Following NoName057(16) DDoSia Project's Targets

blog.sekoia.io/following-noname05716-ddosia-projects-targets/

29 June 2023

Research & Threat Intelligence

DDoSia is a Distributed Denial of Service (DDoS) attack toolkit, developed and used by the pro Russia hacktivist nationalist group NoName057(16) against countries critical of the Russian invasion of Ukraine.

CTI
DDoS
Europe
Hacktivism

Amaury G., Charles M. and Threat & Detection Research Team - TDR June 29 2023

14 minutes reading

Table of contents

Context

DDoSia is a Distributed Denial of Service (DDoS) attack toolkit, developed and used by the pro Russia hacktivist nationalist group **NoName057(16)** against countries critical of the Russian invasion of Ukraine.

The DDoSia project was launched on Telegram in early 2022. The NoName057(16) main group main Telegram channel reached more than **45,000 subscribers** as of June 2023, while the DDoSia project channels reached over 10,000 users. Administrators posted instructions for potential volunteers who want to participate in projects, and they added the possibility to **pay in cryptocurrency** for users who declare a valid TON wallet based on their contribution to the DDoS attacks.

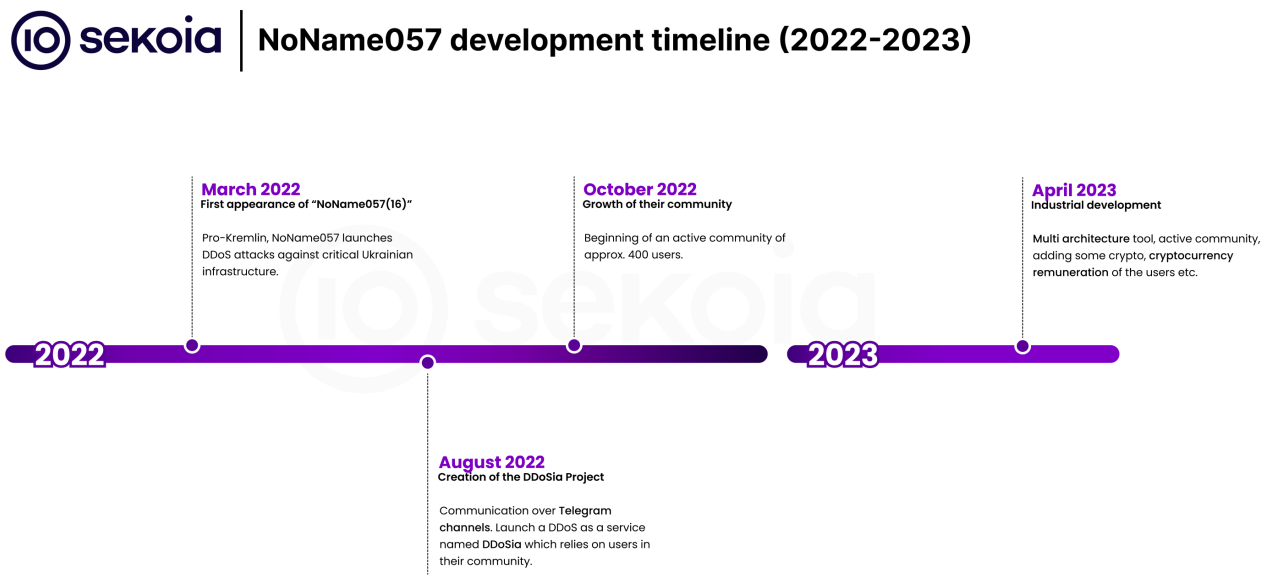


Figure 1. Development timeline of NoName057(16) and DDoSia Project (click to enlarge)

The administrators of the group as well as the community are very active. They were notably observed conducting DDoS attacks against European, Ukrainian, and U.S. websites of government agencies, media, and private companies. Regularly, the group posts messages claiming successful attacks.

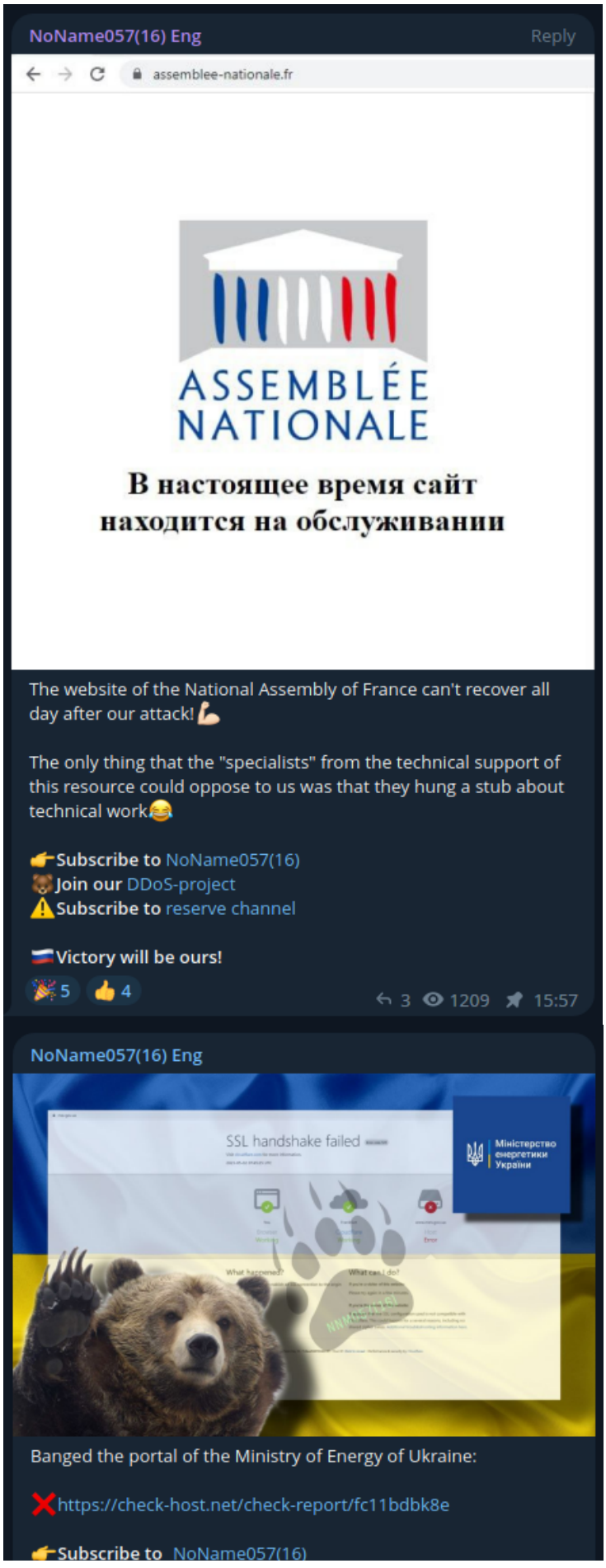
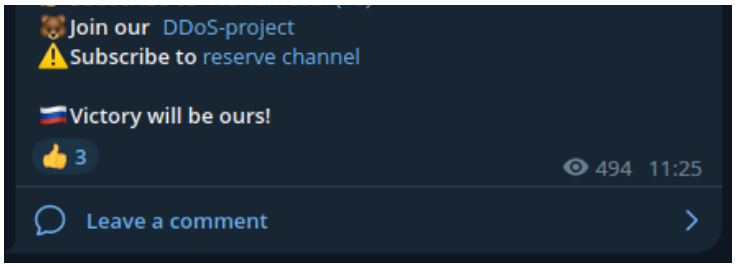


