A Look at:

LummaC2/Lumma Stealer

Document Control

Version Number	Date	Change Summary
0.1	11 Feb 2024	Initial draft.
0.2	12 Feb 2024	Added new IoCs and analysis of sample.

Table of Contents

What is LummaC2/Lumma Stealer?

The Lumma stealer gets its name from the threat actor “Shamel” – who originally offered the stealer for sale on various different underground markets, “Shamel” used the alias “Lumma” in making the posts. “Shamel” is also responsible for the 7.62mm Stealer [1]. The stealer first emerged in 2022, appearing in tweets from August/September 2022.

Lumma primarily targets cryptocurrency wallets, browser extensions and two factor authentication. Lumma is able to pull information from compromised targets including: system data, installed program data, cookies, usernames, passwords, credit card numbers, connection history, cryptocurrency wallet data [2].

Lumma is priced from $250 (The “Experienced” edition) up to $1000 (the “Corporate” edition). The source code for Lumma was also available to purchase for a time for $20000 [2]. Payments are processed via Coinbase using a wide range of cryptocurrency options [3].

Distribution of Lumma

Lumma has typically been distributed under the guise of cracked/fake popular software. This has been seen in the form of a fake crack for Nitro Pro, other variants include claiming to be popular software like VLC or Sony VEGAS Pro, or even as a fake browser update [4]. However, as recently as Jan 2024, Lumma has been seen distributed via malicious URLs on YouTube videos, the videos offer free software to users and encourage them to download a malicious zip file [5].

Lumma has also seen distribution via Discord. A user will receive a message from someone, asking them to test out their new game and offering to pay for their time. Upon accessing the link sent, the Lumma download is triggered multiple times, and if executed, it will talk back to its C2 server and attempt to exfiltrate sensitive information from the machine [6].

Lumma has also been seen targeting YouTubers in spear-phishing campaigns [7]. The download links to the software are typically for well known, trusted file hosting services like SharePoint/OneDrive, Mediafire and DropBox.

Execution of Lumma

Once the stealer has acquired a foothold on a victim machine via direct execution, the malware reaches out to a command and control server to transfer pilfered data. The stealer itself is heavily obfuscated, using many techniques including Control Flow Flattening to make it very hard to determine exactly what is going on.

When a connection is established, Lumma sends a POST request to its C2 server with a hardcode user agent and a parameter “act=life” to check in. Next, another POST request is sent with the Lumma ID and parameter “act=receive-message”, which will then upload a compressed version of any stolen data to the C2 server at URI “/api.” [8] Lumma is still being actively developed and updated to better evade anti-virus detections.

Stealer Sample

Stealer sample acquired from:
https[://]bazaar[.]abuse[.]ch/sample/
19fefb958bd9c9280d07754ab903022a3dc9fc380a6964733a1dcc016aba8150/

Stealer SHA-256: 19fefb958bd9c9280d07754ab903022a3dc9fc380a6964733a1dcc016aba8150

Integrity Checking

Once extracted, the sample is confirmed to be a Windows PE32 Executable with command: file [filename]. The integrity is confirmed by running sha256sum [filename] where the output returns: 19fefb958bd9c9280d07754ab903022a3dc9fc380a6964733a1dcc016aba8150.

Analysis with IDA

An analysis of this with IDA shows heavy obfuscation as expected.

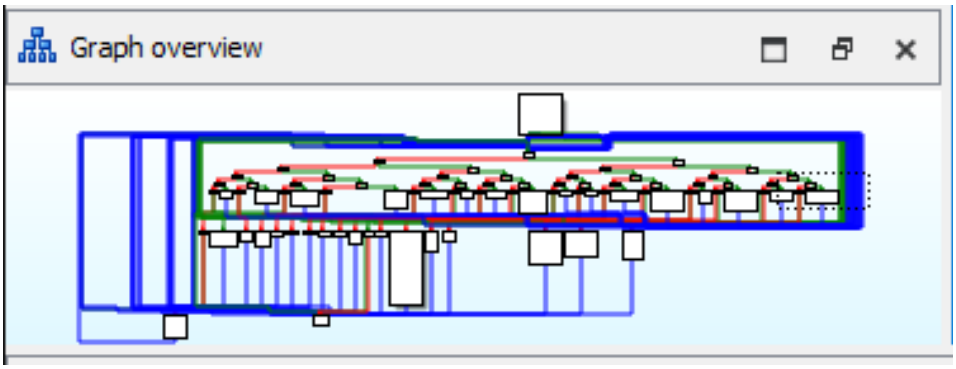

Figure 1: Overview of process.

