DEFENCE TECH
Terra, Cielo, Mare, Spazio, Spazio cibernetico.
PROTEGGIAMOLI

LummaC2

Malware Lab Analysis Report

Summary

DEFENCE TECH

1

Our Malware Lab

1. Our Malware Lab

Defence Tech Malware Lab daily performs dissection of malware with the aim of timely understanding the technological evolutions of attacks, consolidating the knowledge of necessary to make more effective and faster the process of incidents responding, contributing to spreading information about emerging threats into the expert's community and among its clients.

Malware Lab analysts are continuously engaged in searching and experimenting new analysis tools, for increasing accuracy and scope of action with regard to the proliferation of new evasion and anti-analysis techniques adopted by malwares.

The Malware Lab is also committed to the development of proprietary tools for malware analysis and supporting the management and response of incidents.

Besides malware analysis, Malware Lab ideated and implemented an automatic process of extraction of **Indicators of Compromise (IOC)** that is daily run on dozens of new malwares, intercepted in the wide for populating our Knowledge Base.

CORRADO AARON VISAGGIO
Group Chief Scientist Officer & Malware Lab Director
a.visaggio@defencetech.it

DEFENCE TECH

2

Executive Summary

2. Executive Summary

LummaC2, also known as Lumma Stealer, is a dangerous information stealer written in the C programming language. Since 2022, it has been sold as Malware-as-a-Service (Maas) on Dark Web forums or Telegram channels. It primarily targets cryptocurrency wallets, browser extensions and two-factor authentication (2FA), committing credential harvesting and exfiltration. This malware family is distributed by several threat actors that employ different ways to deploy the final payload onto target systems.

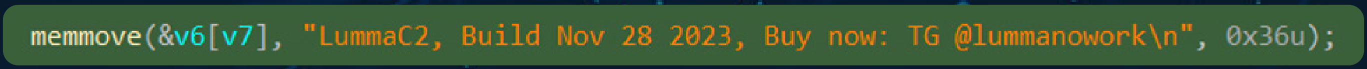
Common methods used to distribute Lumma are:

- **Fake software**
  This first scenario consists of users which search for specific free applications on search engines such as Google or Bing,but are tricked into downloading and installing malware through malvertising or fake webpages.

- **Phishing attacks**
  Lumma can also be delivered through malicious e-mail attachments by using social engineering techniques to trick individuals into running the suspicious files. This is the most common scenario for malware distribution.

- **Direct messages**
  In some cases, threat actors can also target users via social media platforms like forums or messaging applications. By using social engineering methods, they manage to gain the trust of users and persuade them to run their malicious software.

This report focuses on the analysis of the 4.0 version of LummaC2 malware family which is the most recent one. Moreover, it was submitted to the Malware Bazaar Database[1] on December 3rd. The sample carries an identification string which includes the day on which it was built, this is shown in figure 1.



```
memmove(&v6[v7], "LummaC2, Build Nov 28 2023, Buy now: TG @lummanowork\n", 0x36u);
```

Figure 1. Sample's build date

[1] *https://bazaar.abuse.ch/sample/a94fa51a5604486d4f68ca9ed09deec003c9f7f877c76200ee817303a743511e/*

DEFENCE TECH

3

Analysis

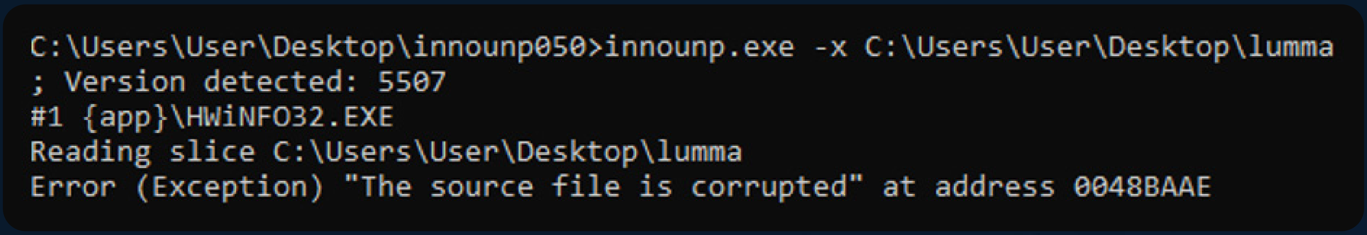
3. Analysis

The LummaC2 malware family has been in use since 2022. Recently, there has been a rise of its diffusion also in Italy, with victims involving both companies and public institutions.

In this report, our focus is placed on uncovering undocumented behaviours in order to contribute to a more comprehensive understanding of this malware family.

