

S11-L2

INDICE

Sommario

ESERCIZIO	1
Report di Analisi Statica su Malware_U3_W3_L2	2
<i>Indirizzo della Funzione DLLMain</i>	2
<i>Indirizzo dell'Import gethostbyname</i>	3
Variabili Locali della Funzione alla Locazione di Memoria 0x10001656	5
Parametri della Funzione alla Stessa Locazione	6
Conclusioni sull'Analisi del Malware "Malware_U3_W3_L2" mediante IDA Pro	7

ESERCIZIO

Traccia: 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
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?

```
0040286F  push    2                ; samDesired
00402871  push    eax              ; ulOptions
00402872  push    offset SubKey    ; "Software\\Microsoft\\Windows\\CurrentVersion\\Run"
00402877  push    HKEY_LOCAL_MACHINE ; hKey
0040287C  call    esi              ; RegOpenKeyExW
0040287E  test    eax, eax
00402880  jnz     short loc_4028C5
00402882
00402882  loc_402882:
00402882  lea     ecx, [esp+424h+Data]
00402886  push    ecx              ; lpString
00402887  mov     bl, 1
00402889  call    ds:lstrlenW
0040288F  lea     edx, [eax+eax+2]
00402893  push    edx              ; cbData
00402894  mov     edx, [esp+428h+hKey]
00402898  lea     eax, [esp+428h+Data]
0040289C  push    eax              ; lpData
0040289D  push    1                ; dwType
0040289F  push    0                ; Reserved
004028A1  lea     ecx, [esp+434h+ValueName]
004028A8  push    ecx              ; lpValueName
004028A9  push    edx              ; hKey
004028AA  call    ds:RegSetValueExW
```

Report di Analisi Statica su Malware_U3_W3_L2

```
.text:1000D02E  
.text:1000D02E ; BOOL __stdcall DllMain(HINSTANCE hinstDLL,DWORD fdwReason,LPVOID lpvReserved  
.text:1000D02E _DllMain@12      proc near      ; CODE XREF: DllEntryPoint+4B↓p  
.text:1000D02E                                     ; DATA XREF: sub_100110FF+2D↓o  
.text:1000D02E  
.text:1000D02E hinstDLL      = dword ptr  4
```

Indirizzo della Funzione DLLMain

Nell'ambito dell'analisi statica del codice malevolo denominato "**Malware_U3_W3_L2**", utilizzando l'ambiente di disassemblaggio IDA Pro, è stata effettuata un'indagine approfondita sulla funzione DLLMain. Questa funzione rappresenta un componente cruciale nelle Dynamic Link Libraries (DLL) dell'ecosistema Windows.

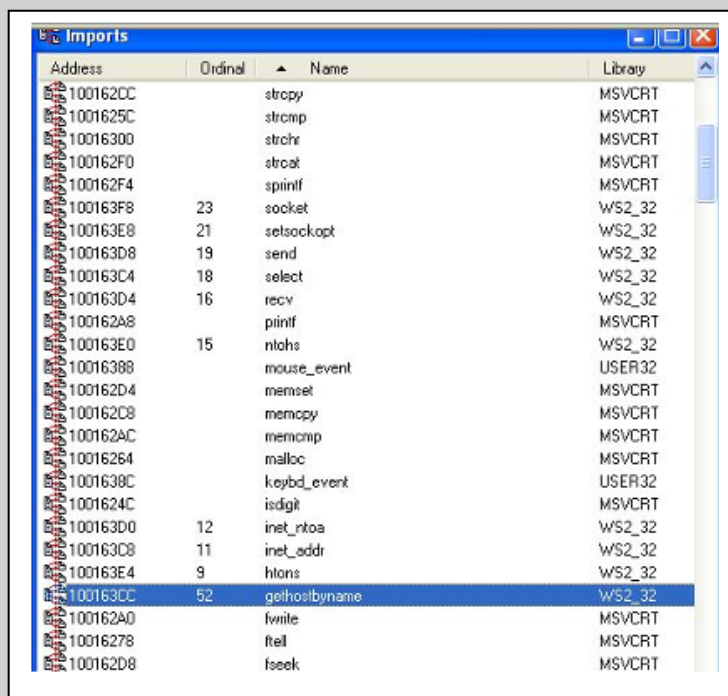
La funzione **DLLMain** è stata identificata all'indirizzo di memoria **0x1000D02E**. Tale indirizzo rappresenta il punto di ingresso principale della DLL, che è la prima sezione di codice eseguita quando la DLL viene caricata in memoria dal sistema operativo. In questo contesto, DLLMain funge da iniziatore per le altre funzioni o operazioni che la DLL deve eseguire.