A dive into the PE file reveals a lot of code obfuscated with the string “576xed”.

```
push    [ebp+var_14]    ; int
push    ebx             ; int
push    [ebp+var_50]    ; lpString2
push    [ebp+lpString]  ; lpString
push    [ebp+var_58]    ; int
call    sub_435930
add     esp, 14h
push    offset aWall576xedetsE ; "Wall576xedets/Eth576xedereum"
call    sub_45DF86
add     esp, 4
mov     esi, eax
push    offset aK_4     ; "k"
call    sub_45DF86
add     esp, 4
mov     edi, eax
push    offset aAppd576xedataE ; "%appd576xedata%\\Ethe576xedreum"
call    sub_45DF86
add     esp, 4
push    [ebp+var_14]    ; int
push    ebx             ; int
push    esi             ; lpString2
push    edi             ; lpString
mov     edi, 2
push    eax             ; int
call    sub_435930
add     esp, 14h
push    [ebp+var_14]    ; int
push    edi             ; int
push    offset aW_0     ; "W"
mov     esi, offset String ; "*"
push    esi             ; lpString
push    offset aAppdataExodusE ; "%appdata%\\Exodus\\exodus.wallet"
call    sub_435930
add     esp, 14h
push    [ebp+var_14]    ; int
push    edi             ; int
push    offset aW_1     ; "W"
push    esi             ; lpString
push    offset asc_4A357A ; "%"
call    sub_435930
add     esp, 14h
push    [ebp+var_14]    ; int
push    edi             ; int
push    offset aW_5     ; "W"
push    esi             ; lpString
push    offset aAppdataAtomicL ; "%appdata%\\atomic\\Local Storage\\level"...
call    sub_435930
add     esp, 14h
```

Figure 2: Obfuscated code.

Deobfuscating the above reveals a list of what Lumma is trying to extract from its targets. This is a wide variety of credentials, logs and other system files. A heavy focus seems to be placed on extracting information from browsers like stored credentials etc.

```
text "UTF-16LE", '%localappdata%\Google\Chrome'
text "UTF-16LE", '\User Data',0
text "UTF-16LE", 'Chromium',0
text "UTF-16LE", '%localappdata%\Chromium\User Data',0
text "UTF-16LE", 'Edge',0
text "UTF-16LE", '%localappdata%\Microsoft\Edge\User Data'
text "UTF-16LE", 'Kometa',0
text "UTF-16LE", '%localappdata%\Kometa\User Data'
text "UTF-16LE", 'Opera Stable',0
text "UTF-16LE", '%appdata%\Opera Software\Opera Stable',0
text "UTF-16LE", 'Opera GX Stable',0
text "UTF-16LE", '%appdata%\Opera Software\Opera GX Stable',0
text "UTF-16LE", 'Opera Neon',0
text "UTF-16LE", '%appdata%\Opera Software\Opera Neon\User Data',0
text "UTF-16LE", 'Brave Software',0
text "UTF-16LE", '%localappdata%\BraveSoftware\Brave-Browser\User Data'
text "UTF-16LE", 'Comodo',0
text "UTF-16LE", '%localappdata%\Comodo\Dragon\User Data',0
text "UTF-16LE", 'CocCoc',0
text "UTF-16LE", '%localappdata%\CocCoc\Browser\User Data',0
```

Figure 3: Snip of deobfuscated code.