3.1 Overview on anti-analysis techniques and unpacking

The sample is disguised as an installer for a legitimate non-commercial diagnostic software named 'HWiNFO'. It contains strings resembling an installer created with Inno Setup[2], however, common Inno unpacking tools fail to properly extract the executable as illustrated in figure 2.



```
C:\Users\User\Desktop\innounp050>innounp.exe -x C:\Users\User\Desktop\lumma
; Version detected: 5507
#1 {app}\HWiNFO32.EXE
Reading slice C:\Users\User\Desktop\lumma
Error (Exception) "The source file is corrupted" at address 0048BAAE
```

Figure 2. Innounp fails to extract the content of the executable

We observed that the packer uses a series of anti-analysis and anti-debugging techniques to obfuscate its behaviour.

[2] *https://jrsoftware.org/isinfo.php*

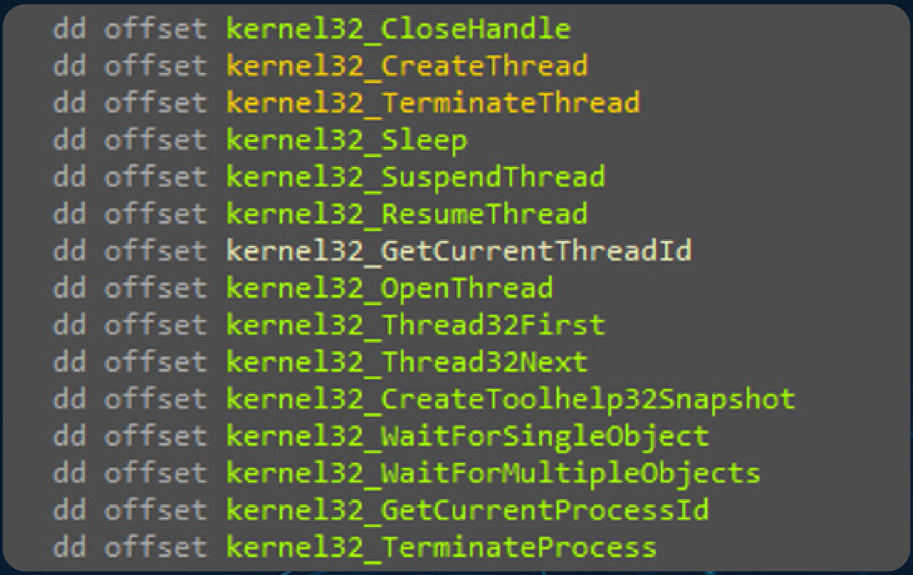
Static analysis is made difficult due to the multiple stages of dynamic code being decrypted and executed. At the same time the sample implements multiple anti-debugging techniques to slow down the analysis process, some of these are:

- The use of debugger and breakpoint detection functions.
- Structured exception handlers (SEH) to masquerade the code flow
- The payload loader being executed on multiple threads.

So, in order to continue with the analysis, we concealed the debugger from the Process Environment Block (PEB) and reduced to a minimum the use of breakpoints in code inside the sample itself.

The Main function initially extracts a shellcode into an allocated memory page by using the 'VirtualAlloc'[3] API and then sets up the permissions of the memory page to RWX (Read-Write-Execute), by using the 'VirtualProtect'[4] API.

Then the shellcode decrypts the next stage on the stack and constructs a data structure containing multiple function pointers it will need for its tasks. The function addresses are dynamically resolved through hash lookup in the loader data structures from the PEB.



```
dd offset kernel32_CloseHandle
dd offset kernel32_CreateThread
dd offset kernel32_TerminateThread
dd offset kernel32_Sleep
dd offset kernel32_SuspendThread
dd offset kernel32_ResumeThread
dd offset kernel32_GetCurrentThreadId
dd offset kernel32_OpenThread
dd offset kernel32_Thread32First
dd offset kernel32_Thread32Next
dd offset kernel32_CreateToolhelp32Snapshot
dd offset kernel32_WaitForSingleObject
dd offset kernel32_WaitForMultipleObjects
dd offset kernel32_GetCurrentProcessId
dd offset kernel32_TerminateProcess
```