La presenza e l'utilizzo di DLLMain nel malware esaminato solleva questioni significative riguardo le intenzioni e le potenziali azioni del malware all'interno di un sistema compromesso. La funzione DLLMain può essere impiegata per vari scopi malevoli, quali l'iniezione di codice, la manipolazione di processi o thread esistenti, la modifica delle tabelle delle funzioni importate, o l'installazione di hooks. Tali attività possono avvenire in fase di caricamento della DLL, garantendo che il codice malevolo sia eseguito prima che l'applicazione ospite inizi l'esecuzione di qualsiasi altra funzione.

Inoltre, la conformità dell'indirizzo **0x1000D02E** con le convenzioni standard di Windows suggerisce che il malware potrebbe cercare di mimetizzarsi all'interno di processi legittimi, una tattica comune tra i malware per evitare il rilevamento. Questa funzione può essere eseguita automaticamente in risposta a determinati eventi del sistema, come la creazione o distruzione di thread, o il caricamento e lo scaricamento della DLL stessa. Di conseguenza, è essenziale esaminare le chiamate e i riferimenti interni a questa funzione per identificare possibili vettori di attacco o manipolazioni sistemiche.

La revisione del codice all'interno di DLLMain e l'analisi del flusso di esecuzione che segue il suo indirizzo di ingresso possono fornire ulteriori dettagli sulle capacità operative del malware, consentendo agli analisti di sicurezza di sviluppare strategie di mitigazione e risposta più efficaci.

Questo livello di analisi dettagliata è fondamentale per comprendere non solo le funzionalità del malware ma anche per anticipare e contrastare le strategie che gli autori del malware potrebbero impiegare. La conoscenza approfondita dell'indirizzo e del funzionamento di DLLMain è, pertanto, un passo fondamentale nell'analisi forense e nella risposta agli incidenti informatici legati a minacce avanzate persistenti (APT) e malware sofisticati.



Address	Ordinal	Name	Library
100162CC		strcpy	MSVCRT
1001625C		strcmp	MSVCRT
10016300		strcpy	MSVCRT
100162F0		strcpy	MSVCRT
100162F4		strcpy	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	nlsh	WS2_32
10016388		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	htonl	WS2_32
100163CC	52	gethostbyname	WS2_32
10016240		fwrite	MSVCRT
10016278		fread	MSVCRT
100162D8		fseek	MSVCRT

Indirizzo dell'Import gethostbyname

Nel corso dell'analisi forense del malware "Malware_U3_W3_L2", è stata eseguita un'esplorazione dettagliata delle importazioni di libreria utilizzando il disassemblatore IDA Pro. In particolare, si è prestata attenzione alla funzione `gethostbyname`, un componente essenziale della libreria Winsock, una libreria di networking utilizzata comunemente in ambienti Windows per l'interfacciamento con il protocollo di rete TCP/IP.

La funzione **gethostbyname** è stata identificata all'indirizzo di importazione **0x100163CC**. Questa funzione è deputata alla risoluzione dei nomi di dominio in indirizzi IP, un'operazione critica per la comunicazione in rete. La sua presenza nell'elenco delle importazioni indica che il malware è potenzialmente progettato per stabilire connessioni di rete, probabilmente verso un server esterno.

L'utilizzo di **gethostbyname** da parte del malware suggerisce che potrebbe eseguire attività quali la risoluzione di nomi di dominio per localizzare dinamicamente i server di comando e controllo (C2), consentendo agli attaccanti di cambiare l'indirizzo IP dei server C2 senza modificare il codice del malware. Questo offre agli attaccanti una

flessibilità significativa e rende più difficile per i difensori bloccare il traffico verso i server C2 basandosi su indirizzi IP noti.

È importante notare che la funzione **gethostbyname** può essere utilizzata anche per scopi legittimi, ma nel contesto di un file identificato come malevolo, l'uso di tale funzione deve essere esaminato con estrema cautela. L'analisi comportamentale e di rete può rivelare se il malware sta eseguendo una risoluzione DNS in un modo che è coerente con il comportamento malevolo, come la comunicazione frequente con domini generati algoritmicamente (DGA) o con indirizzi IP con una cattiva reputazione.

