REPORT 17/01/2023

FACCILONGO DOMENICO

ANALISI DINAMICA AVANZATA CON IDA

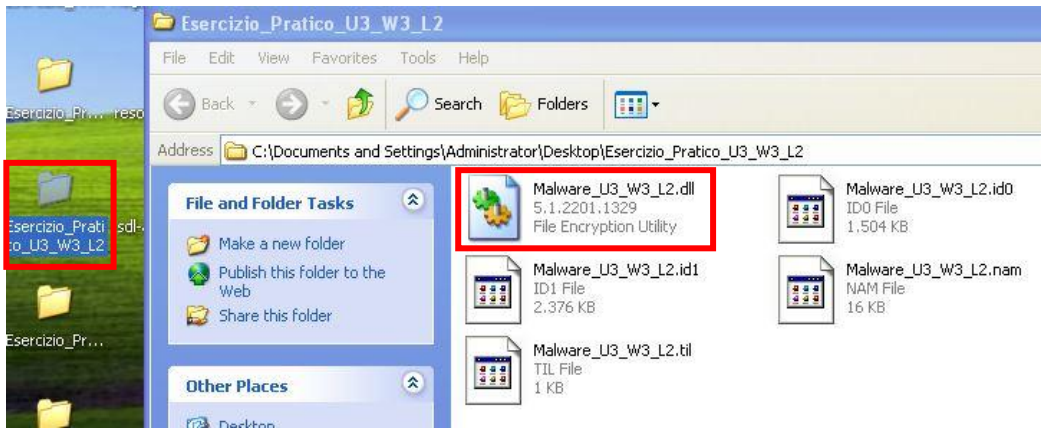
TASK

Lo scopo dell'esercizio di oggi è di acquisire esperienza con IDA, un tool fondamentale per l'analisi statica.
A tal proposito, con riferimento al malware chiamato «**Malware_U3_W3_L2**» presente all'interno della cartella «**Esercizio_Pratico_U3_W3_L2**» sul Desktop della macchina virtuale dedicata all'analisi dei malware, rispondere ai seguenti quesiti, utilizzando IDA Pro.

1. Individuare **l'indirizzo** della funzione **DLLMain** (così com'è, in esadecimale)
2. Dalla scheda «imports» individuare la funzione «**gethostbyname**». Qual è l'indirizzo dell'import?
3. Quante sono le variabili locali della **funzione** alla locazione di memoria 0x10001656?
4. Quanti sono, invece, i parametri della funzione sopra?
5. Inserire altre considerazioni macro livello sul malware (comportamento)

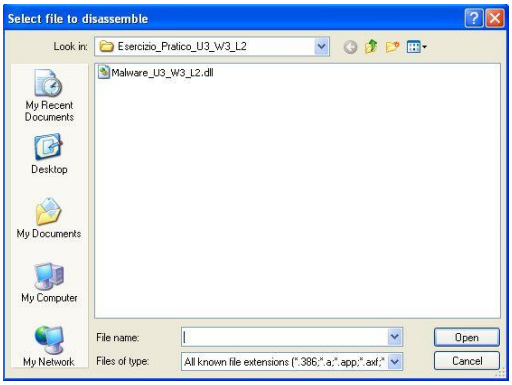
ANALISI E VALUTAZIONI

Come da task dobbiamo andare ad analizzare il malware trovatosi nella virtual box *Malware Analysis_Final* sul desktop nella cartella *Eserizio_Pratico_U3_W3_L2* come da immagine seguente:

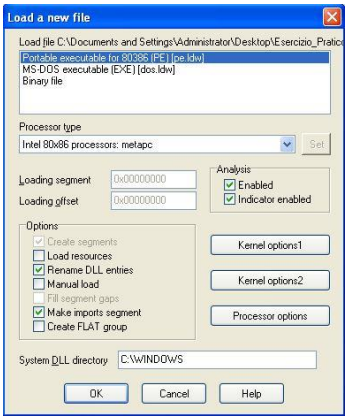


Successivamente andiamo ad aprire il tool fondamentale per l'analisi dinamica avanzata del nostro malware. Il tool **IDA Pro Free** sarà locato sul desktop, e all'apertura avremo la finestra come di seguito, da cui per aprire il file basterà cliccare sull'icona segnata:

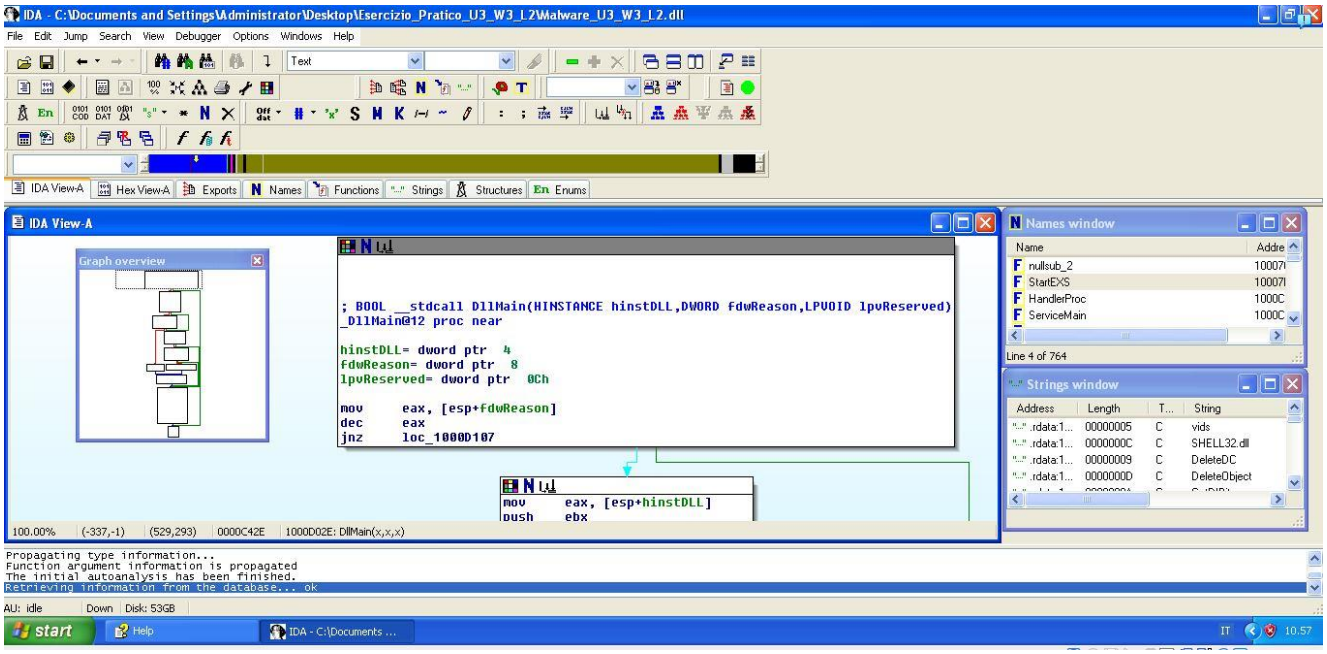
Successivamente dopo aver cliccato, ci apparirà la finestra per selezionare il file interessato, andiamo sul path e clicchiamo su *Open*.



All'apertura del file avremo una nuova finestra che ci spiega tutte le funzioni sul tool che utilizza, eventualmente sulla base della nostra ricerca possiamo scegliere i settaggi che preferiamo, dopo aver effettuato le modifiche eventuali proseguiamo clicclando su *OK*:



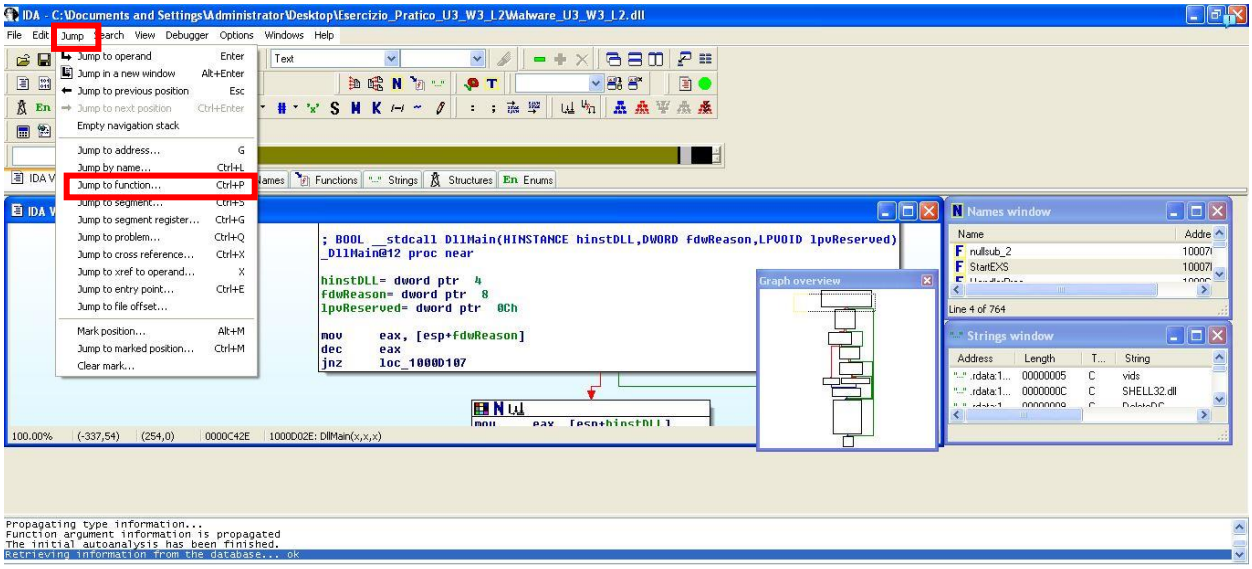
La grafica che ne viene fuori sarà di questo tipo:



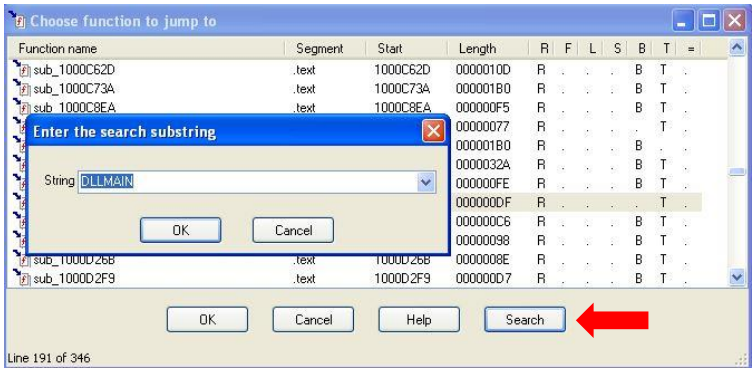
Esaminiamo ora i vari punti richiesti nelle task:

1. INDIVIDUARE L'INDIRIZZO DELLA FUNZIONE DLLMain

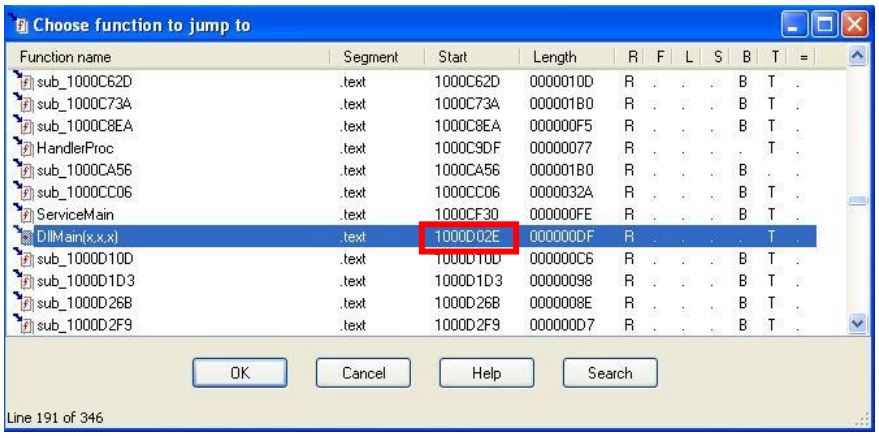
Al fine di soddisfare il primo punto, andiamo nella sezione jump (come indicato in figura) e selezioniamo il comando *jump to function*:



All'apertura avremo la finestra come in figura, selezioniamo *search* e digitiamo il nome della funzione di cui abbiamo bisogno, ovvero **DLLMain**:

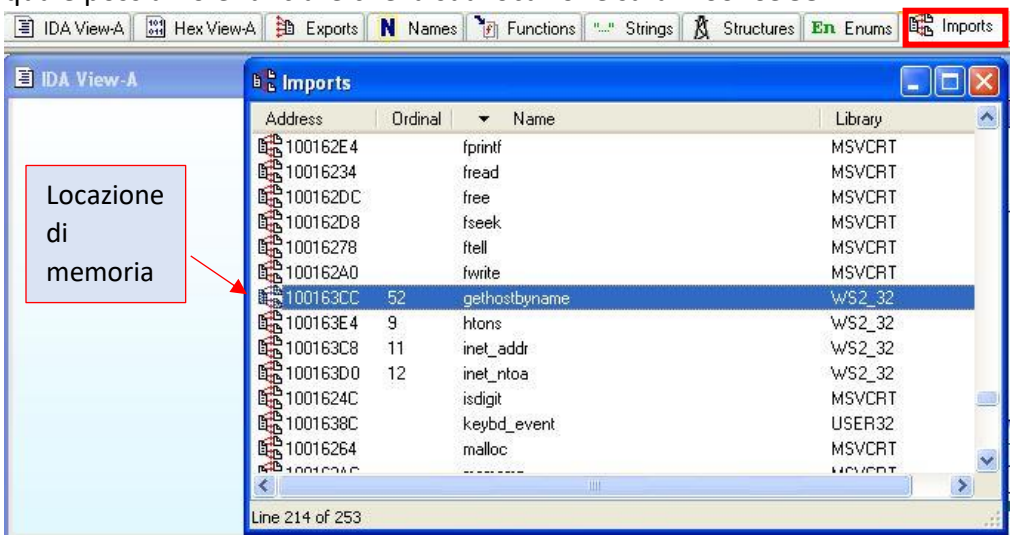


Il risultato ci porterà direttamente alla funzione cercata da cui possiamo stabilire che l'indirizzo di memoria ad essa associato è: **1000D02E**



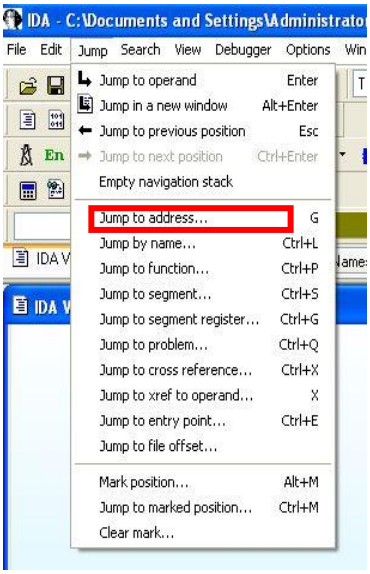
2. **INDICARE LA LOCAZIONE DELLA FUNZIONE <<gethostbyname>>**

In tal caso invece possiamo andare nella sezione *imports* (indicato in figura), selezioniamo Name in modo da mettere in ordine alfabetico e andiamo a cercarci la funzione interessata, dal quale possiamo enunciare che la sua locazione sarà: **100163CC**

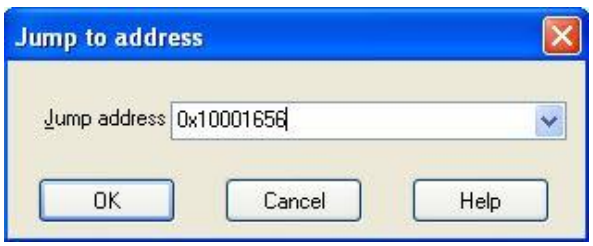


3. **INDICARE LE VARIABILI LOCALI DELLA FUNZIONE ALLA LOCAZIONE 0x10001656**

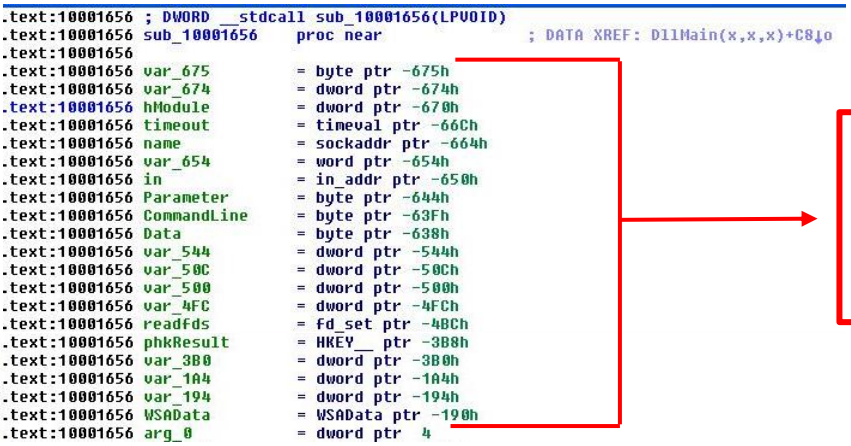
Come primo passo cerchiamo la locazione di memoria che abbiamo come dato dal comando *jump to address…*