Figure 3. Example of the functions' list

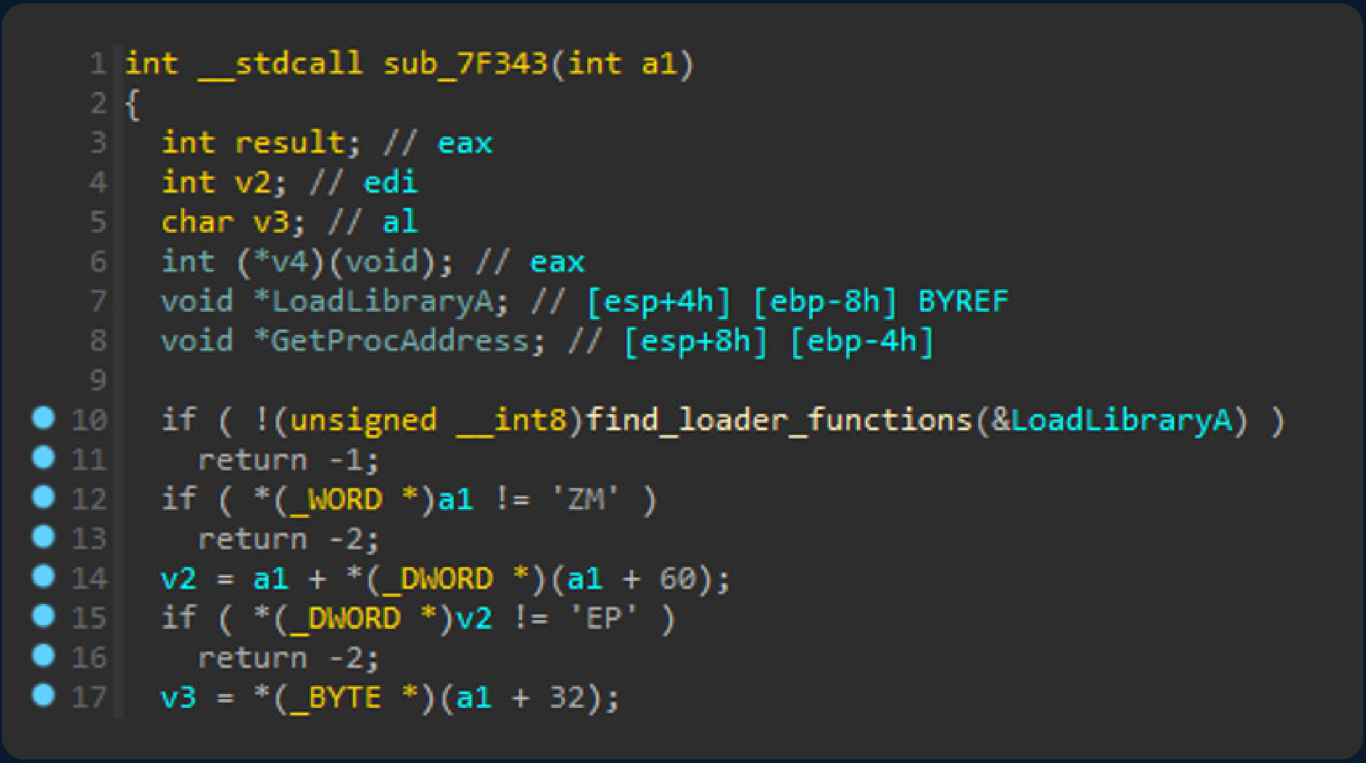
[3] *https://learn.microsoft.com/en-us/windows/win32/api/memoryapi/nf-memoryapi-virtualalloc*

[4] *https://learn.microsoft.com/it-it/windows/win32/api/memoryapi/nf-memoryapi-virtualprotect*

The functions in figure 3, for example, are necessary to suspend all threads of the process before advancing to the next stage.

At this point a new thread is launched which decrypts the PE loader that is used to launch the payload, finally yet another thread is launched with the task of loading and executing the payload.

The payload is dynamically mapped in memory using a PE loader, figure 4 shows the entry point of the loader. Its first argument is a pointer to the file to be executed. At this stage it is possible to simply stop the execution and read out the contents of the memory pointed to by 'a1' to recover a clean and executable copy of the payload.



```
int __stdcall sub_7F343(int a1)
{
  int result; // eax
  int v2; // edi
  char v3; // al
  int (*v4)(void); // eax
  void *LoadLibraryA; // [esp+4h] [ebp-8h] BYREF
  void *GetProcAddress; // [esp+8h] [ebp-4h]

  if ( !(unsigned __int8)find_loader_functions(&LoadLibraryA) )
    return -1;
  if ( *(_WORD *)a1 != 'ZM' )
    return -2;
  v2 = a1 + *(_DWORD *)(a1 + 60);
  if ( *(_DWORD *)v2 != 'EP' )
    return -2;
  v3 = *(_BYTE *)(a1 + 32);
```