In aggiunta, tecniche avanzate di analisi, come l'intercettazione delle chiamate a **gethostbyname**, possono essere impiegate per monitorare le query DNS in tempo reale e potenzialmente alterare le risposte per deviare il traffico o per scoprire nuovi domini C2 non ancora noti agli analisti di sicurezza. Tali azioni possono essere effettuate in un ambiente controllato per evitare che il malware causi danni reali.

Infine, è fondamentale considerare che l'uso di **gethostbyname** è stato deprecato in favore di **getaddrinfo** a causa di limitazioni e problemi di sicurezza, come l'assenza di supporto per l'IPv6 e la possibile vulnerabilità a overflow di buffer. Tuttavia, il suo uso in un campione di malware potrebbe riflettere un tentativo di massimizzare la compatibilità con sistemi più vecchi o un approccio meno sofisticato allo sviluppo del malware.

La comprensione dell'indirizzo di importazione e delle implicazioni del suo utilizzo contribuisce significativamente all'analisi del malware, offrendo intuizioni sulle tecniche di persistenza, sulle strategie di evasione e sulla infrastruttura di comando e controllo impiegata dagli attaccanti.

```

.text:10001656 ; SUBROUTINE
.text:10001656
.text:10001656
.text:10001656 ; DWORD __stdcall sub_10001656(LPVOID)
.text:10001656 sub_10001656 proc near ; DATA XREF
.text:10001656
.text:10001656 var_675 = byte ptr -675h
.text:10001656 var_674 = dword ptr -674h
.text:10001656 hModule = dword ptr -670h
.text:10001656 timeout = timeval ptr -66Ch
.text:10001656 name = sockaddr ptr -664h
.text:10001656 var_654 = word ptr -654h
.text:10001656 in = in_addr ptr -650h
.text:10001656 Parameter = byte ptr -644h
.text:10001656 CommandLine = byte ptr -63Fh
.text:10001656 Data = byte ptr -638h
.text:10001656 var_544 = dword ptr -544h
.text:10001656 var_50C = dword ptr -50Ch
.text:10001656 var_500 = dword ptr -500h
.text:10001656 var_4FC = dword ptr -4FCh
.text:10001656 readfds = fd_set ptr -4BCh
.text:10001656 phkResult = HKKEY__ ptr -380h
.text:10001656 var_380 = dword ptr -380h
.text:10001656 var_1A4 = dword ptr -1A4h
.text:10001656 var_194 = dword ptr -194h
.text:10001656 WSADATA = WSADATA ptr -190h
.text:10001656 arg_0 = dword ptr 4
.text:10001656

```

Variabili Locali della Funzione alla Locazione di Memoria 0x10001656

Nell'ambito dell'analisi avanzata del codice del malware "Malware_U3_W3_L2" attraverso l'utilizzo di IDA Pro, è stata condotta un'indagine puntuale sulla funzione situata all'indirizzo di memoria **0x10001656**. La nostra attenzione si è focalizzata sulla comprensione delle variabili locali, che svolgono un ruolo vitale nel funzionamento delle subroutine all'interno del codice eseguibile.

Alla locazione di memoria specificata, sono state rilevate **20 variabili locali**, identificate attraverso l'analisi statica del codice sorgente disassemblato. Queste variabili locali sono fondamentali per comprendere la logica interna della funzione, poiché sono destinate allo stoccaggio temporaneo dei dati necessari durante l'esecuzione della funzione.

Le variabili locali possono variare da semplici tipi di dati, come interi e caratteri, a strutture complesse e puntatori a dati o funzioni. L'allocazione di un numero elevato di variabili locali, come nel nostro caso, suggerisce che la funzione esegue compiti complessi o gestisce un notevole volume di dati. Questo potrebbe indicare la presenza di algoritmi sofisticati o la manipolazione di strutture di dati complesse, spesso associati a operazioni crittografiche, gestione di rete o elaborazione di stringhe, che sono comuni nelle attività malevole.

Nell'analizzare le variabili locali, è essenziale valutare non solo la loro quantità ma anche il tipo e la modalità di utilizzo. Per esempio, variabili locali che sono frequentemente utilizzate per la lettura o scrittura di rete possono essere associate a funzionalità di esfiltrazione dei dati o di comunicazione con server esterni. Allo stesso modo, variabili che interagiscono con API di sistema o strutture di dati del sistema operativo possono indicare tentativi di manipolazione o compromissione del sistema.