Figure 2. Message published on the



NoName057(16) Telegram group, claiming a successful attack against the the French National Assembly and the Ukrainian Cabinet of Ministers websites

DDoSia was initially written in Python using CPU threads as a way to launch several network requests at the same time. Since the first version, DDoSia relied on HTTP protocol for Command & Control (C2) communication, with JSON configurations distributed by the C2 server, and is available for several operating systems. On 18 April 2023, Avast published an article analyzing network flow between DDoSia users and the C2. On 19 April 2023, **DDoSia administrators released a new version of their sample that implements an additional security mechanism to conceal the list of targets**, which is transmitted from the C2 to the users. Said mechanism is described in the next section.

How DDoSia Project work

Overview of channels used

DDoSia's main communication occurs via the NoName057(16)'s Telegram channel, with one channel in Russian, counting more than 45,000 subscribers, and a second in English. Users can join the DDoSia Project group with the link hxxps://t[.]me/+fiTz615tQ6BhZWFi, gaining access to 7 different channels. NoName057(16) set up a separate Telegram bot from the DDoSia Projects group, available at hxxps://t[.]me/DDosiabot, which allows interaction via predefined commands. A summary of these channels is available in the Figure 3 below:

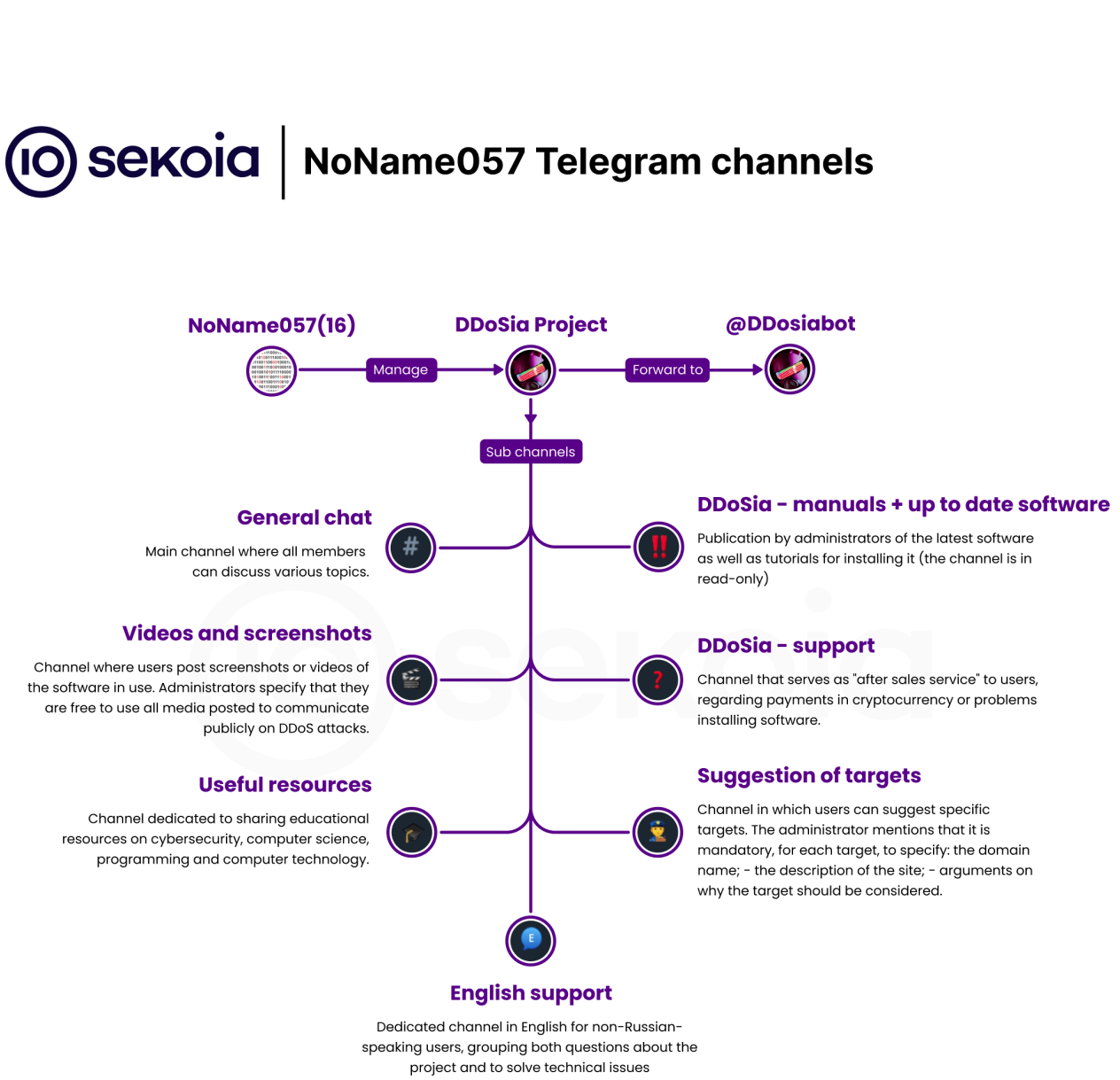


Figure 3. List of active channels of NoName057(16) and DDoSia Project (click to enlarge)
This figure includes an English translation of the channels, originally in Russian.