Amongst the code we can also discern the current Lumma version, which is marked with the build date.

```
text "UTF-16LE", 'ser32.dll',0
aLummac2B db 'LummaC2, Build 20233101',0Ah,0
.LidLumma db 'LID(Lumma ID): ',0
```

Figure 4: Lumma version.

The C2 IP address can also be seen in plaintext:

```C++
aRxasleWalmartb db 'rXaSle--Walmartbrandportal',0
aLid            db 'lid',0              ; DATA XREF: sub_446338+71↑o
aFile           db 'file',0             ; DATA XREF: sub_446338+CA↑o
a8211725580     db '82.117.255.80',0    ; DATA XREF: sub_446338+285↑o
                text "UTF-16LE", '/c2sock',0
                text "UTF-16LE", 'winhttp.dll',0
                text "UTF-16LE", 'TeslaBrowser/5.5',0
```

Figure 5: Lumma C2 IP from sample.

Interestingly, there's also the string 'Walmartbrandportal' – not entirely sure what that references but it may be a link back to how this sample was originally distributed. A final interesting thing is a check that runs for debuggers:

```
.text:0047ED4C                 mov     [ebp+var_58], 40000015h
.text:0047ED53                 mov     [ebp+var_54], 1
.text:0047ED5A                 mov     [ebp+var_4C], eax
.text:0047ED5D                 call    ds:IsDebuggerPresent
```

Figure 6: Debug check.

If a debugger is detected, the code simply dies.

Threat Actors, Associated Accounts and Other Intel

Telegram

Display Name	Link	Description	State
@LummaC2Link Figure 7 Figure 8	hxxps[://]t[.]me/s/LummaC2Link	Channel with master list of Lumma telegram contacts.	ONLINE
@LummaC2Stealer Figure 9 Figure 10	hxxps[://]t[.]me/LummaC2Stealer	Main channel for Lumma stealer, posts updates.	ONLINE
@LummaC2Team Figure 11	hxxps[://]t[.]me/LummaC2Team	Main public channel for the Lumma C2 Team, no preview available.	ONLINE
@lummaseller126 Figure 12	hxxps[://]t[.]me/lummaseller126	Main account for the primary(?) verified seller of Lumma.	ONLINE
Lumma C2 – reviews Figure 13	hxxps[://]t[.]me/+zq0CE4r--ZEyM2Fi	Channel to post reviews for Lumma.	ONLINE
@lummanowork Figure 14	hxxps[://]t[.]me/lummanowork	Bug reporting user account for Lumma.	ONLINE
@LummaC2Blacklist Figure 15 Figure 16	hxxps[://]t[.]me/LummaC2Blacklist	Blacklist of scammers/false sellers of Lumma.	ONLINE

Documentation

Document	**Link**	**Description**	**Stable [Safe] Link**
About Lumma Document (ENG)	hxxps[://]telegra[.]ph/LummaC2---universal-stealer-a-malware-for-professionals-07-27	Document written by the creators outlining the current capabilities of Lumma.	https://github.com/contrxl/malware/blob/main/Lumma_Documents/Lumma_About_ENG.md
About Lumma Document (RU)	hxxps[://]telegra[.]ph/LummaC2---unikalnyj-stiller-instrument-dlya-professionalov-07-05	Document written by the creators outlining the current capabilities of Lumma.	https://github.com/contrxl/malware/blob/main/Lumma_Documents/Lumma_About_RU.md

Indicators of Compromise

IP Addresses & URLs

82[.]117[.]255[.]80
82[.]117[.]255[.]80/c2sock
104[.]21[.]38[.]174
172[.]67[.]137[.]14
192[.]229[.]221[.]95
hxxps[://]combinethemepiggerygoj[.]site/375
lendremindcenterpassew[.]site
qualifiedbehaviorrykej[.]site

Files

19fefb958bd9c9280d07754ab903022a3dc9fc380a6964733a1dcc016aba8150
420e3ed23079824f06ee90685938d2a3
072BB6BC342B35F3476993735B8563AE04D51979DD5F5725488151331D320107
Arglesmorgay
Arglesmorgay.exe
C:\lojurinuwe\girubexufiyom.pdb

