

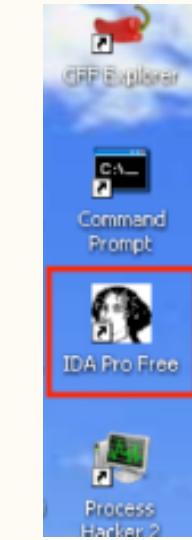
# Analisi statica con IDA

---

Emanuele Di Stefano

## Svolgimento dell'Esercizio

In questo esercizio, utilizzeremo IDA Pro per eseguire un'analisi statica sul malware denominato "Malware\_U3\_W3\_L2". Di seguito viene presentato un approccio leggermente diverso rispetto al precedente, con una descrizione dettagliata dei passaggi necessari per rispondere ai quesiti richiesti.



### 1. Individuazione dell'indirizzo della funzione DLLMain

- Passaggio 1: Avvia IDA Pro e carica il file eseguibile del malware "Malware\_U3\_W3\_L2".
- Passaggio 2: Una volta caricato il file, nella vista "Names Window" di IDA Pro, cerca il nome della funzione DLLMain. Questa funzione è generalmente presente in molti malware poiché rappresenta il punto di ingresso per i moduli DLL.
- Passaggio 3: Dopo aver individuato la funzione DLLMain, prendi nota dell'indirizzo esadecimale visualizzato accanto al nome della funzione. Questo indirizzo rappresenta il punto esatto in cui il codice della funzione DLLMain inizia.

The screenshot shows the IDA Pro interface. On the left, the 'Names' window displays a list of function names and their addresses. The function 'DLLMain(x,x)' is highlighted with a blue selection bar and has its address '1000D02E' visible. On the right, the assembly code view shows the beginning of the function. At the bottom left, the toolbar includes a file icon (highlighted with a red box and an arrow pointing to it) and other standard file operations like Open, Save, and Exit.

Name	Address	P
PSLIST	10007025	P
nullsub_1	1000706F	
nullsub_2	1000707C	
StartEXS	10007ECB	P
HandlerProc	1000C9DF	
ServiceMain	1000CF30	P
DLLMain(x,x)	1000D02E	
InstallRT	1000D847	P
InstallSA	1000DEC1	P
InstallSB	1000E892	P
UninstallSA	1000EA05	P
UninstallSB	1000F138	P
InstallNIDT	1000FAE6	D

## 2. Individuazione della funzione gethostbyname e relativo indirizzo di importazione

Passaggio 1: Vai alla scheda Imports di IDA Pro, che puoi trovare nel menu View sotto Open Subviews > Imports.

Passaggio 2: Nella lista degli imports, cerca la funzione gethostbyname. Questa funzione è spesso utilizzata dai malware per risolvere i nomi di dominio in indirizzi IP.

Passaggio 3: Una volta trovata la funzione, prendi nota dell'indirizzo di importazione che si trova accanto al nome della funzione.

**Descrizione della Funzione:** gethostbyname viene utilizzata per convertire un nome di host (es. "example.com") in un indirizzo IP. Questa funzione è cruciale per stabilire connessioni di rete verso server remoti.