Register and download sample

The channel DDoSia – manuals + up-to-date malware includes a manual on the actions that need to be carried out. The first step is to register via the Telegram bot @DDosiabot. Although dedicated channels for English support exist, the bot is only available in Russian.

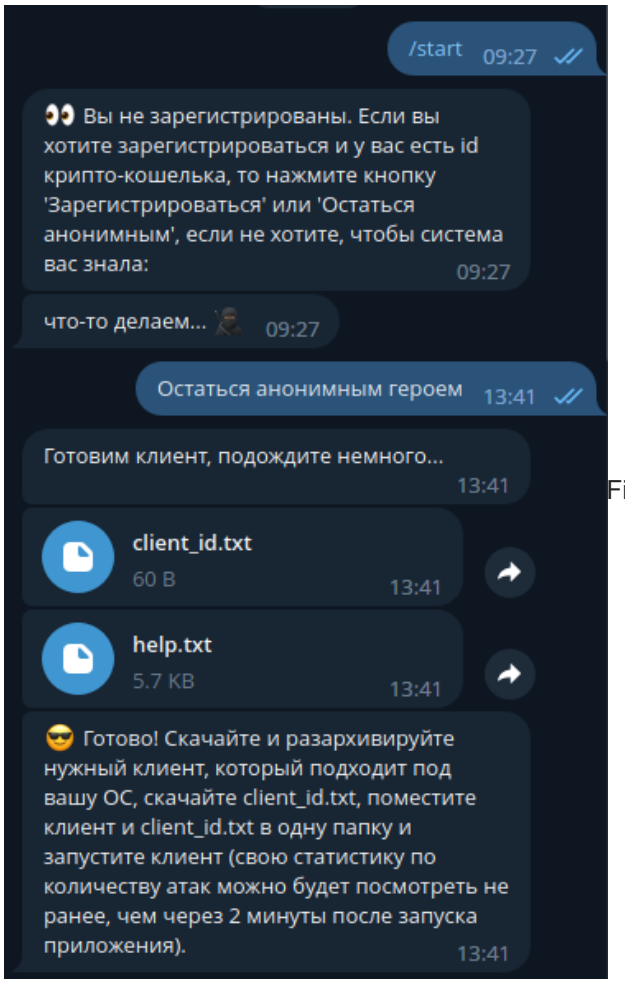


Figure 4. Screenshot of the chat with @DDosiabot

After starting the discussion with the /start command, the bot requires a TON wallet to receive cryptocurrency. As specified in the tutorials presented by the administrators, it is possible to create a TON wallet from a Telegram Bot named @CyptoBot.
Of note, no wallet was provided for this investigation. The bot then transmits two files:

- client_id.txt: a file containing information to uniquely identify a user. This is a hash starting with $2a$16, generated by a Bcrypt password-hashing function;
- help.txt: a file containing several indications on the steps to follow to use the sample as well as Telegram links for installation tutorials.

In addition, one of the bot's functionalities allows to view statistics of its own account as well as those of all bot users combined. It is also possible to ask to recreate the client_id.txt file.

Next step is to retrieve the sample to launch.

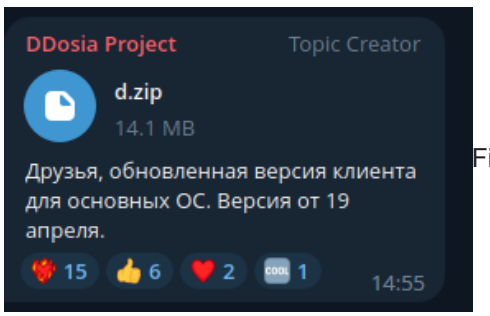


Figure 5. Screenshot of the sample

As shown in Figure 5, this is an archive in ZIP format, named d.zip, containing the client sample. This investigation focuses on the archive released on 19 April 2023. The summary of its contents is available below:

Filename	Filetype
d_linux_amd64	ELF 64-bit LSB executable, x86-64
d_linux_arm	ELF 32-bit LSB executable, ARM
d_mac_amd64	Mach-O 64-bit x86_64 executable
d_mac_arm64	Mach-O 64-bit arm64 executable
d_windows_amd64.exe	PE32+ executable (console) x86-64 for Microsoft Windows
d_windows_arm64.exe	PE32+ executable (console) Aarch64 for Microsoft Windows

Table 1. Summary of the content of the ZIP file

Execute the sample

Once the user has all the necessary files to participate in DDoS attacks, the client_id.txt file must be placed in the same folder as the selected executable. In this example, Sekoia.io analysts used d_windows_amd64.exe. Once the sample is executed, it is a command line prompt, in which it is possible to see the current number of targets, as well as a summary of the network interactions carried out towards a target. The English translation of the command line is as follows:

Go-Stresser версия 1.0 | PID 5420 © NoName057(16)

Authorization passed successfully
Received targets: 54
Successful responses (http code 200): 0
Total responses received: 565
Total requests sent: 1432

DDoSia Project's analysis

After downloading the necessary files, Sekoia.io analysts set up a dedicated infrastructure to retrieve the list of targets.