User Agents

TeslaBrowser/5.5

Appendices

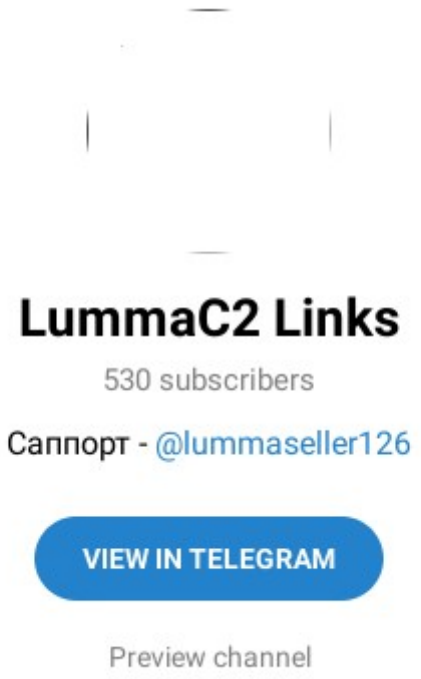

Figure 7: Lumma links Telegram channel.

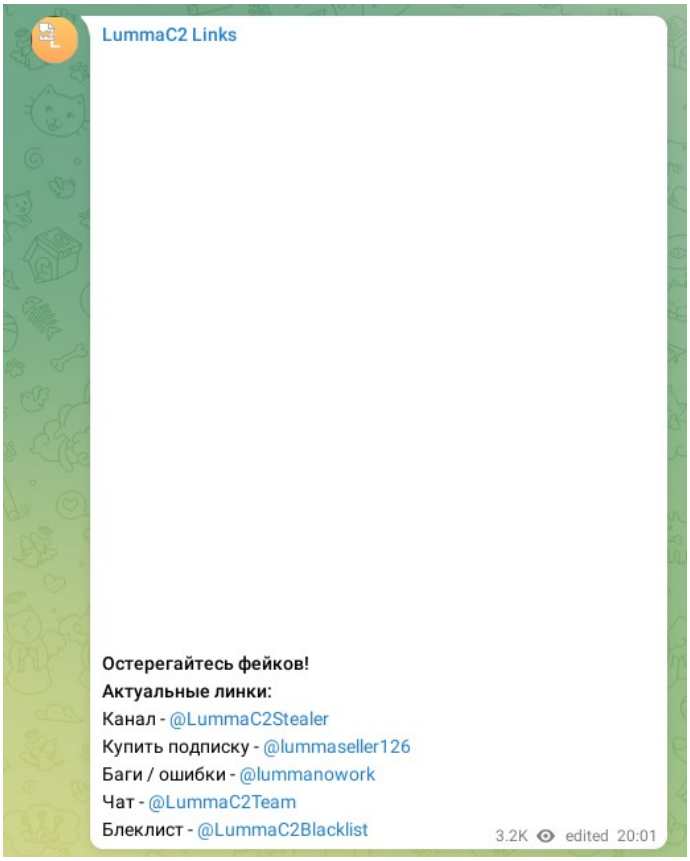

Figure 8: Inside Lumma links Telegram channel.