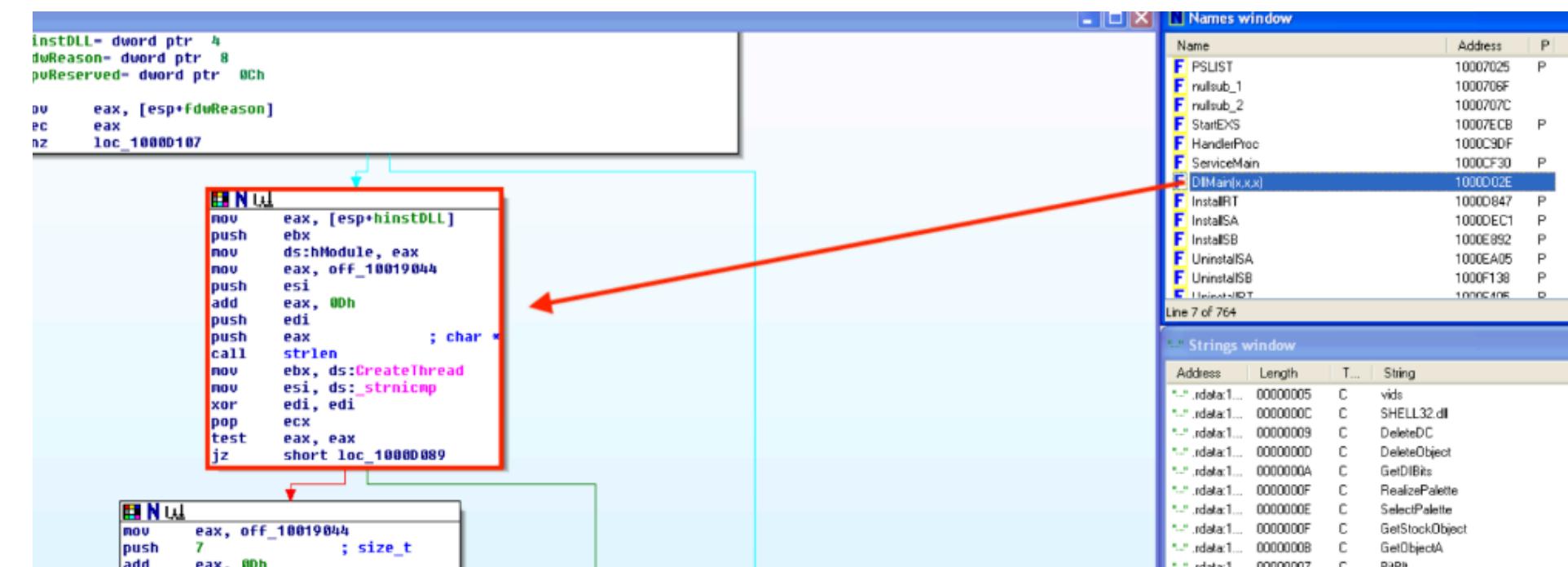
10016300	strchr	MSVCRT
100162F0	strcat	MSVCRT
100162F4	snprintf	MSVCRT
100163F8	23 socket	WS2_32
100163E8	21 setsockopt	WS2_32
100163D8	19 send	WS2_32
100163C4	18 select	WS2_32
100163D4	16 recv	WS2_32
100162A8	printf	MSVCRT
100163E0	15 ntohs	WS2_32
100163B8	mouse_event	USER32
100162D4	memset	MSVCRT
100162C8	memcpy	MSVCRT
100162AC	memcmp	MSVCRT
10016264	malloc	MSVCRT
1001638C	keybd_event	USER32
1001624C	isdigit	MSVCRT
100163D0	12 inet_ntoa	WS2_32
100163C8	11 inet_addr	WS2_32
100163E4	9 htons	WS2_32
100163CC	52 gethostbyname	WS2_32
100162A0	fwrite	MSVCRT
10016278	ftell	MSVCRT
100162D8	fseek	MSVCRT

### 3. Numero di variabili locali alla locazione di memoria 0x10001656

Passaggio 1: Usa la funzione "Jump to Address" (presente nel menu Jump) per navigare direttamente all'indirizzo 0x10001656.

Passaggio 2: Analizza il prologo della funzione alla locazione indicata. Le variabili locali sono solitamente allocate tramite istruzioni come sub esp, ... o tramite operazioni di push sui registri.