Network interactions

After setting up the infrastructure, we performed network sniffing to check what requests were sent between the client and the C2. The summary of network flow is available in the diagram below:

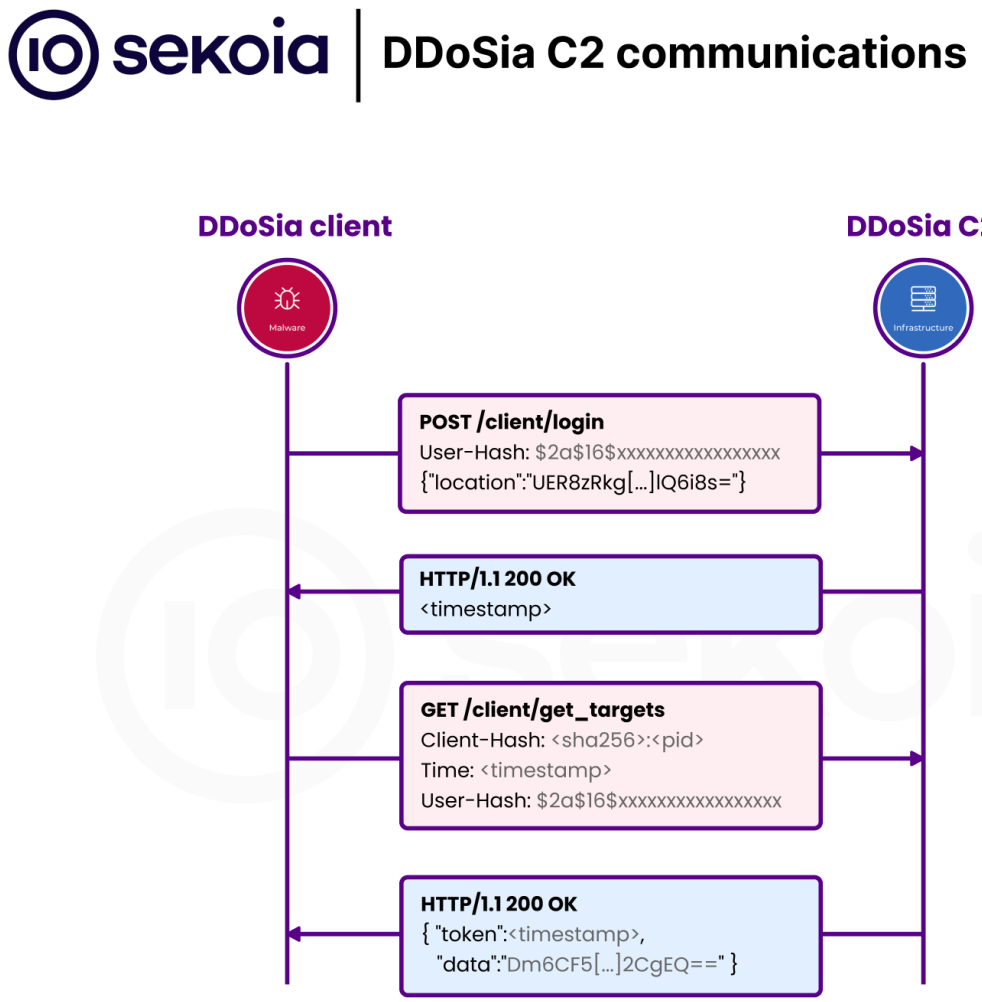


Figure 6. Summary of the network flow between DDoSia user and DDoSia C2
When the malware is launched, it makes a POST request to the URL hxxp://[IP]/client/login to authenticate with the C2. The User-Hash field corresponds to the content of the client_id.txt file, starting with $2a$16$;

The Client-Hash field is a value generated by the sample, which contains the SHA256 sum of the machine's UID, as well as the PID of the malware. This value is located in a folder located in the same location as the executable, in a folder named uid.

POST /client/login HTTP/1.1
Host: 94[.]140.114.239
User-Agent: Go-http-client/1.1
Content-Length: 251
Client-Hash: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx:xxxx
Content-Type: application/json
User-Hash: $2a$16$xxxxxxxxxxxxxxxxx
Accept-Encoding: gzip

{"**location**": "UER8zRkg[…]lQ6i8s="}
The C2 then confirms the authentication request and provides a token to the client, as below:

HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Tue, 25 Apr 2023 19:04:09 GMT
Content-Type: text/plain; charset=utf-8
Content-Length: 19
Connection: keep-alive
Vary: Origin
Access-Control-Allow-Origin:
Access-Control-Allow-Credentials: true
Access-Control-Expose-Headers: Link

1682xxxxxxxxxxxxx
Consequently, the client sends a GET request to the C2 hxxp://[IP]/client/get_targets, this time specifying the Time field, whose value is the one previously sent by the C2, automatically modified by the client.

GET /client/get_targets HTTP/1.1
Host: 94[.]140.114.239
User-Agent: Go-http-client/1.1
Client-Hash: xxxxxxxxxxxxxxxxxxxxxxxxxxxx:xxxx
Content-Type: application/json
Time: 1682xxxxxxxxxxxxx
User-Hash: $2a$16$xxxxxxxxxxxxxxxxx
Accept-Encoding: gzip

This time, the C2 returns a dictionary in JSON format. On one hand the previous but modified token, and on the other hand a data field in which there is an encrypted text. This field contains the list of targets:

HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Tue, 25 Apr 2023 19:04:15 GMT
Content-Type: text/plain; charset=utf-8
Content-Length: 69595
Connection: keep-alive
Vary: Origin
Access-Control-Allow-Origin:
Access-Control-Allow-Credentials: true
Access-Control-Expose-Headers: Link

{“**token**“: 1682xxxxxxxxxxxxx, “**data**“: “Dm6CFMc9Lk4wrY2[…]XW2ZqF2CgzTboVEQ==”}
In this example, the value of the data fields is shortened, as its size is around 70, 000 characters. The next section provides further details related to data encryption’s mechanism.

Reverse engineering on the sample

At this stage, the retrieved list of targets is encrypted. This reverse engineering analysis focuses on the d_windows_amd64.exe executable.