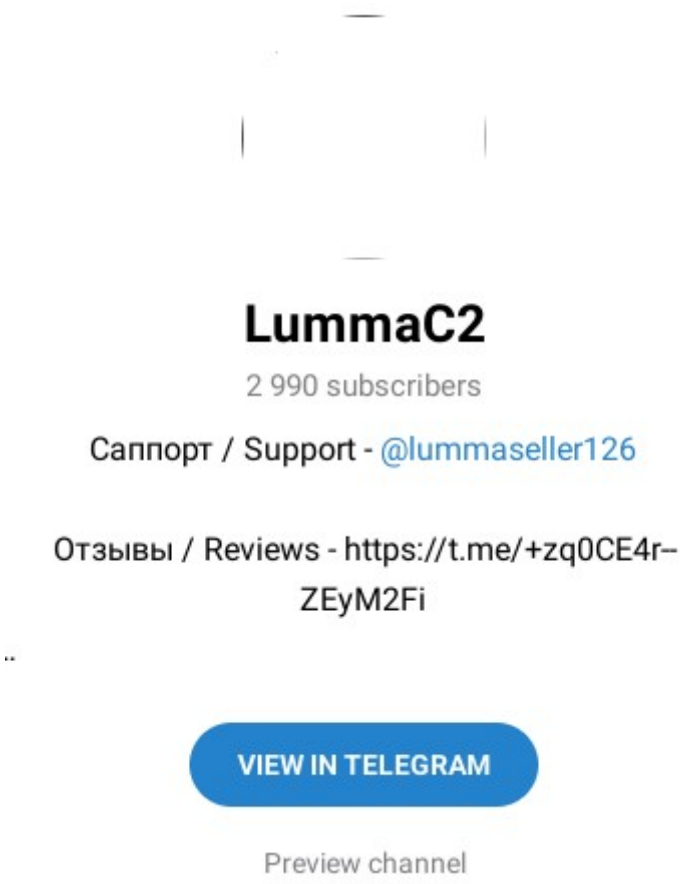

Figure 9: Lumma Stealer Telegram channel.

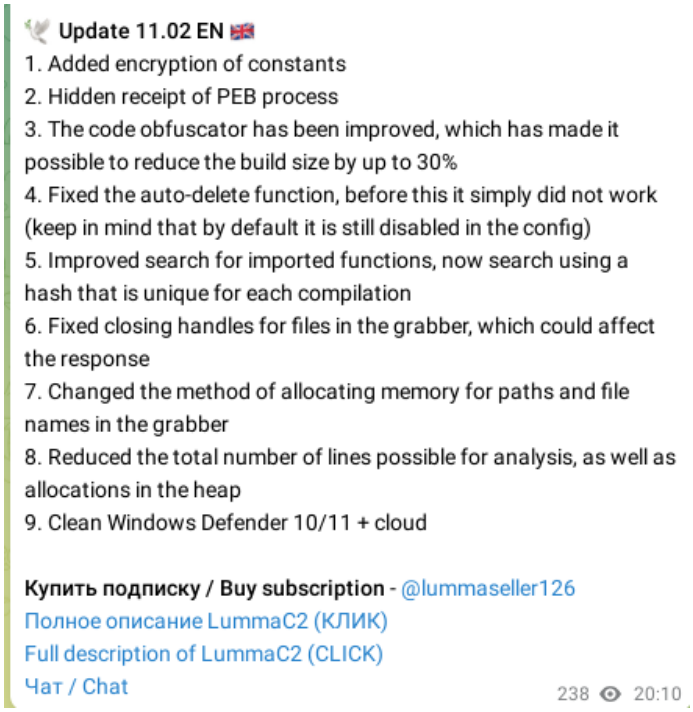

Figure 10: Most recent update notes from Luma stealer Telegram channel.

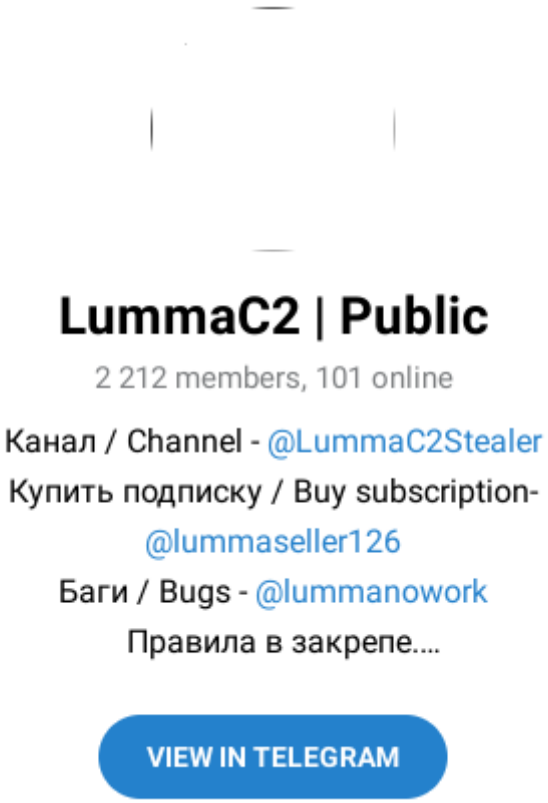

Figure 11: Lumma C2 Team Telegram channel.