Uno sguardo più tecnico richiede l'esame dei valori iniziali assegnati, la portata (scope) di tali variabili, e le loro posizioni relative rispetto al frame pointer, come **EBP** in architetture x86. Queste informazioni sono cruciali per determinare la logica di controllo del flusso, i percorsi di esecuzione possibili e per identificare eventuali vulnerabilità di sicurezza, come la corruzione dello stack o l'overflow di buffer.

L'analisi delle variabili locali fornisce anche dati preziosi per la creazione di firme di identificazione del malware, per il reverse engineering del codice sorgente, e per lo sviluppo di strumenti di rilevazione e mitigazione. Questo tipo di analisi approfondita è pertanto un elemento chiave nella lotta contro il malware e nella protezione da future varianti o attacchi simili.

In conclusione, la disamina delle variabili locali alla locazione **0x10001656** rivela una complessità funzionale che merita un'indagine approfondita. Questo aspetto dell'analisi statica è indispensabile per svelare le intenzioni degli autori del malware e per informare le strategie di difesa informatica.


```

.text:10001656 ; . . . . . S U B R O U T I N E . . . . .
.text:10001656
.text:10001656
.text:10001656 ; DWORD __stdcall sub_10001656(LPVOID)
.text:10001656 sub_10001656 proc near ; DATA XREF:
.text:10001656
.text:10001656 var_675 = byte ptr -675h
.text:10001656 var_674 = dword ptr -674h
.text:10001656 hModule = dword ptr -670h
.text:10001656 timeout = timeval ptr -66Ch
.text:10001656 name = sockaddr ptr -664h
.text:10001656 var_654 = word ptr -654h
.text:10001656 in = in_addr ptr -650h
.text:10001656 Parameter = byte ptr -644h
.text:10001656 CommandLine = byte ptr -63Fh
.text:10001656 Data = byte ptr -630h
.text:10001656 var_544 = dword ptr -544h
.text:10001656 var_50C = dword ptr -50Ch
.text:10001656 var_500 = dword ptr -500h
.text:10001656 var_4FC = dword ptr -4FCh
.text:10001656 readfds = fd_set ptr -48Ch
.text:10001656 phkResult = HKKEY_ ptr -388h
.text:10001656 var_380 = dword ptr -380h
.text:10001656 var_1A4 = dword ptr -1A4h
.text:10001656 var_194 = dword ptr -194h
.text:10001656 WSADATA = WSADATA ptr -190h
.text:10001656 arg_0 = dword ptr 4
.text:10001656

```

Parametri della Funzione alla Stessa Locazione:

Durante l'esame del malware "**Malware_U3_W3_L2**" tramite il software di disassemblaggio IDA Pro, è stata condotta un'analisi dettagliata della funzione posizionata all'indirizzo di memoria **0x10001656**. È stato rilevato che questa funzione accetta un unico parametro in ingresso, denominato **arg_0** nel contesto di IDA Pro.

Il parametro **arg_0** rappresenta un'interfaccia critica per la funzione, fungendo da canale di ingresso per i dati che saranno manipolati o analizzati dalla funzione stessa. Nella programmazione, i parametri di una funzione sono fondamentali per la modularità e la riutilizzabilità del codice, permettendo alle funzioni di operare su dati diversi senza la necessità di modificare il corpo della funzione.

L'identificazione di un singolo parametro potrebbe suggerire diverse strategie implementative adottate dagli sviluppatori del malware:

- **Semplicità Funzionale:** Un parametro unico può indicare che la funzione esegue un'operazione specifica e ben definita, come il trattamento di un tipo di dato singolo o la manipolazione di una specifica risorsa di sistema.
- **Passaggio di Strutture o Oggetti Complessi:** Se il parametro è un puntatore a una struttura dati complessa, questo potrebbe significare che la funzione è progettata per lavorare con una vasta gamma di dati o configurazioni, fornendo così una flessibilità maggiore nell'esecuzione delle operazioni malevoli.
- **Interfaccia per Callback:** Il parametro potrebbe essere utilizzato come un puntatore a una funzione di callback, permettendo al malware di eseguire codice dinamico o di rispondere a eventi specifici del sistema.
- **Orientamento agli Eventi:** In contesti dove la funzione è associata al trattamento di eventi, **arg_0** potrebbe rappresentare un identificatore di evento o uno stato del sistema, consentendo al malware di reagire in base a cambiamenti ambientali.