This version of DDoSia was written in Go language. Contrary to usually seen Go binaries, this new version does not provide the expected result and decompilation errors are observed. Focusing on the functions performing the HTTP requests and decryption process results in the following graph:

Figure 7. Graph of instructions after decompilation in IDA software

In this graph, brown parts correspond to instructions which are not considered in a function, which means that it is not possible to interpret them. Despite this, several functions seem relevant. The first interesting function initiates a structure where the IPv4 address and different URLs are called:

```
__int64 __fastcall main_function(__int64 *p_int6470, __int64 a2)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  if ( (unsigned __int64)v29 <= *(_QWORD *)(v3 + 16) )
    j_runtime_morestack();
  v25 = sub_6C0280((__int64)p_int6470, a2);
  v27 = v5;
  v23 = p_int6470;
  main_obj = (main_struct *)runtime_newobject();
  main_obj->qword20 = 3LL;
  main_obj->qword0 = 1LL;
  main_obj->qword8 = 4LL;
  main_obj->qword18 = 70LL;
  main_obj->qword10 = 13LL;
  main_obj->qword28 = 60LL;
  main_obj->size_ip = 21LL;
  main_obj->str_ip = "http://94.140.114.239";
  main_obj->size_client_login_uri = 13LL;
  main_obj->str_client_login_uri = "/client/login";
  main_obj->size_set_attack_count_uri = 24LL;
  main_obj->str_set_attack_count_uri = "/client/set_attack_count";
  main_obj->size_get_target_uri = 19LL;
  main_obj->str_get_target_uri = "/client/get_targets";
  main_obj->qword78 = v2;
  if...
  main_obj->qword88 = v23;
  if...
  if...
  v24 = main_obj;
  runtime_GOMAXPROCS();
  // Init GoStresser
  init_GoStresser();
  // Try to authenticate
  if ( MakeClientLogin((__int64)p_int6470, a2, v7) )
  {
    // If Authentication OK
    print((__int64)main_obj, v20);
    GetTargets(1LL, 1LL);
```

Figure 8. Initiates a structure which contains IPs and URLs to request

The Figure 9 is an extract of the GetTargets function, where a GET request is created and then sent. After unmarshalling the JSON, the Decrypt_AESGCM function is called. It makes the authentication via the MakeClientLogin function, which, if successful, retrieves the targets.

```
// Create GET Request http://94.140.114.239/client/get_targets
Create_GET_Request(2LL, 2LL, v7, pGlobal_Mainobj);
// Try to unmarshall the json
if ( encoding_json_Unmarshal("\b", v44) )
{
  // If Unmarshall fails
  v47 = v1;
  v48 = v1;
  v49 = v1;
  return v1;
}
else
{
  // Else, we can decrypt the content of the response
  v8 = v44;
  Global_Mainobj[1].qword0 = *v44;
  v9 = v8[3];
  Decrypt_AESGCM(v9, pGlobal_Mainobj->int6470, pGlobal_Mainobj, v8[2]);
```

Figure 9. Initiates the GoStresser tool authentication

This Figure 10 contains two functions which were automatically renamed crypto/AES.NewCipher and crypto/AES.newGCMWithNonceAndTagSize. Their purpose is to initiate the AES encryption.

```
// Append the User-Hash and the Token
size_NewString = fmt_Sprintf(2LL, 2LL, v10, &v28);
runtime_stringtoslicebyte(2LL, 2LL, v12, 4LL);
if ( size_NewString <= 32 )
  sizeToRead = 0LL;
else
  sizeToRead = size_NewString - 32;
if ( sizeToRead > size_NewString )
  sub_465C20(2LL, 2LL, sizeToRead, size_NewString);
v16 = size_NewString - sizeToRead;
// move the ptr to the beginning of the key
ptr_NewKey = ptr_end_of_NewString - sizeToRead;
// https://pkg.go.dev/crypto/aes
// func NewCipher(key []byte) (cipher.Block, error)
// => The key is passed as a parameter (rcx)
crypto_aes_NewCipher();
if ( v19 )
  return 0LL;
// Tag size : 16
// Nonce size : 12
AEAD = crypto_cipher_newGCMWithNonceAndTagSize(16LL, ptr_NewKey, v18, 12LL);
if ( v21 )
  return 0LL;
v26[9] = v16;
v22 = AEAD->NewGCM();
if ( v26[0] )
{
  if ( v22 > a1 )
    sub_465BE0(16LL, ptr_NewKey, a1, v22);
  if ( v22 > v26[0] )
    sub_465C20(16LL, ptr_NewKey, a1, v26[0]);
  // https://go.dev/src/crypto/cipher/gcm.go
  // Open(dst, nonce, ciphertext, data []byte)
  // => Decrypt the data. IV is passed as a parameter
  return AEAD->Open(0LL, v27, v27 + (v22 & ((v22 - a1) >> 63)), 0LL);
```

Figure 10. Initiate the AES encryption

This first step allowed Sekoia.io analysts to identify that data are AES-GCM encrypted. As is, finding the generation process of the key and of the IV are difficult to understand. To bypass this step, it was decided to use a dynamic analysis approach of the sample.

As a reminder, the client receives a JSON with two fields: an integer, named token and a base64 encoded field, named data. Dynamic analysis allowed for the calculation of all necessary values to decrypt the data:

Key calculation:

- The value of the token is divided it by 5 (whole division);
- The result is added to the User-Hash (that begins by $2a$16$);
- Take the last 32 characters of the User-Hash and convert them in a hex string.

IV calculation

- Take the ciphertext, decode it in base64;
- Take the 12 first characters and convert them in bytes.

TAG calculation

- Take the ciphertext, decode it in base64;
- Take the 16 last characters convert them in bytes.

Finally, the ciphertext corresponds to the value of the data field, from which the first 12 and last 16 chars are removed. Now it is possible to get the value of the data field in plain text.

Analysis of the decrypted content

Once the data is decrypted, it is possible to see that it is a dictionary in JSON format.
The dictionary is divided into two parts. The first field is called randoms, and the second field is called targets.
Targets field contains an integer array of fields, in which several of them are specific:

Targets field

{
“**target_id**“:”645026fc0c81901a3b3aa4f5”,
“**request_id**“:”645026fd0c81901a3b3aa4f6”,
“**host**“:”id[.]kyivcity.gov.ua”,
“**ip**“:”104[.]18.20.41”,

“**type**“:”http”,
“**method**“:”POST”,
“**port**“:443,
“**use_ssl**“:true,
“**path**“:”/login/email”,
“**body**“:{
“**type**“:”str”,
“**value**“:”login=$_1%40gmail.com\u0026password=$-1\”
},
“**headers**“:null
}

In addition to the IPv4 address, there are fields to target specific URLs. On some targets, it is possible to find that beyond the metadata, content can be added to the DDoS request thanks to the body field, as shown in the JSON data above.

