp+var_4], eax
cmp     [ebp+var_4], 0
jz      short loc_40102B
```

- in ROSSO possiamo vedere la creazione di un stack di memoria di grandezza indefinita.
- in BLU il codice ci suggerisce che il programma ha utlizzato 2 variabili per verificare con la funzione invocata in CALL la connesione ad internet della macchina.
-in VERDE è evidente che il programma abbia eseguito una condizione di tipo IF/ELSE data dalla coppia CMP-JZ (compare e jump)

```
push    offset aSuccessInterne ; "Success: Internet Connection\n"
call    sub_40117F
add     esp, 4
mov     eax, 1
jmp     short loc_40103A
```

-in ROSSO vediamo che il malware esegue la funzione PRINTF per stampare a video la frase di avvenuta connessione.
- con le altre istruzioni sposta e aggiunge valori ai registri esp e eax, infine salta a allo stack conclusivo.

```
loc_40102B:                             ; "Error 1.1: No Internet\n"
push    offset aError1_1NoInte
call    sub_40117F
add     esp, 4
xor     eax, eax
```

-in ROSSO possiamo intuire che la connessione ai servizi di rete non sia andata a buon fine, il programma crea un messaggio di errore e lo stampa a video con la funzione printf.
-in VERDE ripulisce il registro eax con xor per salvare memoria

```
loc_40103A:
mov     esp, ebp
pop     ebp
retn
sub_401000 endp
```

-con questa porzione di codice il programma ripulisce lo stack.

4)Ipotizzare il funzionamento
Se vogliamo stimare quale funzione svolge il codice analizzato possiamo presupporre che esso vada a testare la connessione ai servizi di rete della macchina attaccata. Con un ciclo if compara i dati delle variabili della funzione stabilita così da compiere l'adeguato salto di memoria seconda del risultato. Infine ripulisce lo stack.