Figure 4. PE file loader

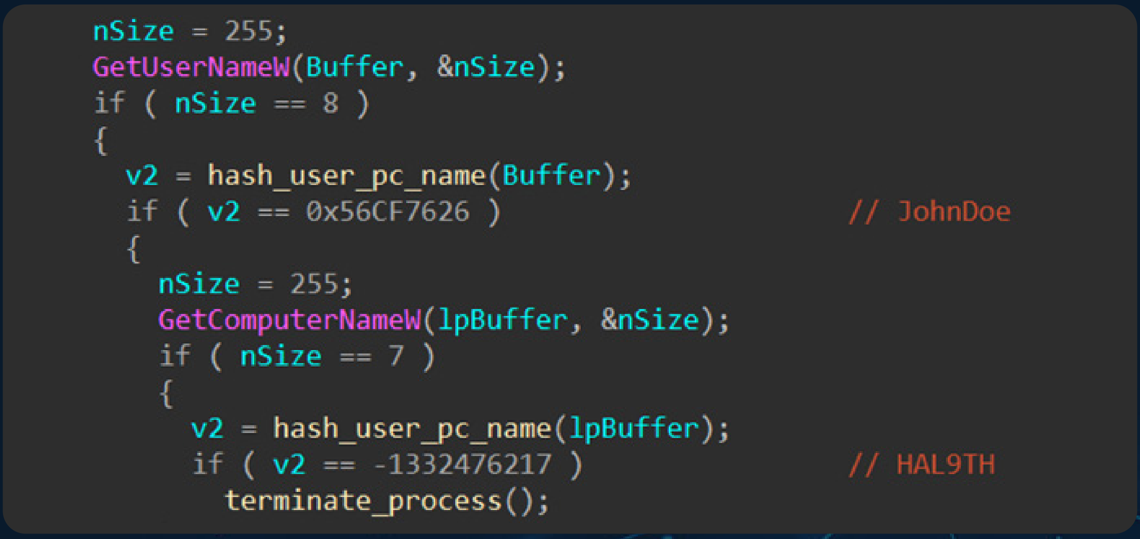
3.2 Technical analysis and behaviour

The extracted payload can be identified through its strings as a LummaC2 sample compiled on November 28th, 2023.

While some plaintext strings are visible, most of the ones related to stealing and exfiltrating are encrypted with different algorithms.

3.2.1 Anti-emulation check for Windows Defender

Windows Defender uses an emulator to determine the behaviour of executables. However, Lumma performs an anti-emulation check by verifying whether the computer name matches 'JohnDoe' and the username matches 'HAL9TH' as illustrated in the next figure. These are hard-coded values within the emulation layer of Defender, so if they are present, the malware immediately stops running.



```
nSize = 255;
GetUserNameW(Buffer, &nSize);
if ( nSize == 8 )
{
  v2 = hash_user_pc_name(Buffer);
  if ( v2 == 0x56CF7626 )                    // JohnDoe
  {
    nSize = 255;
    GetComputerNameW(lpBuffer, &nSize);
    if ( nSize == 7 )
    {
      v2 = hash_user_pc_name(lpBuffer);
      if ( v2 == -1332476217 )               // HAL9TH
        terminate_process();
```

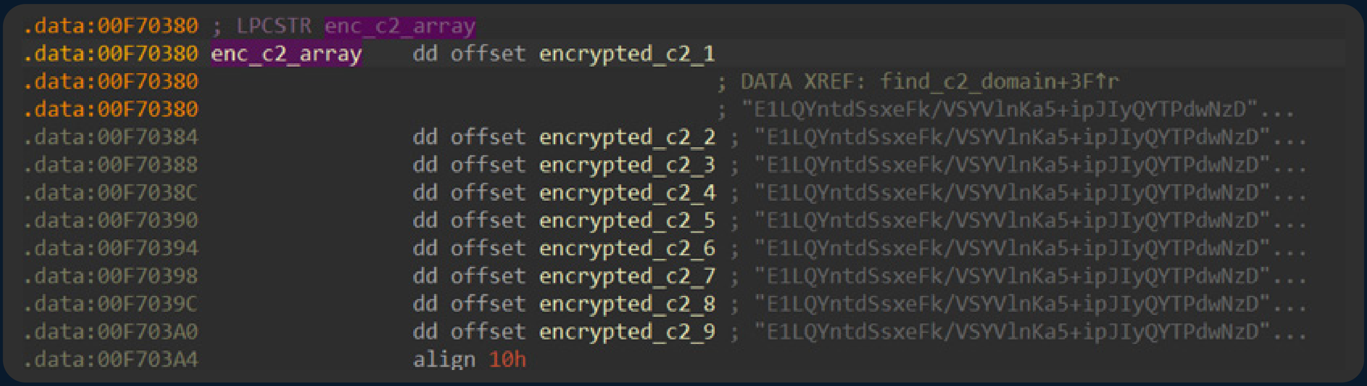
Figure 5. Defender anti-emulation

This behaviour is common in many other malware families.