Randoms field

This table contains a list of fields that appear to be used to generate random strings in sent requests.

{
“**name**“:”Все персонажи 6-12”,
“**id**“:”62d8fccfb44b5774ee96ec0a”,
“**digit**“:true,
“**upper**“:true,
“**lower**“:true,
“**min**“:6,
“**max**“:12
}

In the target example above, in the body[“value”] field, we can find variables such as $_1 or $-1 that appear to be replaced by these random strings. It is highly likely that this random data generation allows to bypass the cache mechanisms of the C2 target by making network requests different from each other.

Analysis of targeted websites and countries

After the values sent by DDoSia C2 server were successfully decrypted, TDR analysts developed a tool **automatically gathering targeted domains**, allowing a victimology analysis. The following section analyzes the data, over a **period from 8 May to 26 June 2023**.

The following graph shows the most targeted countries, based on the TLD of the targeted urls. Commercial or domains unrelated to a country-level TLD are excluded (.com, .info, .net, .org, .space).

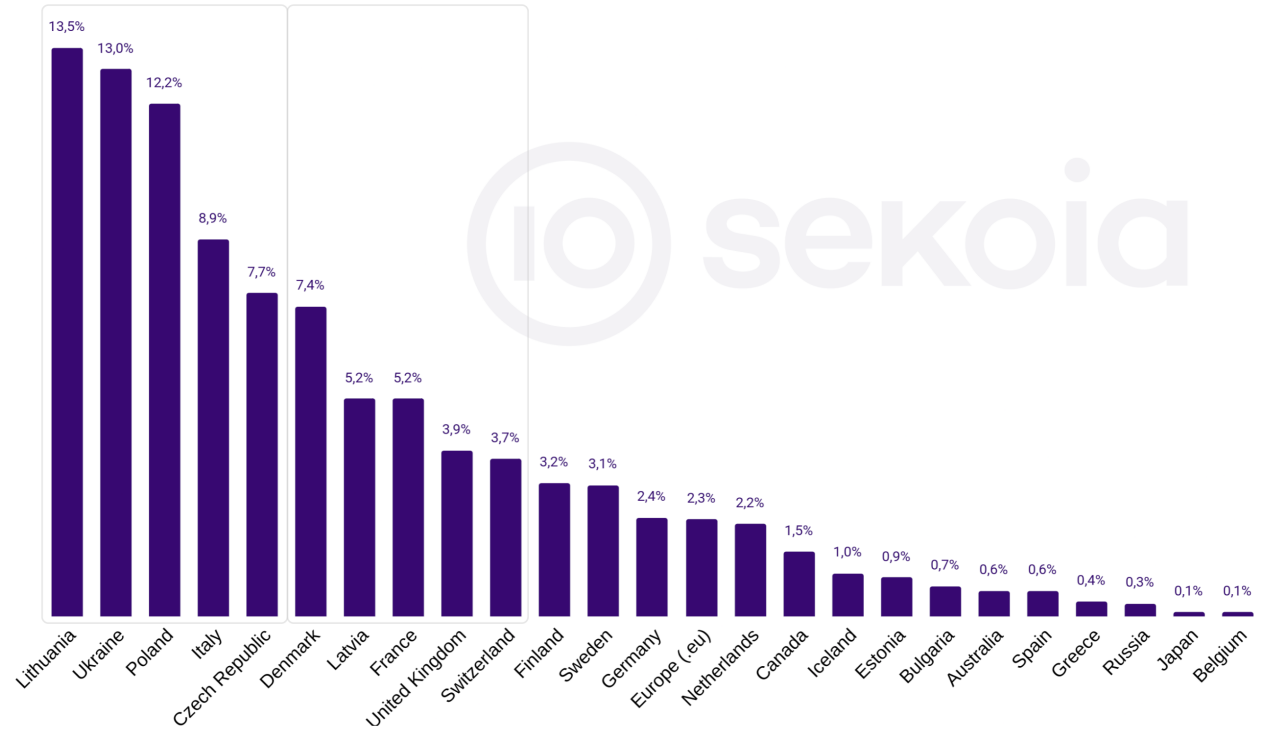
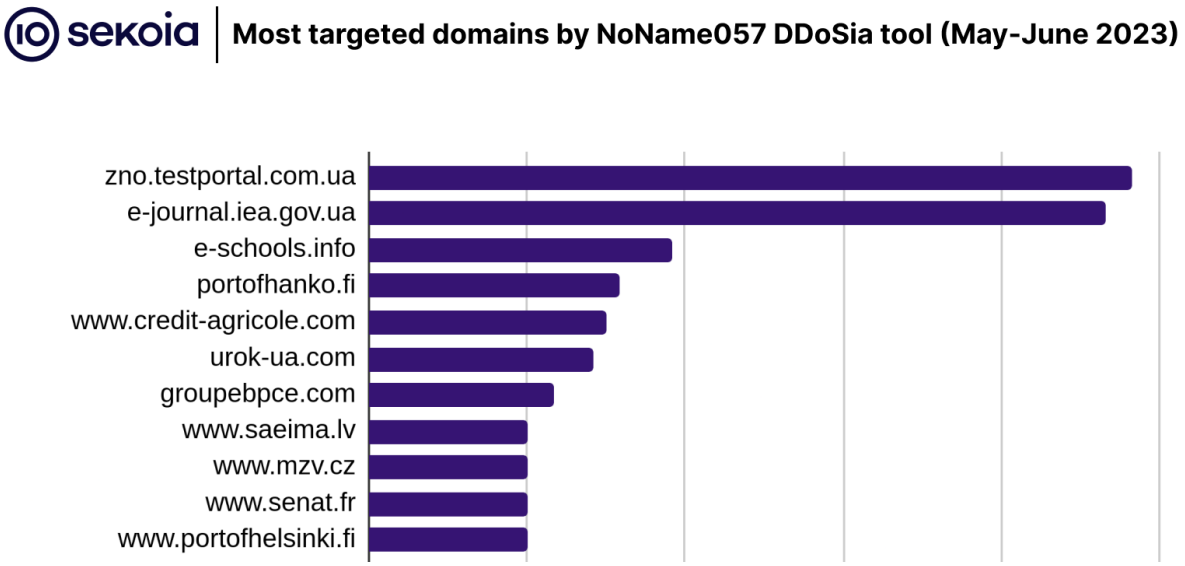


Figure 11. Percentage of top-level domain by targeted countries **(click to enlarge)**
Based on this graph, we clearly identify that the pro-Kremlin hacktivist group NoName057(16), **primarily focuses on** Ukraine and NATO countries, including the Eastern Flank (Lithuania, Poland, Czech Republic and Latvia). It is highly likely that this stems from the fact that those countries are the most vocal in public declarations against Russia and pro-Ukraine, as well as providing military support and capabilities.

A second group, mostly Western countries, is the **secondary DDoSia target**, including France, the United Kingdom, Italy, Canada and other EU countries, almost certainly as they supported Ukraine both politically, militarily and economically since the beginning of the conflict.

Sekoia.io in-house tool detected a total of 486 different websites impacted. The following graph shows the top 50:



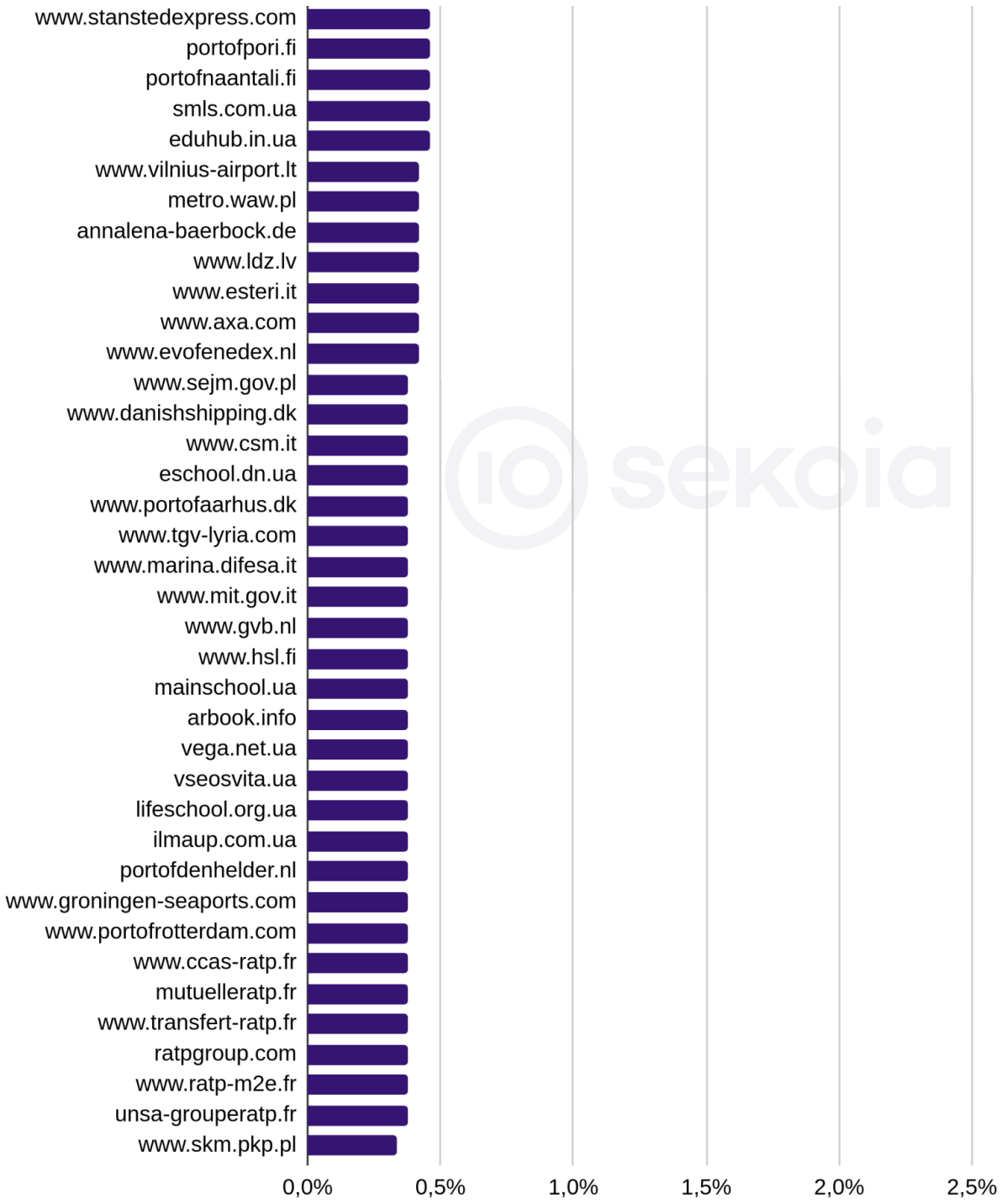


Figure 12. **Top 50 targeted websites (click to enlarge)**
From 8 May to 26 June 2023, few conclusions can be drawn:

The top 2 targets are Ukrainien websites, targeted twice as often as the others.The first victim; zno.testportal.com[.]ua is related to testportal.gov[.]ua, a state institution (Ukrainian Center for Educational Quality Assessment) which delivers external independent evaluation of students.

The second domain, e-journal.iea.gov[.]ua is a online education platform created by the Ukrainian government to face COVID restrictions.

Based on this statistical observation, Sekoia.io analysts assess it is plausible that NoName057(16) targeted education-related resources during the exam period (May and June), to maximize the media coverage of their DDoS operation.

Among the other impacted domains, we identify **multiple economic sectors**, including education, financial and transport sectors, as well as governmental entities. Indeed, two of the targets within the top 10 are related to the financial sector; the AXA bank (top 5) and the BPCE group (top 7). Public entities such as the French Senate or the Italian government can also be found among the most targeted websites. Similarly, few domains belonging to the French transport group RATP were equally actively targeted.

PMC Wagner websites targeted

Sekoia.io analysts observed that the only targets throughout the day of the 24 June 2023 were wagnercentr[.]ru and wagner2022[.]ru, congruent with the attempted offensive from the Wagner group in Russia. This is the first observed attack against one single victim, as the NoName057(16) group usually targets an average of 15 different victims per day. Another considerable difference can be noted, while they usually do so for other victims, the attackers did not communicate about the attack on their Telegram channel.