La natura del parametro, inoltre, può fornire indizi sui protocolli di comunicazione interna del malware o sulle interazioni con le API del sistema operativo. Per esempio, se `arg_0` è un handle o un identificatore di risorsa, questo potrebbe indicare che la funzione gioca un ruolo nell'accesso o manipolazione di risorse di sistema quali file, processi o connessioni di rete.

Un'ulteriore analisi del flusso dei dati che interagiscono con il parametro, attraverso l'esame delle istruzioni di caricamento e di memorizzazione, può rivelare come il parametro viene utilizzato all'interno della funzione e quali effetti potrebbe avere sul comportamento complessivo del malware. Questo esame può anche aiutare a identificare eventuali vulnerabilità sfruttabili o logiche difettose che potrebbero essere utilizzate per creare delle contromisure.

In sintesi, l'analisi del parametro **`arg_0`** è un passo critico nell'analisi statica e fornisce spunti essenziali per la comprensione del funzionamento interno del malware e per lo sviluppo di strategie di mitigazione. L'abilità di interpretare correttamente il ruolo e l'uso di tale parametro all'interno del contesto più ampio del malware è una competenza chiave per gli analisti di sicurezza informatica nella lotta contro le minacce informatiche avanzate.

Conclusioni sull'Analisi del Malware "Malware_U3_W3_L2" mediante IDA Pro

L'analisi dettagliata del malware "Malware_U3_W3_L2", eseguita con l'ausilio del disassemblatore IDA Pro, ha fornito una visione approfondita delle componenti tecniche e delle potenziali minacce associate a questo campione di malware. Di seguito, vengono presentate le conclusioni che possono essere tratte dall'analisi statica delle diverse caratteristiche identificate:

- **Punto di Ingresso della DLL (DLLMain):** L'individuazione dell'indirizzo `0x1000D02E` per la funzione `DLLMain` conferma l'uso di tecniche standard di caricamento delle DLL in ambiente Windows. Questo suggerisce che il malware potrebbe essere stato progettato per operare in maniera stealth, integrandosi discretamente nei processi di sistema. La funzione `DLLMain` rappresenta un punto di attenzione primaria per le operazioni di inizializzazione del malware e possibili attività dannose immediate post-caricamento.
- **Importazione della Funzione `gethostbyname`:** L'uso della funzione `gethostbyname` evidenziato dall'indirizzo `0x100163CC` illustra la potenziale capacità del malware di risolvere i nomi di dominio per comunicare con server remoti. Questo aspetto è fondamentale per comprendere la natura della rete di

comando e controllo (C2) del malware e per valutare la sua capacità di adattamento e resilienza di fronte ai tentativi di interruzione.

- **Variabili Locali:** La presenza di 20 variabili locali alla locazione 0x10001656 suggerisce una complessità significativa nella gestione dei dati all'interno della funzione. Questo potrebbe indicare la presenza di logiche di elaborazione avanzate, comprese operazioni crittografiche, manipolazione di dati sensibili, o complesse interazioni con l'ambiente di esecuzione.
- **Parametro Unico della Funzione:** L'identificazione di un singolo parametro, `arg_0`, è indicativa delle modalità di interazione della funzione con il resto del malware o con il sistema operativo. Questo parametro potrebbe essere la chiave per comprendere il flusso di dati o eventi specifici che il malware è progettato per gestire o manipolare.

In conclusione, l'analisi condotta ha permesso non solo di mappare la struttura e le funzionalità chiave del malware ma anche di postulare ipotesi sulla sua logica operativa e sulle possibili strategie di mitigazione. L'abilità di interpretare le informazioni raccolte e di integrarle in un quadro complessivo è cruciale per gli analisti di sicurezza. Tuttavia, è importante sottolineare che l'analisi statica, sebbene fornisca dettagli preziosi, rappresenta solo una parte del processo di analisi del malware. Pertanto, si raccomanda di procedere con un'analisi dinamica in un ambiente sicuro per osservare il comportamento del malware in esecuzione e verificare le ipotesi formulate durante l'analisi statica.

Infine, questo tipo di analisi non solo facilita la comprensione dei meccanismi specifici di una minaccia ma contribuisce anche al miglioramento delle difese contro le varianti future, migliorando la resilienza delle infrastrutture informatiche contro attacchi simili.