Da cui alla nuova finestra che ci appare inseriamo la locazione:



Il risultato ottenuto ci porta direttamente alla locazione cercata, dal quale le variabili locali della funzione **Subroutine (sub_10001656)** sono:



In generale le Var locali sono ad un offset negativo rispetto al registro EBP. Per un totale di 20 variabili all'interno della funzione.

4. INDICARE I PARAMETRI DELLA FUNZIONE PRECEDENTE

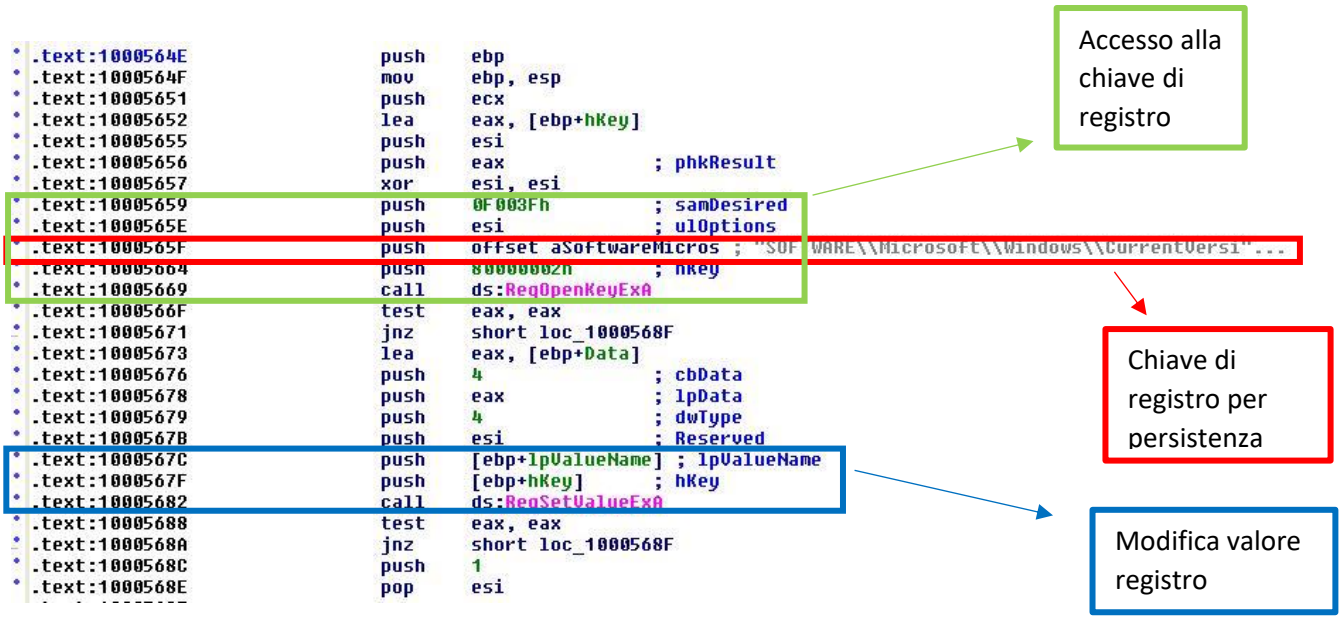
Prendendo in considerazione la foto precedente l'unico parametro che ritroviamo è l'ultimo, ovvero:

```
.text:10001656 arg_0           = dword ptr  4
```