As a nationalist hacktivist group, NoName057(16) is very **reactive to political communication**. For example, on 21 June 2023, shortly after French president Macron announced the incoming delivery of air defense system to Kiev, our tool detected multiple targets related to the French transport group RATP, targeting the following websites: www.ratp[.]fr, www.ratp-m2e[.]fr, mutuelleratp[.]fr, www.cgt-ratp[.]fr, www.transfert-ratp[.]fr. This reaction likely reflects NoName' stand to quickly and systematically conduct their campaigns as retaliation to what they perceive being a provocation or an offense to Russia.

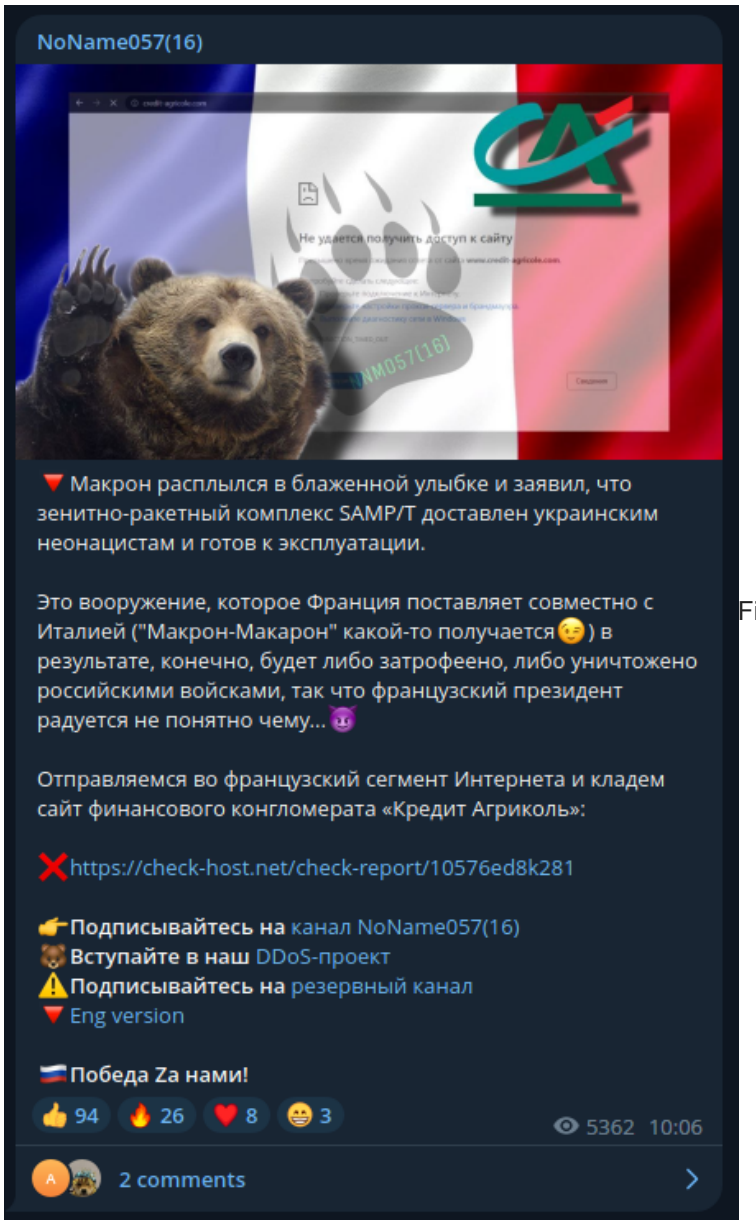


Figure 13. Screenshot of a Telegram message from NoName057(16) channel, published on 21 June 2023

The English translation of the body of this message is as follows:

🔻Macron smiled blissfully and announced that the SAMP/T anti-aircraft missile system was delivered to Ukrainian neo-Nazis and is ready for use.
This weapon, which France supplies jointly with Italy (“Macron-Macaron” is something like that😉) as a result, of course, will be either taken away or destroyed by the Russian troops, so the French president is happy about it… 😈

Conclusion

The NoName057(16) group continues to update the DDoSia Project. Sekoia.io analyst’s observations concur with Avast’s analysis and provide an update on the newly implemented encryption mechanism.

NoName057(16) is making efforts to make their malware compatible with multiple operating systems, almost certainly reflecting their intent to make their malware available to a large number of users, resulting in the targeting of a broader set of victims.

Sekoia.io analysts assess that strengthening the security of their software is part of NoName057(16)'s efforts to continuously develop their capabilities, almost certainly driven by their active community as well as the increasing scrutiny of their activities from the CTI community. It is highly likely we will observe further developments in the short term.

Indicators Of Compromise (IoCs)

IoC Name	Info	SHA256 sum
d_linux_amd64	DDoSia malware	761075da6b30bb2bcbb5727420e86895b79f7f6f5cebdf90ec6ca85feb78e926
d_linux_arm	DDoSia malware	fae9b6df2987b25d52a95d3e2572ea578f3599be88920c64fd2de09d1703890a
d_mac_amd64	DDoSia malware	8e1769763253594e32f2ade0f1c7bd139205275054c9f5e57fefd8142c75441f
d_mac_arm64	DDoSia malware	9a1f1c491274cf5e1ecce2f77c1273aafc43440c9a27ec17d63fa21a89e91715
d_windows_amd64.exe	DDoSia malware	726c2c2b35cb1adbe59039193030f23e552a28226ecf0b175ec5eba9dbcd336e
d_windows_arm64.exe	DDoSia malware	7e12ec75f0f2324464d473128ae04d447d497c2da46c1ae699d8163080817d38
94[.]140.114.239	DDoSia C2	N/A

Chat with our team!

Would you like to know more about our solutions?
Do you want to discover our XDR and CTI products?
Do you have a cybersecurity project in your organization?
Make an appointment and meet us!

Contact us

Thank you for reading this blogpost. **We welcome any reaction, feedback or critics about this analysis. Please contact us on tdr[at]sekoia.io**

Feel free to read other TDR analysis here :

Comments are closed.