3.2.2 Command and control domains

The analysed sample can communicate with up to 9 different C2 servers, their addresses are stored in an array of strings. Each string is encoded with Base64 and encrypted with a custom algorithm.

Decrypting these strings involves decoding the Base64 into a byte array and using the first 32 bytes as a repeating XOR key for the rest of the string, this algorithm is also used for decrypting the configuration received from the C2 server. The sample, however, implements two more variations of XOR encryptions for other hardcoded strings.



```
.data:00F70380 ; LPCSTR enc_c2_array
.data:00F70380 enc_c2_array    dd offset encrypted_c2_1
.data:00F70380                                         ; DATA XREF: find_c2_domain+3F↑r
.data:00F70380                                         ; "E1LQYntdSsxeFk/VSYVlnKa5+ipJIyQYTPdwNzD"...
.data:00F70384                 dd offset encrypted_c2_2 ; "E1LQYntdSsxeFk/VSYVlnKa5+ipJIyQYTPdwNzD"...
.data:00F70388                 dd offset encrypted_c2_3 ; "E1LQYntdSsxeFk/VSYVlnKa5+ipJIyQYTPdwNzD"...
.data:00F7038C                 dd offset encrypted_c2_4 ; "E1LQYntdSsxeFk/VSYVlnKa5+ipJIyQYTPdwNzD"...
.data:00F70390                 dd offset encrypted_c2_5 ; "E1LQYntdSsxeFk/VSYVlnKa5+ipJIyQYTPdwNzD"...
.data:00F70394                 dd offset encrypted_c2_6 ; "E1LQYntdSsxeFk/VSYVlnKa5+ipJIyQYTPdwNzD"...
.data:00F70398                 dd offset encrypted_c2_7 ; "E1LQYntdSsxeFk/VSYVlnKa5+ipJIyQYTPdwNzD"...
.data:00F7039C                 dd offset encrypted_c2_8 ; "E1LQYntdSsxeFk/VSYVlnKa5+ipJIyQYTPdwNzD"...
.data:00F703A0                 dd offset encrypted_c2_9 ; "E1LQYntdSsxeFk/VSYVlnKa5+ipJIyQYTPdwNzD"...
.data:00F703A4                 align 10h
```

Figure 6. List of encrypted C&C

In this case, the last value of the array decodes to an empty string, indicating a total of eight domains in the list. The decrypted list is provided in the IoC section.

LummaC2 tries to connect to each C2 domain sequentially until it receives a response containing the string "ok". Only one connection attempt is made to each server, if no valid C2 is found the sample continues the execution only to crash a bit later where the C2 address is needed to pull the configuration.

We observed that the C2 IP addresses of this sample are protected using CloudFlare[5] services. Detecting the suspicious behaviour, CloudFlare added an automated Phishing warning before forwarding each request to the real server. Cloudflare support seems to be a special feature of Lumma since it implements a custom bypass for this warning to allow the sample to communicate even when the domain is reported as malicious.