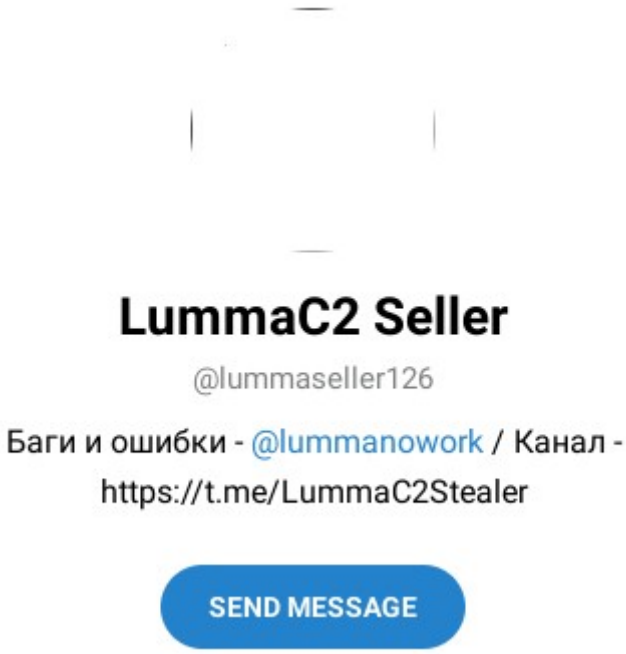

Figure 12: Lumma seller Telegram channel.

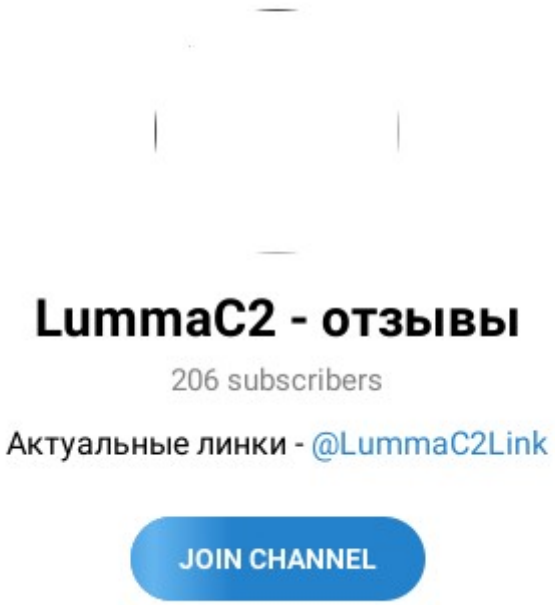

Figure 13: Lumma reviews Telegram channel.

Figure 14: Lumma bug reporting Telegram user.