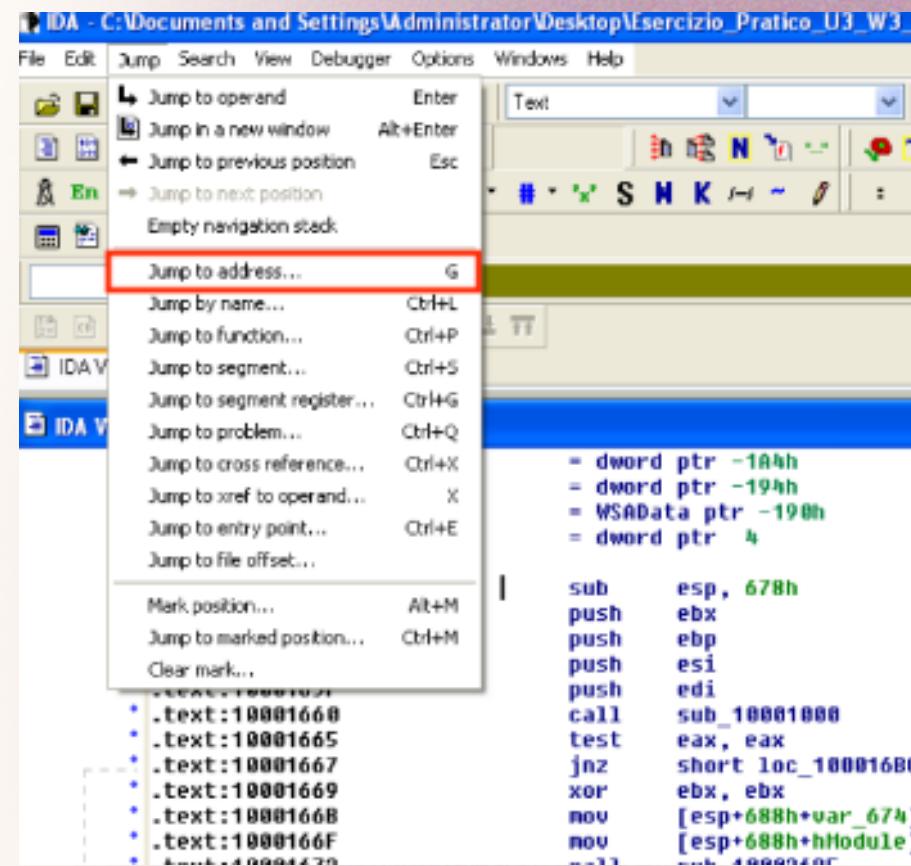
Passaggio 3: Conta il numero totale di variabili locali allocate nella funzione a partire da questo indirizzo.



#### 4. Numero di parametri della funzione alla locazione di memoria Ox10001656

Passaggio 1: Nella stessa locazione di memoria Ox10001656, osserva il prologo della funzione per identificare il numero di parametri. Questi possono essere passati tramite registri o stack, come tramite push.

Passaggio 2: Identifica e conta il numero di parametri passati alla funzione, che possono includere variabili di ingresso critiche per il funzionamento del malware.



```
.text:10001656 arg_0        = dword ptr  4
.text:10001656
sub    esp, 678h
push   ebx
push   ebp
push   esi
push   edi
call   sub_10001600
test   eax, eax
jnz   short loc_100016BC
xor    ebx, ebx
mov    [esp+688h+var_674], ebx
mov    [esp+688h+hModule], ebx
call   sub_10003695
mov    dword_1008E5C4, eax
call   sub_100036C3
push   3498h               ; dwMilliseconds
mov    dword_1008E5C8, eax
call   ds:sleep
call   sub_100110FF
lea    eax, [esp+688h+WSAData]
push   eax
push   202h               ; lpWSAData
push   202h               ; wVersionRequested
call   ds:WSAStartup
cmp    eax, ebx
jz    short loc_100016CB
push   eax
push   offset aWsastartupError ; "WSAStartup() error: %d\n"
call   ds:_imp_printf
pop    ecx
pop    ecx
loc_100016BC:           ; CODE XREF: sub_10001656+11↑j
.pop   edi
```

## 5. Considerazioni Generali sul Malware

Durante l'analisi del malware, possiamo fare alcune osservazioni generali:

- persistenza: Il malware potrebbe implementare tecniche di persistenza modificando le chiavi del registro di Windows o installando se stesso come servizio per riavviarsi automaticamente.
- Evasione: potrebbe includere tecniche di evasione per evitare il rilevamento da parte degli strumenti di analisi o delle soluzioni antivirus, come l'uso di tecniche anti-debugging.
- Comunicazione di Rete: l'uso di gethostbyname indica che il malware probabilmente si connette a un server remoto per ricevere comandi o inviare dati.
- Offuscamento: il codice potrebbe essere offuscato per rendere più difficile l'analisi statica, utilizzando stringhe criptate o chiamate indirette alle funzioni.