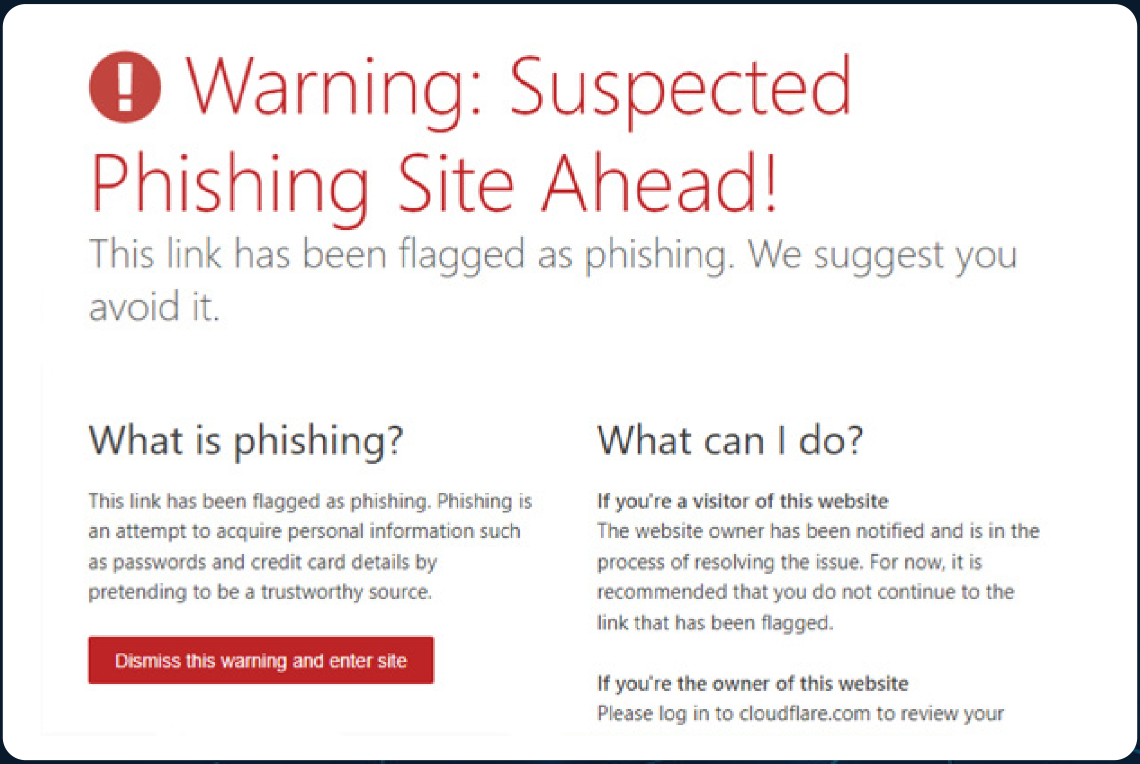
[5] *https://www.cloudflare.com/it-it/*

3.2.3 CloudFlare anti-phishing bypass

During the initial C2 discovery phase, the sample sends a POST request to the "/api" path with the body containing "act=life" to check the connectivity to the server. If the server is available, it responds with the string "ok", as already mentioned before.

The presence of the "ok" string indicates that the communication is not blocked by CloudFlare or that the current C2 server does not use their services, when this is the case Lumma performs no special actions and continues with the normal flow of operation.

However, when CloudFlare detects that a domain is involved in a malicious campaign, it intercepts each request with the warning shown in figure 7. When users are manually visiting the website, they can press the "Ignore" button to continue navigating without further interruptions.



Warning: Suspected Phishing Site Ahead!

This link has been flagged as phishing. We suggest you avoid it.

What is phishing?

This link has been flagged as phishing. Phishing is an attempt to acquire personal information such as passwords and credit card details by pretending to be a trustworthy source.

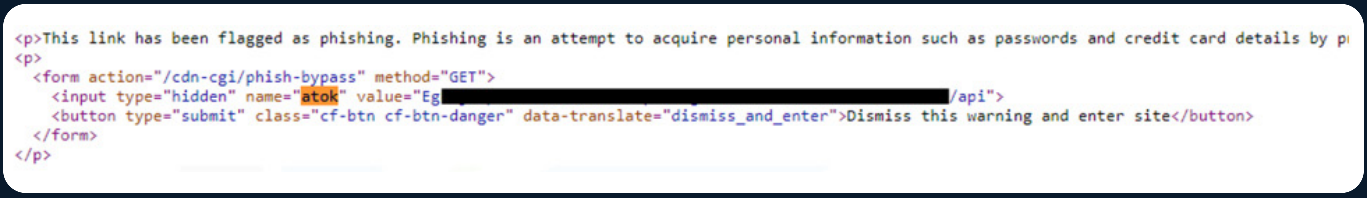
What can I do?

If you're a visitor of this website
The website owner has been notified and is in the process of resolving the issue. For now, it is recommended that you do not continue to the link that has been flagged.

If you're the owner of this website
Please log in to cloudflare.com to review your

Figure 7. CloudFlare warning page

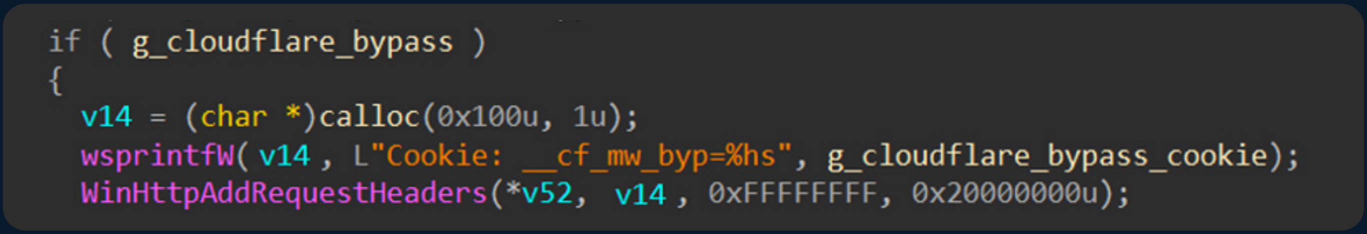
Lumma is able to detect the blocked page by checking whether the response contains the string "section". If the string is found, the malware then searches for the value of the "atok" field, visible in figure 8, which is used as POST parameter when the user presses the button to ignore the warning and proceed.



```
<p>This link has been flagged as phishing. Phishing is an attempt to acquire personal information such as passwords and credit card details by p
<p>
  <form action="/cdn-cgi/phish-bypass" method="GET">
    <input type="hidden" name="atok" value="Eg[redacted]/api">
    <button type="submit" class="cf-btn cf-btn-danger" data-translate="dismiss_and_enter">Dismiss this warning and enter site</button>
  </form>
</p>
```

Figure 8. Field 'atok' searched in the page source code

The sample stores this token value and subsequently adds it to each request within a cookie called '__cf_mw_byp', as shown in figure 9. This technique allows to automatically bypass the warning page and communicate with the C2 without the need for any user interaction.



```
if ( g_cloudflare_bypass )
{
  v14 = (char *)calloc(0x100u, 1u);
  wsprintfW( v14 , L"Cookie: __cf_mw_byp=%hs", g_cloudflare_bypass_cookie);
  WinHttpAddRequestHeaders(*v52,  v14 , 0xFFFFFFFF, 0x20000000u);
```

Figure 9. Warning bypass

3.2.4 Malware configuration

The sample retrieves the configuration by sending a POST request to the C2 server with the following body:

act=recive_message&lid=HbVgyI&j=default&ver=4.0

This request indicates that the sample version is 4.0 and the ID of this build is "HbVgyI".
The server replies with a Base64 string encrypted with the same algorithm used to protect the C&C's list. Decoding the response produces a Json string containing the configuration of the sample; the string is longer than 500 lines, the following snippet is an extract of some relevant fields:

```
"v": 4,
"se": true,
"ad": false,
"ex": [
 {
   "en": "ejbalbakoplchlghecdalmeeeajnimhm",
   "ez": "MetaMask"
 },
 ],
"c": [
 {
   "t": 0,
   "p": "%appdata%\\Electrum\\wallets",
   "m": ["*"],
   "z": "Wallets/Electrum",
   "d": 1,
   "fs": 20971520
 },
 {
   "t": 1,
   "p": "%localappdata%\\Google\\Chrome\\User Data",
   "z": "Chrome"
 },
```

The Json configuration starts with three fields: "v", "se" and, "ad". The first one is never read in the code, but we assume it indicates the version as it matches the version string in the request.

"se" is a Boolean flag indicating whether the sample should acquire a screenshot of the system before terminating.
"ad" is a Boolean flag indicating whether the sample should delete itself before terminating, when this flag is true the sample executes the following command after all the files of interest have been collected:

cmd.exe /c timeout /nobreak /t 3 & fsutil file setZeroData offset=0 length=<size of the sample> '<path of the sample>' & erase '<path of the sample>' & exit

This self-deletion command waits three seconds, so the sample has the time to terminate itself and then uses 'fsutil' to completely overwrite the executable on disk with zeroes to reduce the chance of recovering it for forensics analysis, then it uses the erase command to actually delete the file.

The final two fields of the configuration are "ex" which contains a list of browser extensions and "c" which contains a list of paths on disk. These two lists represent the files the threat actor is interested in exfiltrating.

The following tables summarize the targets of the configuration of the sample we analysed.

Crypto wallet extensions for Chromium browsers:

MetaMask	Trust Wallet	TronLink	Ronin Wallet	Sui
BNB[6] Wallet	Yoroi	Math	Coinbase	CryptoCom
Guarda	EQUA	Jaxx Liberty	BitApp	Sollet
iWallet	EnKrypt	MEW[7] CX	Guild	Nabox
Staturn	NeoLine	Clover	Rabby	DAppPlay
Pontem	Martian	Nami	Petra	Hycon Lite Client
ExodusWeb3	PolkadotJS	SubWallet	Talisman	Liquality
Station	Keplr	Auro	Polymesh	ICONex
KHC[8]	Temple	TezBox	BitClip	Steem KeyChain
Nash	ZilPay	Coin98	Cyano	Byone
LeafWallet	Phantom	UniSat	OneKey	

[6] *Binance Chain*

[7] *MyEtherWallet*

[8] *KoHo Chain*

Password manager extension for Chromium browsers:
Bitwarden

Authenticator extensions for Chromium browsers:

Authenticator	Authy	EOS[9] Authenticator	GAuth Authenticator	Trezor Password Manager[10]

Target list of desktop programs installed in the system:

Name	**Type**	**Name**	**Type**
TheBat	Mail Client	Pegasus	Mail Client
Mailbird	Mail Client	EmClient	Mail Client
Chrome Beta	Browser	Opera	Browser
Opera Neon	Browser	Opera GX Stable	Browser
Edge	Browser	Brave	Browser
EpicPrivacyBrowser	Browser	Vivaldi	Browser
360Browser	Browser	CocCoc	Browser
Mozilla Firefox	Browser	Chrome	Browser
AnyDesk	Remote desktop	FileZilla	FTP client
KeePass	Password manager	Steam	Digital store
Telegram	Messaging platform	Electrum	Crypto wallet
Ethereum	Crypto wallet	Exodus	Crypto wallet
Ledger Live	Crypto wallet	Atomic	Crypto wallet

[9] Epic Online Services, it's an Epic Games's service extension. Epic Games is a cross-platform game engine.
[10] It has been deprecated.

Coinomi	Crypto wallet	Authy Desktop	2FA authenticator
Bitcoin core	Crypto wallet	Binance	Crypto wallet
JAXX	Crypto wallet		

Furthermore, the configuration has two special fields called "Important Files/Profile" and "Important Files/Desktop" which will steal all txt and pdf files from the user profile and desktop folders; this is in the hope of exfiltrating crypto wallet credentials or other sensitive files.

Finally, regardless of the C2 configuration, the sample seems hardcoded to always steal the stored credentials of Thunderbird and the Windows 10 e-mail app as well as the list of installed programs and some basic information about the system.

All the stolen information is gathered in multiple zip files, which are sent one by one to the C2 server using POST requests to the "/api" endpoint.

3.3 IOC

The next table contains IoC of the LummaC2 sample analysed in this report.

Note: detection rates are as of time of writing, given the low rates they are likely to increase over the course of the following days as AV vendors update their products.

Type	Value	Note
SHA-256	a94fa51a5604486d4f68ca9ed09deec003c9f7f877 c76200ee817303a743511e	PE file VirusTotal - 17/56
Domain	buffettrickopsd[.]pw	C2 AlienVault VirusTotal - 11/88
Domain	tirechinecarpett[.]pw	C2 AlienVault VirusTotal - 20/88
Domain	musclefarelongea[.]pw	C2 AlienVault VirusTotal - 19/88
Domain	fanlumpactiras[.]pw	C2 AlienVault VirusTotal - 18/88
Domain	ownerbuffersuperw[.]pw	C2 AlienVault VirusTotal - 16/88
Domain	freckletropsao[.]pw	C2 AlienVault VirusTotal - 17/88
Domain	hemispheredonkkl[.]pw	C2 AlienVault VirusTotal - 18/88
Domain	medicinebuckerrysa[.]pw	C2 AlienVault VirusTotal - 17/88

Table 1. IoC

DEFENCE TECH

4

Conclusions

4. Conclusions

LummaC2 targets credentials stored in browsers and various desktop applications, the stolen data regularly ends up on dark web markets. Cybercriminals can then purchase the data and use it to gain access to an organization's network or systems. For this reason, information stealers are a highly dangerous category of malware that make it critical for organizations to swiftly detect and mitigate infections.

MaaS platforms usually provide 'script kiddies'[11], unskilled threat actors, with the tools necessary to orchestrate, automate and execute successful malware attacks with minimal technical skills.

An organization's first line of defence to avoid victimization is to ensure that employees are properly trained to avoid phishing attempts and quickly report to the relevant figures any suspicious message.

Security administrators should also enforce a robust credential hygiene program, enforcing MFA (Multi-Factor Authentication) using trusted authentication applications like Google or Microsoft Authenticator, introducing biometric authentication and/or physical access tokens where applicable.

[11] An individual who lacks programming skills but uses script or tools developed by others to attempt cyberattacks, rather than creating their own.

DEFENCE TECH

Terra, Cielo, Mare, Spazio, Spazio cibernetico.
PROTEGGIAMOLI

DONEXIT IT SECURITY

NEXT INGEGNERIA DEI SISTEMI

FORAMIL RADAR TECHNOLOGIES & DEFENCE SYSTEMS

INNODESI electronic systems

Defence Tech Holding S.p.A Società Benefit

Via Giacomo Peroni, 452 - 00131 Roma
tel. 06.45752720 - fax 06.45752721
info@defencetech.it - www.defencetech.it