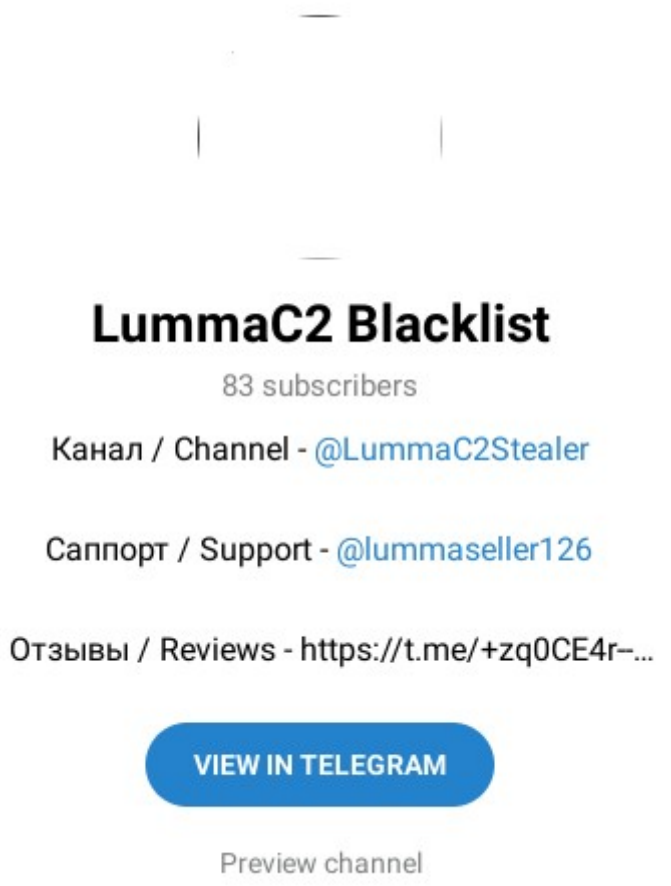

Figure 15: Lumma blacklist Telegram channel.

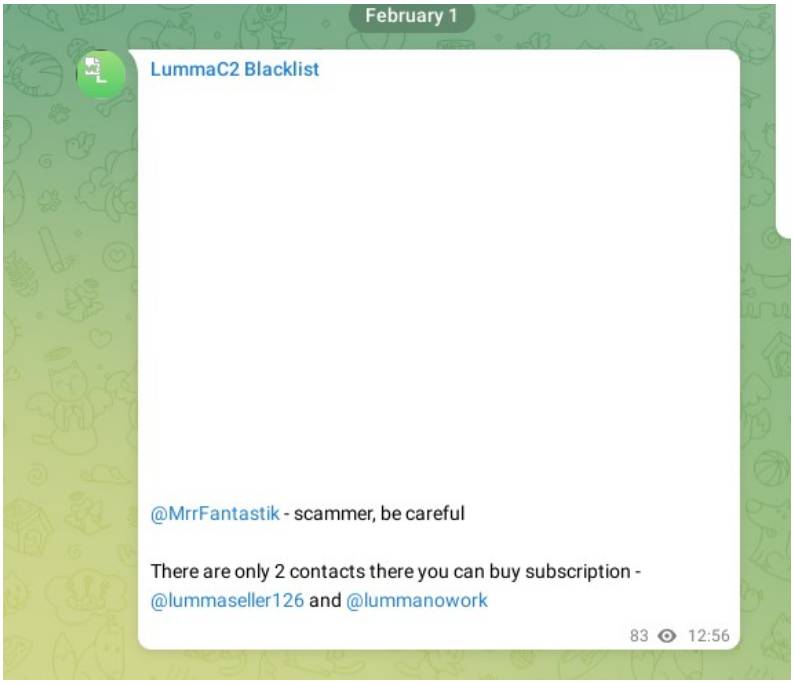

Figure 16: Inside Lumma blacklist telegram channel.

Bibliography

1: KELA Cyber Intelligence Centre, The Next Generation of Info Stealers, 2022
2: Emily Megan Lim, The Rise of the Lumma Info-Stealer, 2023
3: Alberto Marin, Everything you need to know about the LummaC2 stealer: leveraging IDA Python and Unicorn to deobfuscate Windows API hashing, 2023
4: eSentire Threat Response Unit, Fake Browser Updates Distribute LummaC Stealer, Amadey and PrivateLoader Malware , 2023
5: Cyware Alerts - Hacker News, Researchers Disclose New Lumma Stealer Campaign Distributed via YouTube., 2024
6: Carl Malipot, Beware: Lumma Stealer Distributed via Discord CDN, 2023
7: Jiho Kim, Sebin Lee, Lumma Stealer targets YouTubers via Spear-phishing Email, 2023
8: Cara Lin, Deceptive Cracked Software Spreads Lumma Variant on YouTube, 2024

Table of Figures