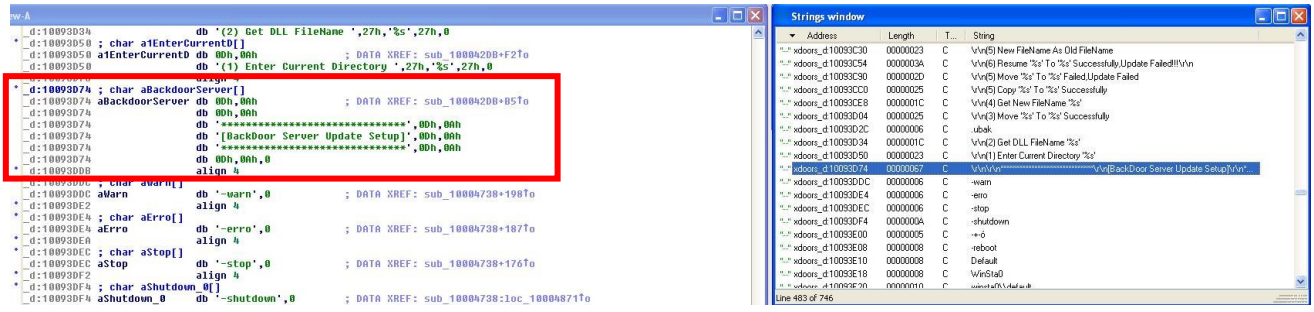
In generale i parametri si trovano si trovano ad un offset (differenza rispetto ad un valore di riferimento) positivo rispetto ad EBP.

5. INDICARE IL COMPORTAMENTO DEL MALWARE

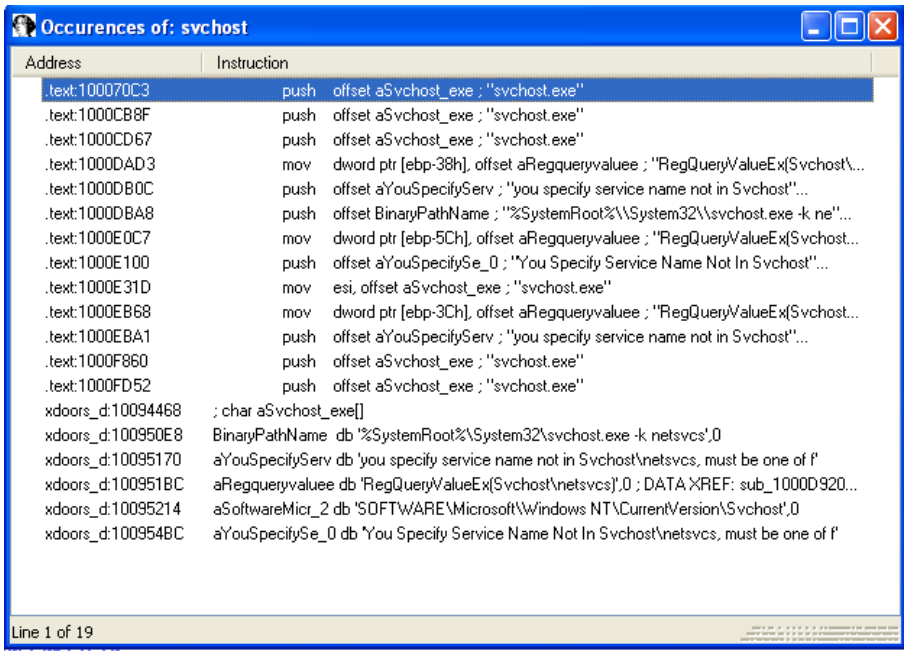
In conclusione andiamo ad ipotizzare il funzionamento del malware. Come prima cosa possiamo dedurre che il file ottiene la **persistenza**, in quanto ritroviamo le funzioni delle chiavi di registro in cui il file eseguibile va ad aprirle e a modificarle, così come ritroviamo anche la chiave di registro utilizzata dal malware per ottenere la persistenza (vedi figura):



Successivamente possiamo identificare anche che il malware funge da backdoor, infatti cercando in *String Windows* la dicitura **backdoor** quello che ne viene fuori è proprio la sezione di dato in cui espressamente dice "**Backdoor Server Update Setup**":



Inoltre possiamo dire che il malware crea un processo in **system32/svchost** dove andrà a *nascondersi*:



Come ultima ipotesi possiamo dire che il malware crea anche una **remote shell** dal quale ottiene informazioni della macchina vittima, di fatti, da immagine seguente si può evincere come il malware effettua comando come ad esempio ipconfig: