

[BUILD WEEK 3 – GIORNO 4]

ANALISI DINAMICA BASICA

Analisi del malware *Malware_Build_Week_U3.exe*

Tasks:

1. Preparazione dell'ambiente di test sulla VM ed esecuzione del malware
2. Rilevazione di eventuali modifiche all'interno della cartella di origine dell'eseguibile in seguito all'esecuzione del malware
3. Analisi dell'interazione del malware con il **registro**: identificazione della chiave di registro creata e del valore ad essa associato
4. Analisi dell'interazione del malware con il **file system**: identificazione della chiamata di sistema responsabile della modifica del contenuto della cartella di origine dell'eseguibile
5. Descrizione del comportamento del malware in base alle informazioni raccolte tramite analisi statica e dinamica

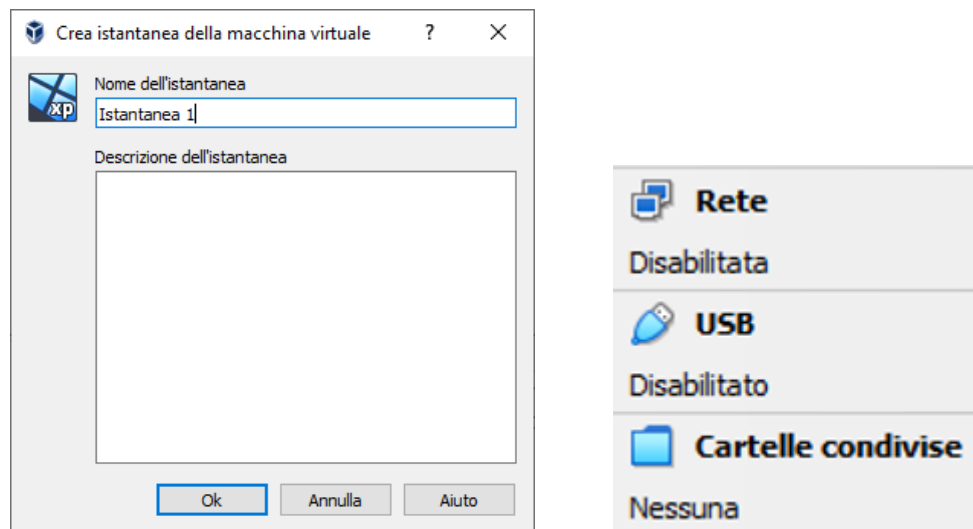
1. Preparazione dell'ambiente di test sulla VM ed esecuzione del malware

Le attività odierne sono incentrate sull'analisi dinamica basica del file eseguibile di test **Malware_Build_Week_U3.exe**.

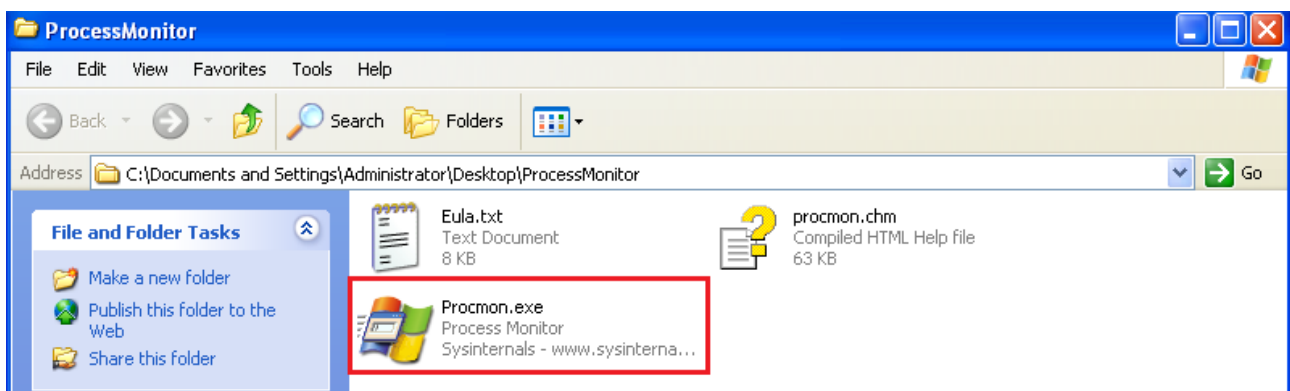
L'**analisi dinamica basica** comprende una gamma di attività di analisi che presuppongono l'**esecuzione del malware** in ambiente dedicato e protetto. È generalmente effettuata dopo l'analisi statica basica, per sopperire ai limiti di quest'ultima (infatti, al contrario dell'analisi statica, l'analisi dinamica permette di osservare in maniera diretta le funzionalità di un malware in esecuzione su un sistema) e confermare o smentire le ipotesi formulate durante l'analisi statica in merito al funzionamento del malware.

A monte delle operazioni di analisi, è utile creare un'istantanea della macchina virtuale di test (*Malware Analysis_Final* – OS Windows XP SP3) per poter eventualmente ripristinare le funzionalità della stessa in caso di problemi generati dall'esecuzione del malware; altrettanto importante è

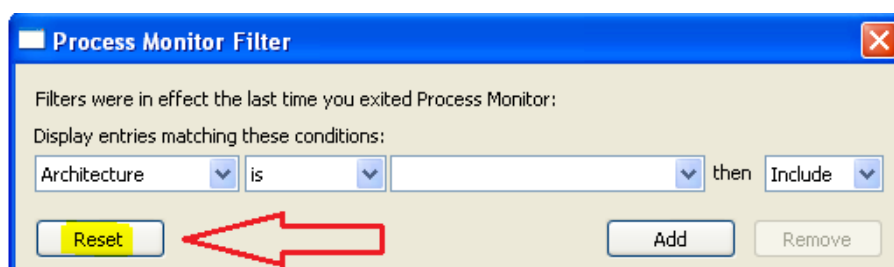
isolare l'ambiente di test disabilitando le schede di rete, la connettività USB ed eventuali cartelle condivise:



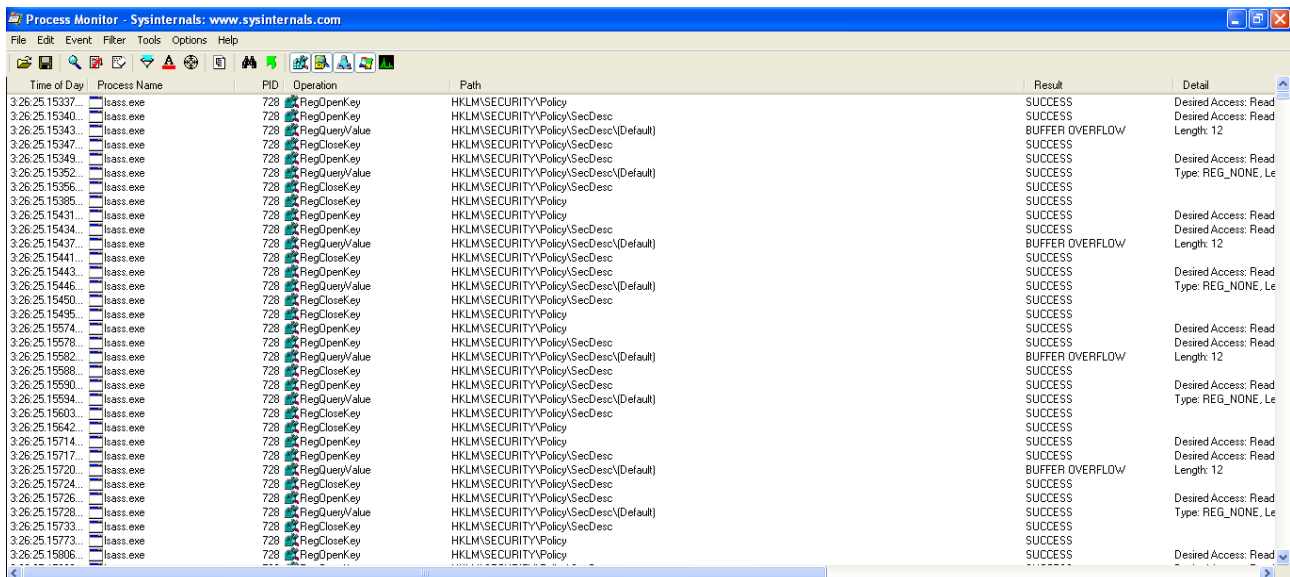
Adesso possiamo avviare la VM. Per le attività di analisi odierne, ci serviremo di **Process Monitor**: si tratta di un tool per sistemi operativi Windows che permette di monitorare i processi attivi, il file system, il registro di Windows e l'attività di rete.



Una volta avviato il tool, ci accertiamo che non sia attivo alcun filtro cliccando sul tasto "Reset" per avere una panoramica completa della situazione iniziale:



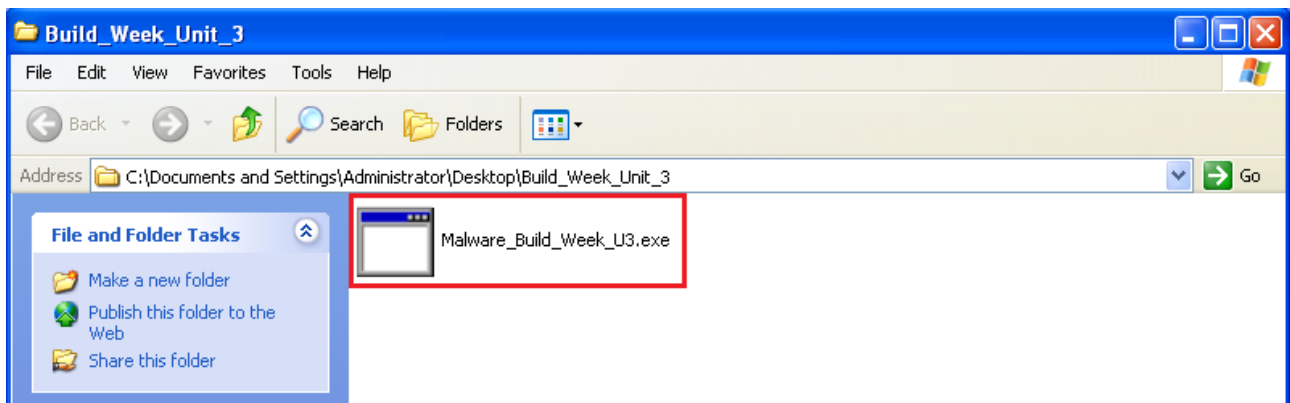
Confermiamo la scelta con “Apply” ed “OK” ed il software inizia la cattura in tempo reale degli eventi relativi ai parametri di analisi già citati:



The screenshot shows the Process Monitor application window with a table of registry operations. The table has columns for Time of Day, Process Name, PID, Operation, Path, Result, and Detail. The operations are performed by lsass.exe (PID 728) and involve opening, querying, and closing registry keys and values under the path HKLM\SECURITY\Policy\SecDesc\{Default}.

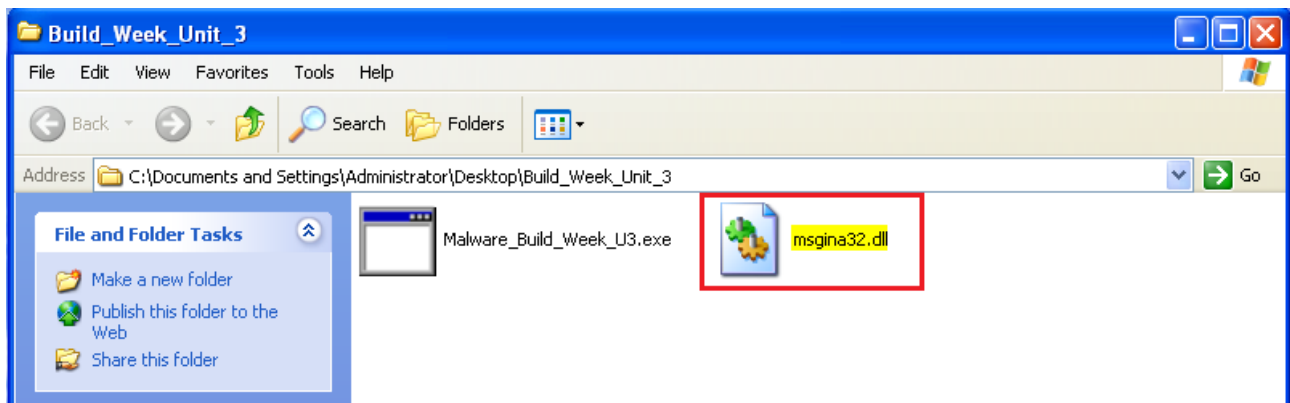
| Time of Day | Process Name | PID | Operation | Path | Result | Detail |
|------------------|--------------|-----|---------------|--|-----------------|------------------------------------|
| 3:26:25.15337... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy | SUCCESS | |
| 3:26:25.15340... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: Read |
| 3:26:25.15343... | lsass.exe | 728 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\{Default} | BUFFER OVERFLOW | Desired Access: Read Length: 12 |
| 3:26:25.15347... | lsass.exe | 728 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 3:26:25.15349... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: Read |
| 3:26:25.15352... | lsass.exe | 728 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\{Default} | SUCCESS | Type: REG_NONE, Le |
| 3:26:25.15356... | lsass.exe | 728 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 3:26:25.15385... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy | SUCCESS | |
| 3:26:25.15431... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: Read |
| 3:26:25.15434... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: Read |
| 3:26:25.15437... | lsass.exe | 728 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\{Default} | BUFFER OVERFLOW | Desired Access: Read Length: 12 |
| 3:26:25.15441... | lsass.exe | 728 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 3:26:25.15443... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: Read |
| 3:26:25.15446... | lsass.exe | 728 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\{Default} | SUCCESS | Type: REG_NONE, Le |
| 3:26:25.15450... | lsass.exe | 728 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 3:26:25.15495... | lsass.exe | 728 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 3:26:25.15574... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy | SUCCESS | |
| 3:26:25.15578... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: Read |
| 3:26:25.15582... | lsass.exe | 728 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\{Default} | BUFFER OVERFLOW | Desired Access: Read Length: 12 |
| 3:26:25.15590... | lsass.exe | 728 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 3:26:25.15590... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: Read |
| 3:26:25.15594... | lsass.exe | 728 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\{Default} | SUCCESS | Type: REG_NONE, Le |
| 3:26:25.15603... | lsass.exe | 728 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 3:26:25.15642... | lsass.exe | 728 | RegCloseKey | HKLM\SECURITY\Policy | SUCCESS | |
| 3:26:25.15714... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy | SUCCESS | Desired Access: Read |
| 3:26:25.15717... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: Read |
| 3:26:25.15720... | lsass.exe | 728 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\{Default} | BUFFER OVERFLOW | Desired Access: Read Length: 12 |
| 3:26:25.15724... | lsass.exe | 728 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 3:26:25.15726... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy\SecDesc | SUCCESS | Desired Access: Read |
| 3:26:25.15728... | lsass.exe | 728 | RegQueryValue | HKLM\SECURITY\Policy\SecDesc\{Default} | SUCCESS | Type: REG_NONE, Le |
| 3:26:25.15733... | lsass.exe | 728 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 3:26:25.15773... | lsass.exe | 728 | RegCloseKey | HKLM\SECURITY\Policy\SecDesc | SUCCESS | |
| 3:26:25.15806... | lsass.exe | 728 | RegOpenKey... | HKLM\SECURITY\Policy | SUCCESS | Desired Access: Read |

Adesso possiamo eseguire il malware da analizzare. Lo troviamo all’interno del path **C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3**. Lo avviamo facendo doppio click sull'icona dell’eseguiabile.



2. Rilevazione di eventuali modifiche all’interno della cartella di origine dell’eseguiabile in seguito all’esecuzione del malware

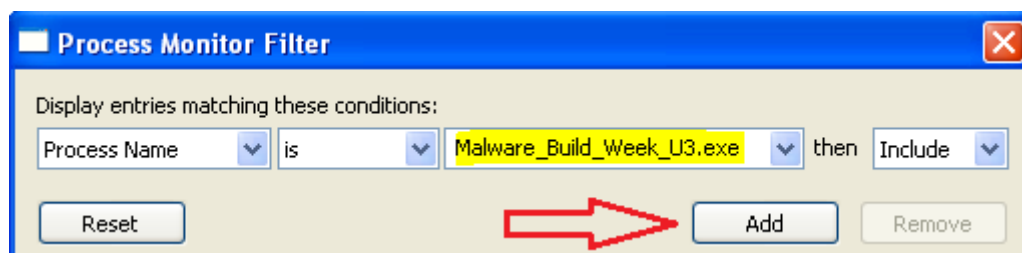
In seguito all’avvio del malware, notiamo che all’interno della cartella di origine dello stesso viene creato un nuovo file dal nome **msgina32.dll**:



Quanto appena visto conferma le ipotesi formulate durante l'analisi statica: il rilevamento delle chiamate di funzione **FindResource**, **LoadResource**, **LockResource** e **SizeofResource** importate dalla libreria KERNEL32.dll indica effettivamente che l'eseguibile analizzato è un **dropper**, ossia un programma malevolo che contiene al suo interno (specificamente, nella sezione "risorse" .rsrc dell'header del formato PE) un malware. Nel momento in cui viene eseguito, un dropper estrae il malware che contiene per avviarlo oppure salvarlo sul disco – in questo caso nello stesso path in cui si trova l'eseguibile.

3. Analisi dell'interazione del malware con il **registro**: identificazione della chiave di registro creata e del valore ad essa associato

Per analizzare le modifiche apportate dal malware al sistema, filtriamo i risultati ottenuti dalla cattura di Process Monitor nel seguente modo:



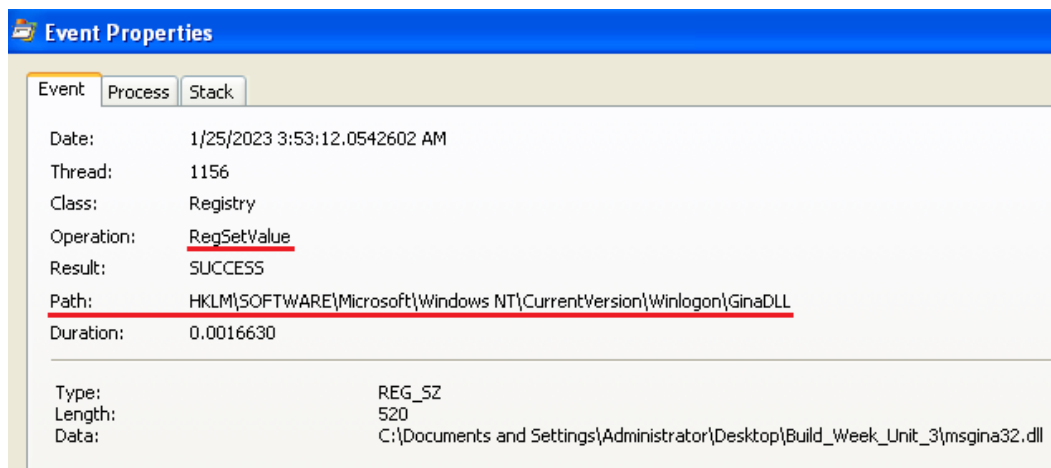
In particolare, vogliamo filtrare ulteriormente i risultati in modo da mostrare solamente gli eventi relativi all'interazione del malware con il registro di Windows. Le **chiavi di registro** sono variabili di configurazione di sistema: di conseguenza è importante, in fase di analisi, conoscere le modifiche eventualmente apportate ad esse da parte di un malware, per capire quali configurazioni sono state alterate. I valori delle chiavi rappresentano tutto ciò che viene caricato all'avvio del sistema.

Impostiamo dunque il filtro sulla sezione *Show Registry Activity* →



Andiamo dunque ad analizzare gli eventi in base al filtro appena inserito e notiamo la seguente situazione:

| | | | |
|---------------------------|------|--------------|---|
| Malware_Build_Week_U3.... | 1868 | RegOpenKey | HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\kernel32.dll |
| Malware_Build_Week_U3.... | 1868 | RegCreateKey | HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon |
| Malware_Build_Week_U3.... | 1868 | RegSetValue | HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\GinaDLL |
| Malware_Build_Week_U3.... | 1868 | RegCloseKey | HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon |

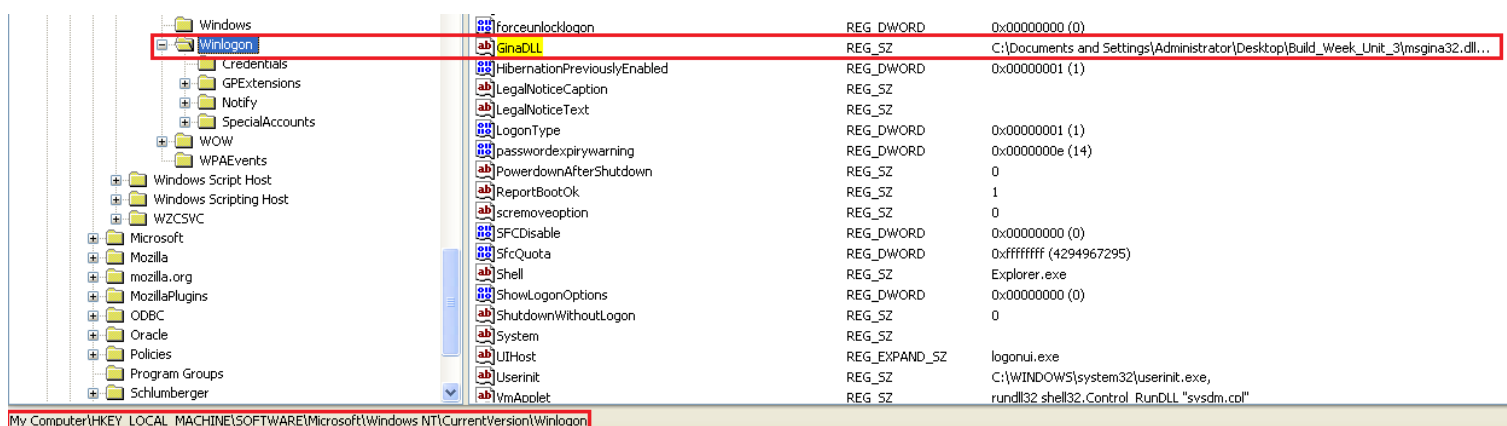


Come si può notare, tramite le chiamate di funzione **RegCreateKey** e **RegSetValue** il malware ha generato una chiave di registro all'interno del percorso **Software\Microsoft\Windows NT\CurrentVersion\Winlogon** e le ha successivamente assegnato il valore di "GinaDLL". Abbiamo verificato questa informazione anche servendoci di altri tool:

OllyDBG


| | | | |
|----------|---------------|--|-----------------------|
| 00401035 | 51 | PUSH ECX | BufSize |
| 00401036 | 8B55 08 | MOV EDX, DWORD PTR SS:[EBP+8] | Buffer |
| 00401039 | 52 | PUSH EDX | ValueType = REG_SZ |
| 0040103A | 6A 01 | PUSH 1 | Reserved = 0 |
| 0040103C | 6A 00 | PUSH 0 | ValueName = "GinaDLL" |
| 0040103E | 68 4C804000 | PUSH Malware_.0040804C | hKey |
| 00401043 | 8B45 FC | MOV EAX, DWORD PTR SS:[EBP-4] | RegSetValueExA |
| 00401046 | 50 | PUSH EAX | |
| 00401047 | FF15 00704000 | CALL DWORD PTR DS:[<&ADVAPI32.RegSetValueExA>] | |

Regedit (= *Registry Editor*, è uno strumento installato di default sui sistemi operativi Windows che viene utilizzato per visualizzare e/o modificare tutte le chiavi di registro)



4. Analisi dell'interazione del malware con il **file system**: identificazione della chiamata di sistema responsabile della modifica del contenuto della cartella di origine dell'eseguibile

Adesso vogliamo invece analizzare l'interazione del malware con il file system allo scopo di individuare la chiamata di sistema responsabile della modifica del contenuto della cartella in cui si trova l'eseguibile in esame.

A tale scopo, impostiamo il filtro sulla sezione *Show File System Activity* → 

Analizzando i risultati ottenuti, possiamo notare la presenza della chiamata di funzione **CreateFile** e, poco più avanti, della chiamata di funzione **WriteFile**:

| | | | |
|--------------------------|------|------------|--|
| Malware_Build_Week_U3... | 1868 | CreateFile | C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\msgina32.dll |
| Malware_Build_Week_U3... | 1868 | CreateFile | C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3 |
| Malware_Build_Week_U3... | 1868 | CloseFile | C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3 |
| Malware_Build_Week_U3... | 1868 | WriteFile | C:\Documents and Settings\Administrator\Desktop\Build_Week_Unit_3\msgina32.dll |

Come si vede, entrambe le chiamate di funzione puntano al percorso in cui si trova il malware ed il loro scopo è quello di creare un file vuoto (**CreateFile**) e scrivere al suo interno il malware appena estratto (**WriteFile**).

Quanto appena emerso non ci sorprende in quanto il comportamento tipico di un dropper propone, generalmente, due possibili scenari che fanno seguito all'estrazione del malware:

- La **creazione di un processo** (tramite chiamata di funzione *CreateProcess*), al fine di eseguire immediatamente il malware appena estratto
- Il **salvataggio del malware sul disco** in vista di un futuro utilizzo. In questo scenario, analogamente a quanto visto oggi, il dropper utilizzerà la coppia di funzioni *CreateFile* e *WriteFile* rispettivamente per creare un file vuoto e scrivere al suo interno il malware appena estratto

5. Descrizione del comportamento del malware in base alle informazioni raccolte tramite analisi statica e dinamica

A valle delle attività di analisi statica e dinamica fin qui svolte, è possibile trarre delle conclusioni sul comportamento del malware in esame. Inoltre, la verifica delle ipotesi formulate durante la fase di analisi statica ha avuto esito positivo: l'analisi dinamica ha confermato che il malware in oggetto

- **è un dropper**, in quanto contiene al suo interno un malware che viene effettivamente estratto (e salvato, in questo caso) al momento dell'esecuzione del file

- **ottiene la persistenza** dentro il sistema della macchina vittima, in quanto aggiunge se stesso alle entry dei programmi che devono essere eseguiti all'avvio del PC, in modo tale da essere avviato in maniera automatica e senza alcun intervento da parte dell'utente. A tale scopo, il malware crea una nuova chiave di registro tramite la chiamata di funzione *RegCreateKeyExA* e la configura con i valori desiderati tramite la chiamata di funzione *RegSetValueExA*. In fase di analisi dinamica, per verificare l'effettivo ottenimento della persistenza abbiamo riavviato la VM in seguito all'esecuzione del malware ed abbiamo potuto constatare che, al riavvio, la chiave "GinaDLL" è ancora presente nel registro di Windows.