

# DIGITAL DISRUPTION:

*Hactivism and DDoS*

DDoS Workshop ADC 2023  
November 27<sup>nd</sup>, 2023

Martijn Peijer  
SOC Belastingdienst



Image Source: Utopia.Fans

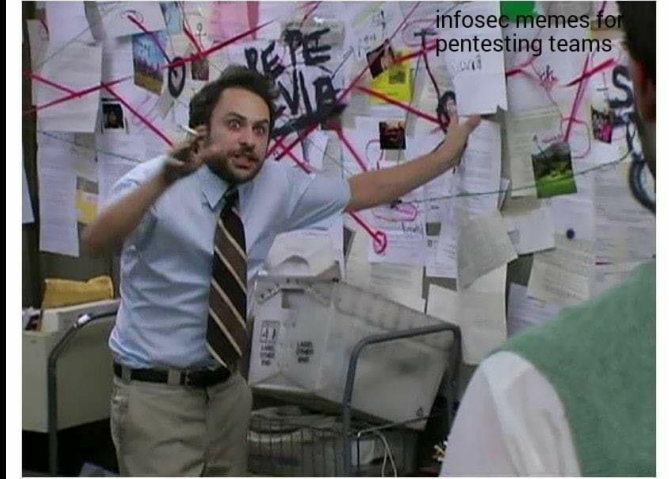


## ~ whoami

- Martijn Peijer
- Security Analyst & Ethical Hacker
- RED-Team - SOC - Belastingdienst
  
- Threat Hunting
- Vulnerability Scanning
- CVDs/RDs
- Hacking
- DDoS Tests
- CTI
- OSINT



When the government tells you to work from home but you're on the red team so you're not sure whose home you're supposed to work from.



:funny.co

THEY CAN'T DDOS YOU



IF YOU DDOS YOURSELF

makeameme.org

When you use CTRL + C instead of copying using right click



# CONTENTS

- **Hactivism, Targets & Methods**
- DDoS Attack Types & Botnets
- Hactivist Groups, Attacks & Cyberwarfare
- Anonymous Sudan, Killnet & Dark Parliament
- NoName057(16)
- Technical Analysis of DDoSia Client
- Red Cross Rules of Engagement for Hackers
- What can you do?
- Questions





# HACKTIVISM

## Hacktivist:

*“A person who gains unauthorized access to computer files or networks in order to further social or political ends.”*



Image Source: X (@HackTheHague)

# CYBERTERRORISM

*“Uses the same tactics and methods as hacktivists, but to cause fear, harm, or disruption. In severe cases, cyberterrorism may involve targeting critical infrastructure or systems to cause loss of life or severe economic damage.”*



Image Source: AFMP

# HACKTIVISM TARGETS

- Political
- Social
- Religious
- Anarchist



Image Source: Fruugo NL

# HACTIVISM TARGETS BY SECTOR

Top industries attacked by Hactivist Alliance

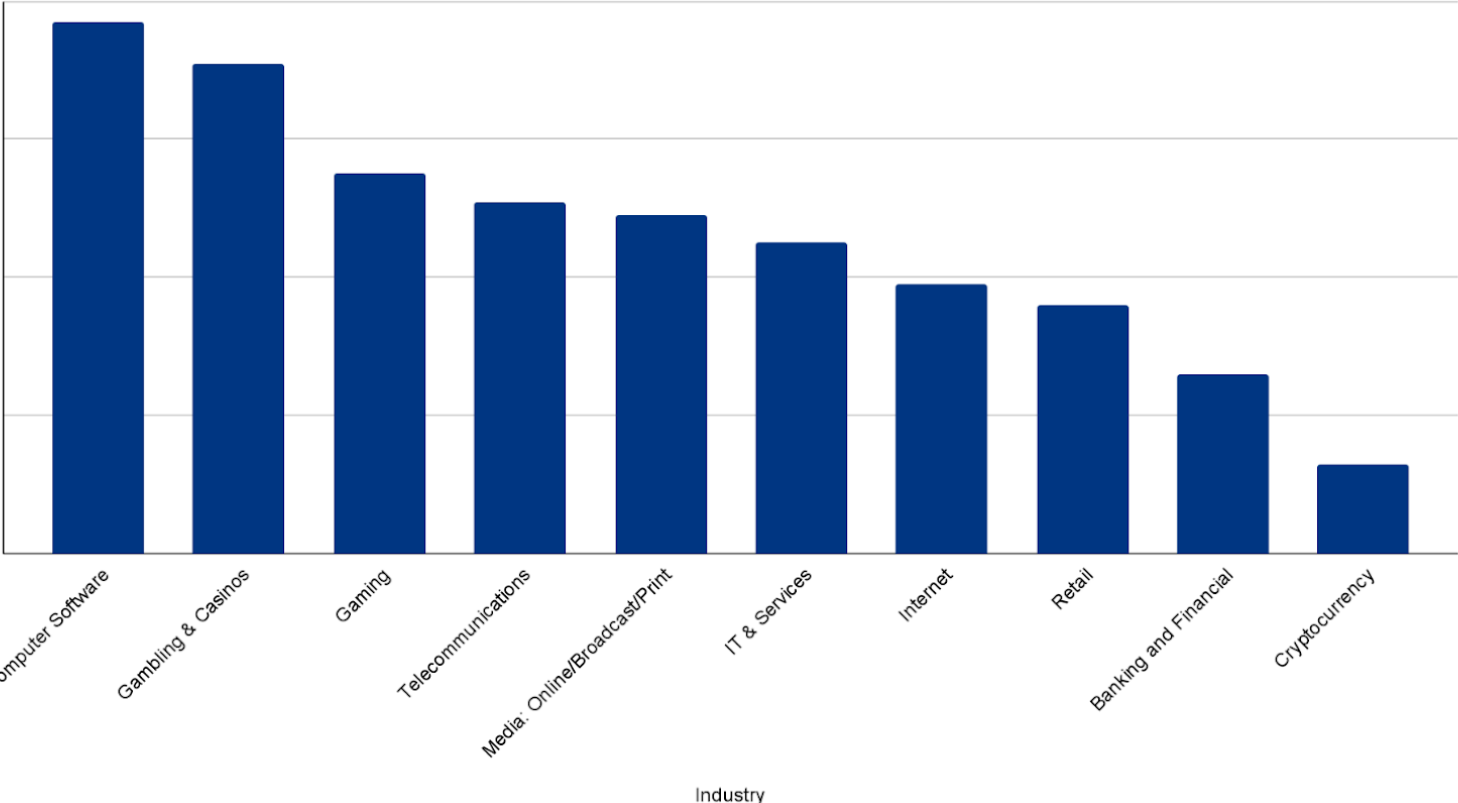


Image Source: Cloudflare

# HACKTIVISM METHODS

- Doxing
- Anonymous blogging
- Informational leaks
- Geo-bombing
- Website mirroring
- Defacement (code changing)
- DoS and DDoS Attacks



Image Source: iStockPhoto



# CONTENTS

- Hacktivism, Targets & Methods
- **DDoS Attack Types & Botnets**
- Hacktivist Groups, Attacks & Cyberwarfare
- Anonymous Sudan, Killnet & Dark Parliament
- NoName057(16)
- Technical Analysis of DDoSia Client
- Red Cross Rules of Engagement for Hackers
- What can you do?
- Questions



# DDOS ATTACK TYPES BY HACKTIVIST ALLIANCES

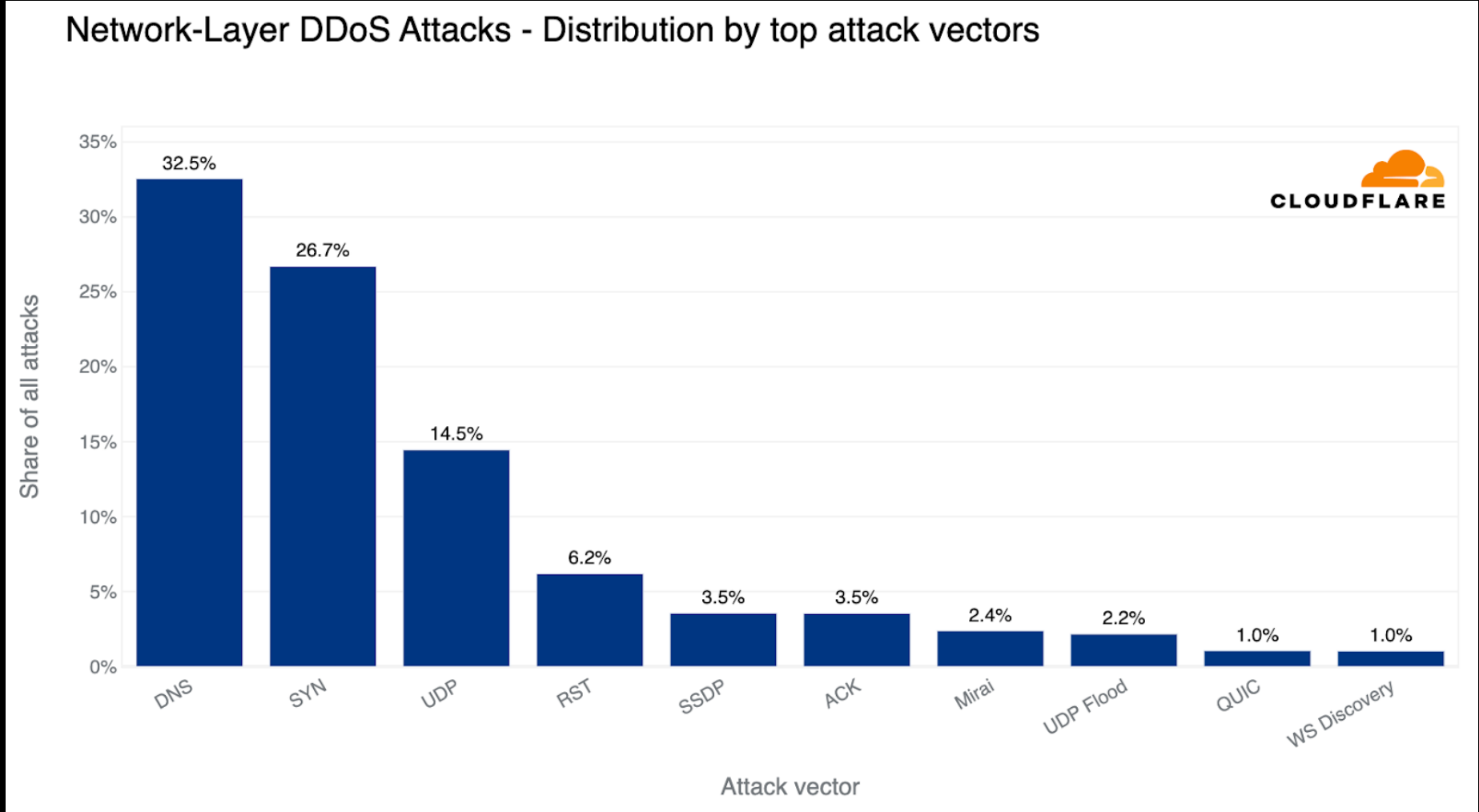
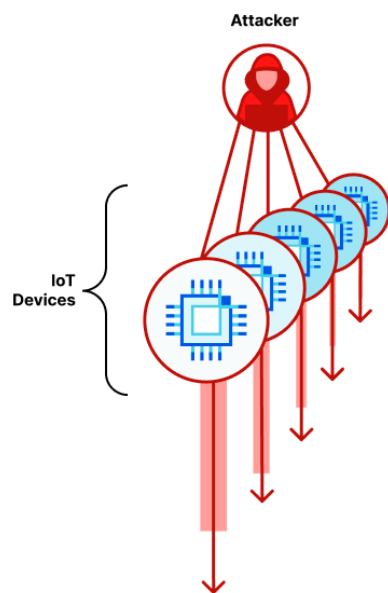


Image Source: Cloudflare

# DDOS BOTNETS

IoT-based botnet attack



VPS-based botnet attack

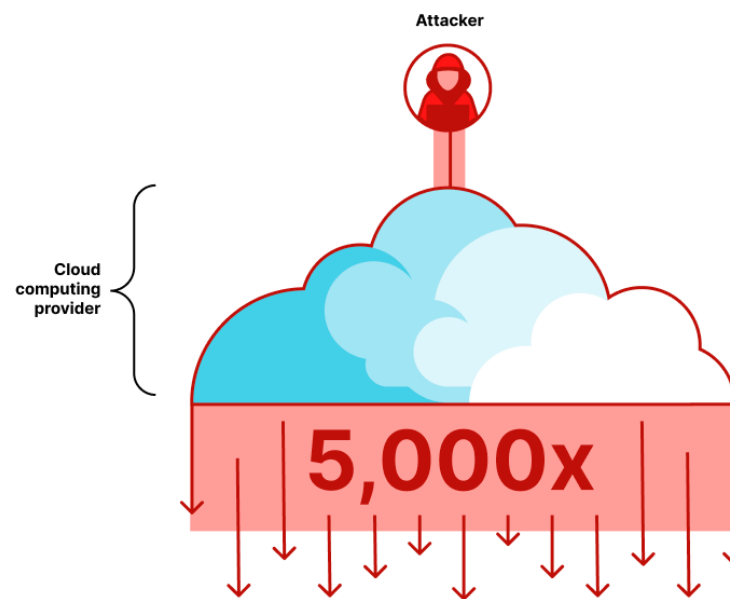


Image Source: Cloudflare

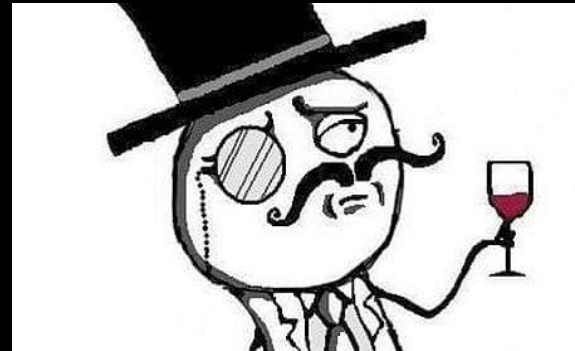
# CONTENTS

- Hacktivism, Targets & Methods
- DDoS Attack Types & Botnets
- **Hacktivist Groups, Attacks & Cyberwarfare**
- Anonymous Sudan, Killnet & Dark Parliament
- NoName057(16)
- Technical Analysis of DDoSia Client
- Red Cross Rules of Engagement for Hackers
- What can you do?
- Questions





# HACKTIVIST GROUPS



# LARGE ATTACKS

- **2008, Anonymous:**  
Attack on Church of Scientology
- **2010, WikiLeaks:**  
Exposure of Afghanistan & Iraq War documents
- **2011 & 2017, Anonymous:**  
Operation Darknet
- **2011, LulzSec:**  
Attack on Sony
- **2013, Syria Electronic Army:**  
Fake news through defacement
- **2022, Anonymous:**  
Attacks on Russia
- **2023: NoName:**  
Attacks on the West

# CYBERWARFARE

- More and more groups joining the scene
- 35 Pro-Palestinian, 4 Pro-Israelian
- Russian groups joining
- DDoS, Data Breaches, Doxing & Defacing



Image Source: Universiteit van Amsterdam

# CONTENTS

- Hacktivism, Targets & Methods
- DDoS Attack Types & Botnets
- Hacktivist Groups, Attacks & Cyberwarfare
- **Anonymous Sudan, Killnet & Dark Parliament**
- NoName057(16)
- Technical Analysis of DDoSia Client
- Red Cross Rules of Engagement for Hackers
- What can you do?
- Questions





# ANONYMOUS SUDAN

- Formed January 2023
- Targeting countries and organizations
- “Anti-muslim activity”
- Seemingly also Russian members
- Usually mentions upcoming targets on Telegram

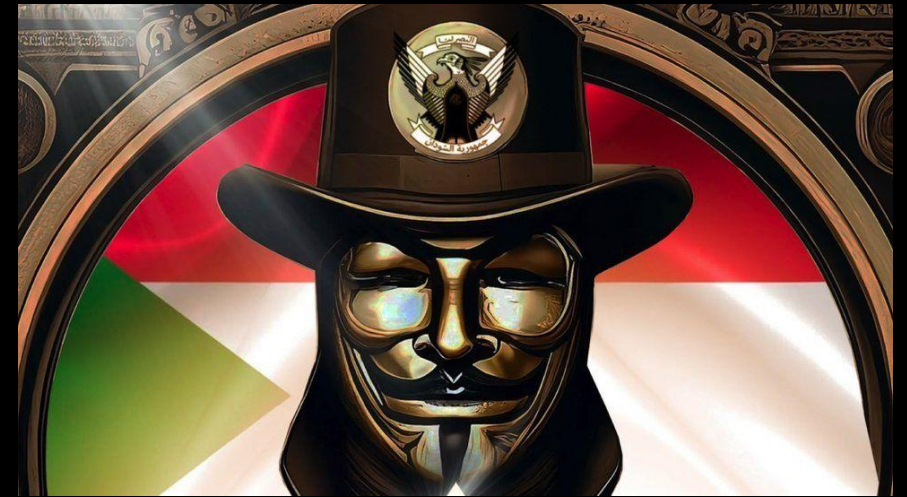


Image Source: BBC

## Notable attacks:

- Feb. 2023: Sweden & Denmark for burning Qurans
- Mar. 2023: Australian universities, hospitals and airports
- Apr. 2023: Israeli websites for military activity in Palestine
- Jul. 2023: Fan-fiction site AO3 for LGBTQ+ and NSFW content

# KILLNET

- Formed March 2022
- Pro-Russian
- Attacking any country supporting Ukraine
- Usually after certain events (i.e. promising delivery of tanks or fighter jets)
- Close with Anonymous Sudan

## Notable attacks:

- Eurovision 2022
- Latvia's largest ever attack
- Lockheed Martin
- Dutch Health Sector in January 2023



Image Source: Medium.com

# DARKNET PARLIAMENT

## Consists of:

- REvil
- Anonymous Sudan
- Killnet
- & more...

## Large Attacks:

- SWIFT
- Microsoft

## Microsoft confirms Azure, Outlook outages caused by DDoS attacks

By [Lawrence Abrams](#)

June 18, 2023 10:40 AM 8



Microsoft has confirmed that recent outages to Azure, Outlook, and OneDrive web portals resulted from Layer 7 DDoS attacks against the company's services.

The attacks are being attributed to a threat actor tracked by Microsoft as Storm-1359, who calls themselves [Anonymous Sudan](#).

Source: BleepingComputer

# DARKNET PARLIAMENT ATTACKS

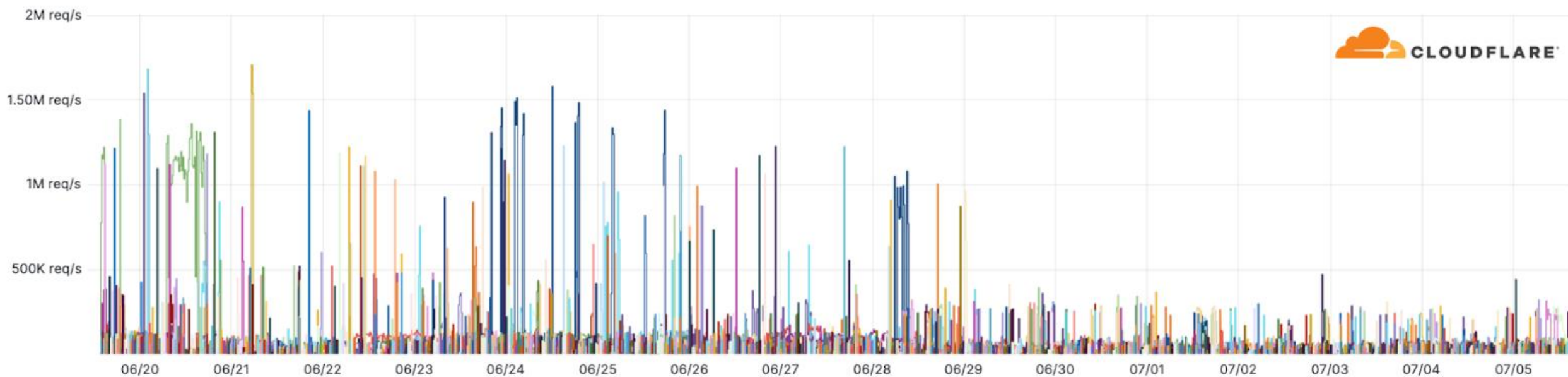


Image Source: Cloudflare



# CONTENTS

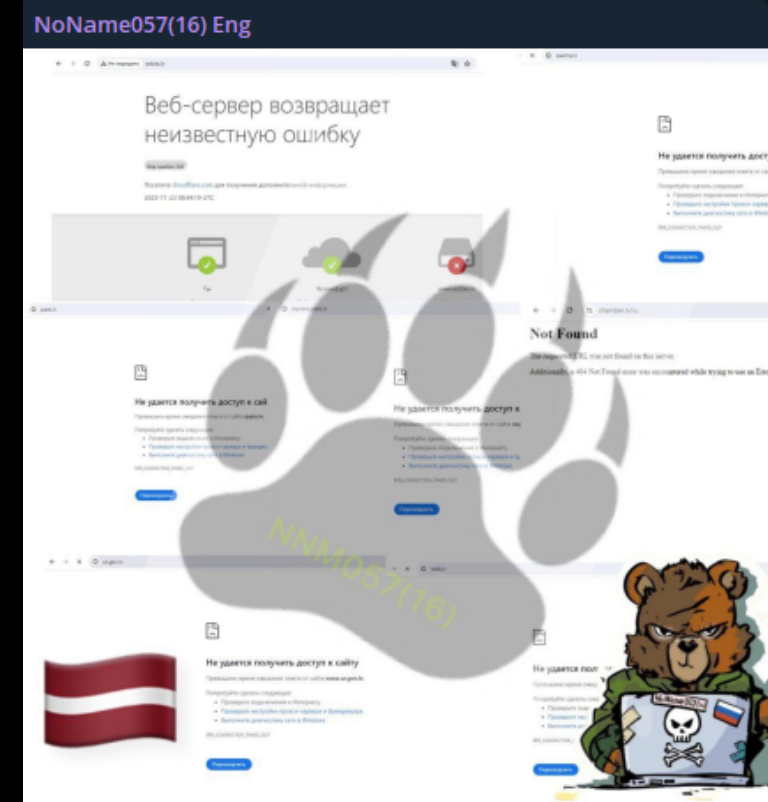
- Hacktivism, Targets & Methods
- DDoS Attack Types & Botnets
- Hacktivist Groups, Attacks & Cyberwarfare
- Anonymous Sudan, Killnet & Dark Parliament
- **NoName057(16)**
- Technical Analysis of DDoSia Client
- Red Cross Rules of Engagement for Hackers
- What can you do?
- Questions



# NONAME057(16)

- Formed March 2022
- 4 Telegram channels: Russian, English & 2 backup channels
- Partners: Killnet, Legion, NetKillnet, Beregini, NemeZida, HakNet
- First Bobik botnet, now the DDoSia Project
- NoName: ca. 52.000 members, DDoSia ca. 12.000 members
- Targets: mainly Estonia, Lithuania, Latvia and Poland. Ukraine less common

Image Source: Telegram @NoName



⚠ Speaker of the Latvian Seimas Daiga Mieriņa stated the need to continue supporting Kyiv, because her country daily feels a “hybrid threat from Belarus” on its borders.

In this case, we will continue to increase the number of our DDoS missiles fired towards Latvia 🇺🇦

The Russophobic authorities of the West must make it clear to themselves that the problems of their country must come before Zelensky's wishes!

✗ **Electronic document management and electronic signature service:**

[check-host.net/check-report/136f917ak539](https://check-host.net/check-report/136f917ak539)

✗ **Seimas of Latvia:**

[check-host.net/check-report/136f929dk4f5](https://check-host.net/check-report/136f929dk4f5)

✗ **Latvian Post:**

[check-host.net/check-report/136f94eekeda](https://check-host.net/check-report/136f94eekeda)

# NONAME MANIFESTO



## Manifesto NoName057(16)

Every action creates a reaction. An open information war is being waged against Russia. Western Russophobes, using the administrative, financial and technical resources of foreign states, carry out attacks on the infrastructure of the Russian Federation.

We do not intend to sit idly by and in response to their hostile, openly anti-Russian actions, we will respond proportionately. It is unacceptable for Russophobia to become the norm!

We will never harm the innocent and our actions are a response to the rash acts of all those who have taken an openly hostile position. We have enough knowledge, strength and experience to restore justice where it has been violated. We don't attack our own because of our beliefs. Our Motherland is our point of strength.

We do not work on commercial orders and do not settle scores between competitors.

We are ready to cooperate with hacker groups and "free shooters" who share our values listed in the Manifesto.

The strength is in the truth, and we stand on that!



Greetings, comrades!

The hacker group NoName057(16) is on the warpath with Ukrainian under-hackers and their corrupt henchmen!

These fans of the neo-fascists who seized power in Ukraine are trying to attack the Internet resources of our country and intimidate our compatriots with their attacks on social networks and other communication channels. In response to their miserable attempts, we are carrying out massive attacks on dire propaganda resources that blatantly lie to people about Russia's special operation in Ukraine, as well as on the websites of Ukrainian unfortunate hackers who are trying to support Zelensky's neo-Nazi regime and a handful of drug addicts and Nazis from his pack!

We have a number of successful attacks on Ukrainian resources behind us, as a result of which users' access to them was paralyzed. And this is just the beginning.

Enemies, we want to recall the words of the famous Russian commander Alexander Nevsky: "Whoever comes to us with a sword will die by the sword!"

Here we will talk about our cases and attacks.

Image Sources: Telegram @NoName



# NONAME'S DUTCH TARGETS

- 9292 OV
- OV-Chipkaart.nl
- Gvb.nl
- A-bike.nl
- Gemeente Vlaardingen
- Rederij-Doeksen.nl
- Port of Amsterdam
- Groningen Seaports
- Lelystad Airport
- SNS Bank
- PostNL
- Rijksoverheid.nl
- House of Representatives
- Rijkswaterstaat
- Rechtspraak.nl
- DIGID
- NCSC





# CONTENTS

- Hacktivism, Targets & Methods
- DDoS Attack Types & Botnets
- Hacktivist Groups, Attacks & Cyberwarfare
- Anonymous Sudan, Killnet & Dark Parliament
- NoName057(16)
- **Technical Analysis of DDoSia Client**
- Red Cross Rules of Engagement for Hackers
- What can you do?
- Questions



# DDOSIA PROJECT

- DDoSia – Manuals + Actual Software
- Suggestion of Targets
- DDoSia – Support
- General Chat
- Useful material
- English support
- Your videos and screenshots of working with the DDoSia client

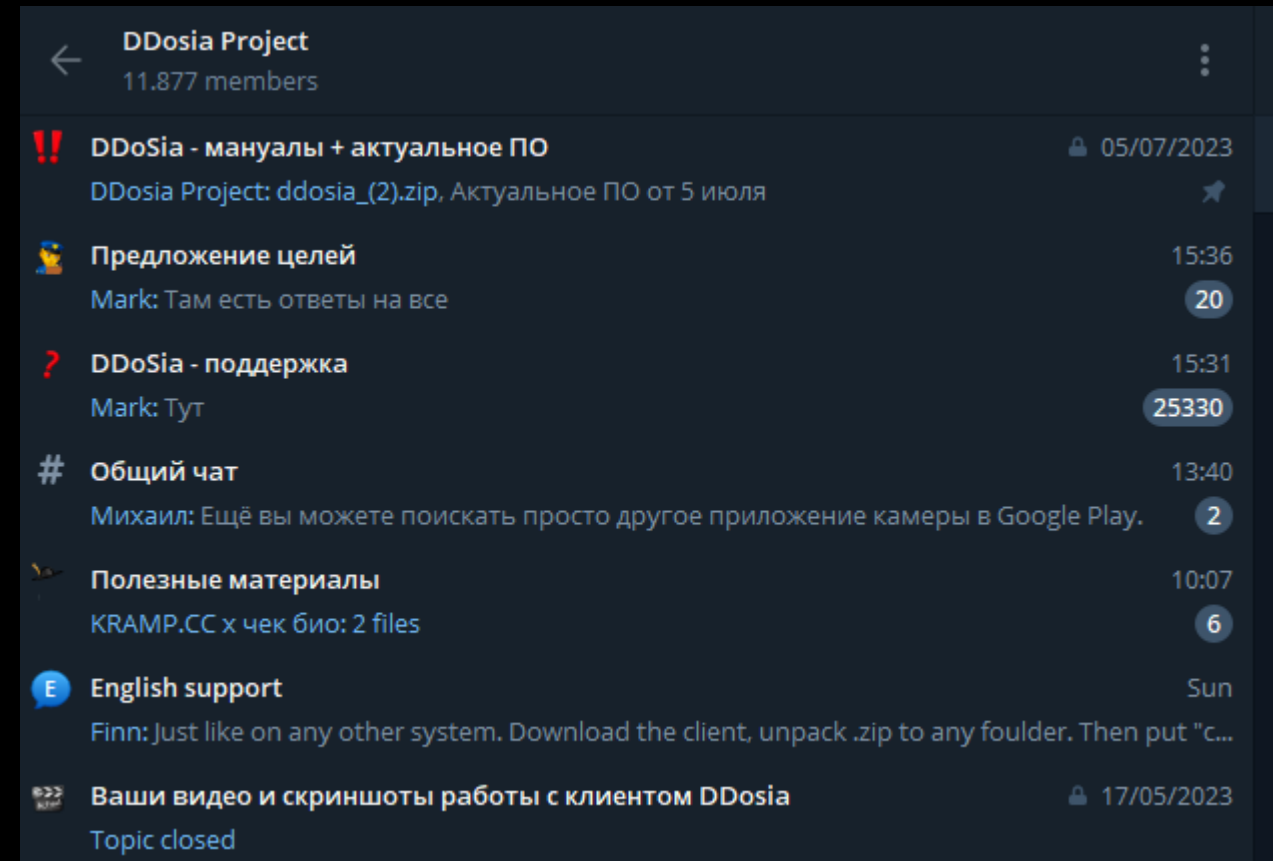
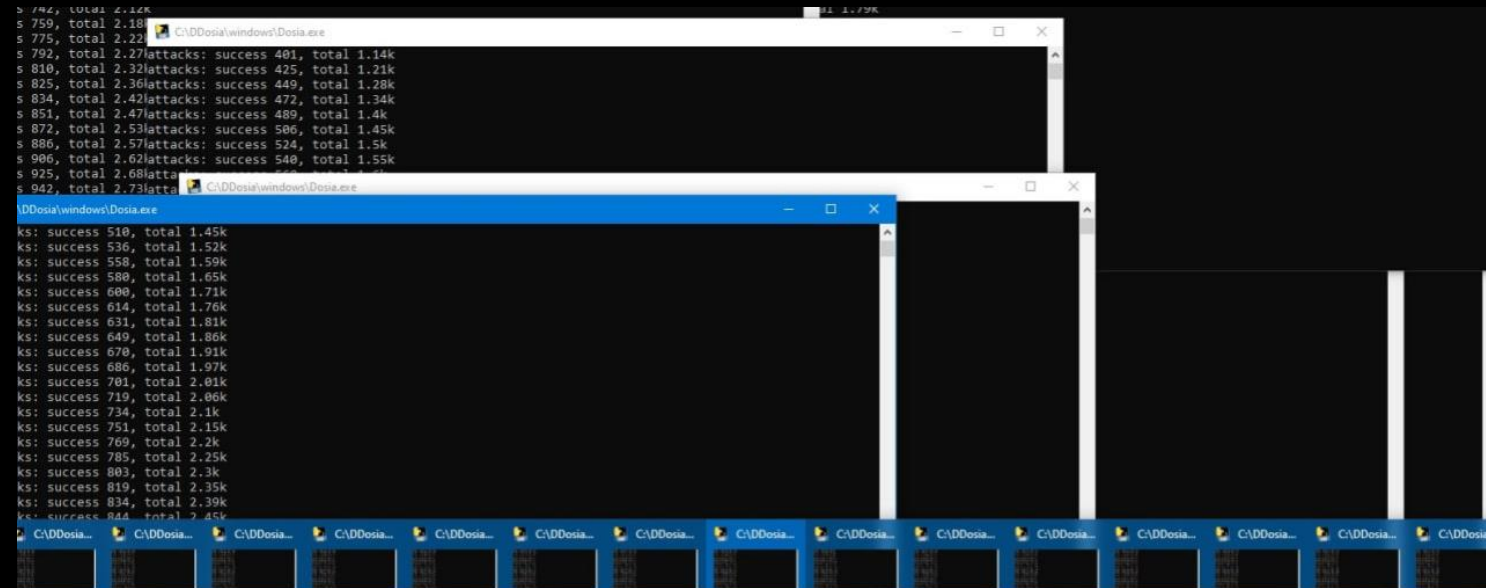


Image Source: Telegram @DDoSia

# DDOSIA PROJECT – ATTACK TYPES

- **HTTP GET/POST** – Request randomization & Customization
- **Nginx\_loris** – Classic Slowloris on Nginx. Setting up multiple connections and sending parts of HTTP requests to keep the connectios open.
- **HTTP2** – Same as with HTTP, but on the HTTP2 protocol.
- **TCP** – TCP\_SYN Floods with random spoofed IPs and ports.

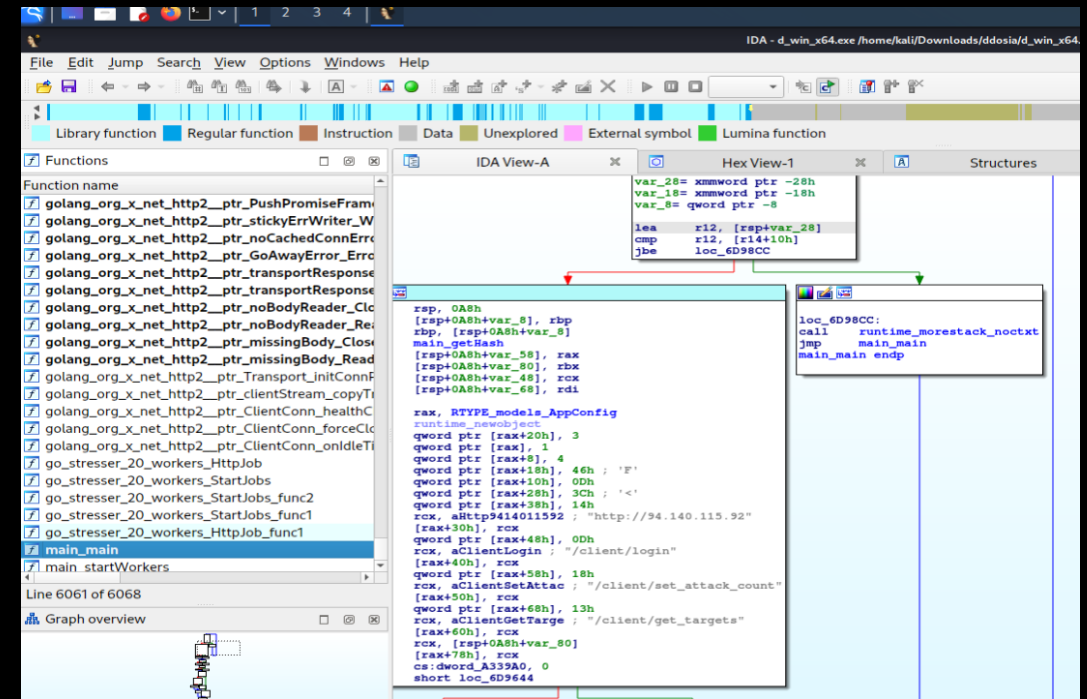
Image Source: Yarix



- Python: 1800 requests per minute with four cores and 20 threads
- Go should be 8 times more efficient, so 14.400 requests
- Multiplied by about 12,000 users (max) = 172.800.000 requests / min

# DDOSIA CLIENT - REVERSING

- Reversed in IDA
- Written in GoLang
- Windows, MacOS and Linux clients (x64 and arm64)
- Nothing really obfuscated or encrypted
- So we can find the current C2:  
94[.]140[.]115[.]92
- And URI's such as '/client/login' and '/client/get\_targets'



```
qword ptr [rax+20h], 3Ch ;
qword ptr [rax+38h], 14h
rcx, aHttp9414011592 ; "http://94.140.115.92"
[rax+30h], rcx
qword ptr [rax+48h], 0Dh
rcx, aClientLogin ; "/client/login"
[rax+40h], rcx
qword ptr [rax+58h], 18h
rcx, aClientSetAttac ; "/client/set_attack_count"
[rax+50h], rcx
qword ptr [rax+68h], 13h
rcx, aClientGetTarge ; "/client/get_targets"
[rax+60h], rcx
rcx, [rsp+0A8h+var_80]
```

# DDOSIA CLIENT - REVERSING

- Crypto/AESCipherGCM is being used to encrypt the targets after the “GET /client/get\_targets” with AES-GCM
- Sekoia luckily found out how to decrypt this already using dynamic analysis (Source: <https://blog.sekoia.io/following-noname05716-ddosia-projects-targets/>)

```
; crypto/aes.(*aesCipherGCM).Decrypt
; void __golang_crypto_aes_ptr_aesCipherGCM_Decrypt(_ptr_aes_aesCipherGCM, _slice_uint8, _slice_uint8)
crypto_aes_ptr_aesCipherGCM_Decrypt proc near
var_8= qword ptr -8
arg_0= qword ptr 8
arg_8= _slice_uint8 ptr 10h
arg_20= _slice_uint8 ptr 28h
cmp     rsp, [r14+10h]
jbe     short loc_4DE778

sub     rsp, 40h
mov     [rsp+40h+var_8], rbp
lea     rbp, [rsp+40h+var_8]
mov     r12, [r14+20h]
test    r12, r12
jnz     short loc_4DE7CD

loc_4DE778:
mov     [rsp+arg_0], rax
mov     [rsp+arg_8.ptr], rbx
mov     [rsp+arg_8.len], rcx
mov     [rsp+arg_8.cap], rdi
mov     [rsp+arg_20.ptr], rsi
mov     [rsp+arg_20.len], r8
mov     [rsp+arg_20.cap], r9
nop
dword ptr [rax+rax+00h]
call    runtime_moresstack_noctxt
mov     rax, [rsp+arg_0] ; _ptr_aes_aesCipherGCM
mov     rbx, [rsp+arg_8.ptr] ; _slice_uint8
mov     rcx, [rsp+arg_8.len]
mov     rdi, [rsp+arg_8.cap]
mov     rsi, [rsp+arg_20.ptr] ; _slice_uint8
mov     r8, [rsp+arg_20.len]
mov     r9, [rsp+arg_20.cap]
jmp     crypto_aes_ptr_aesCipherGCM_Decrypt
```

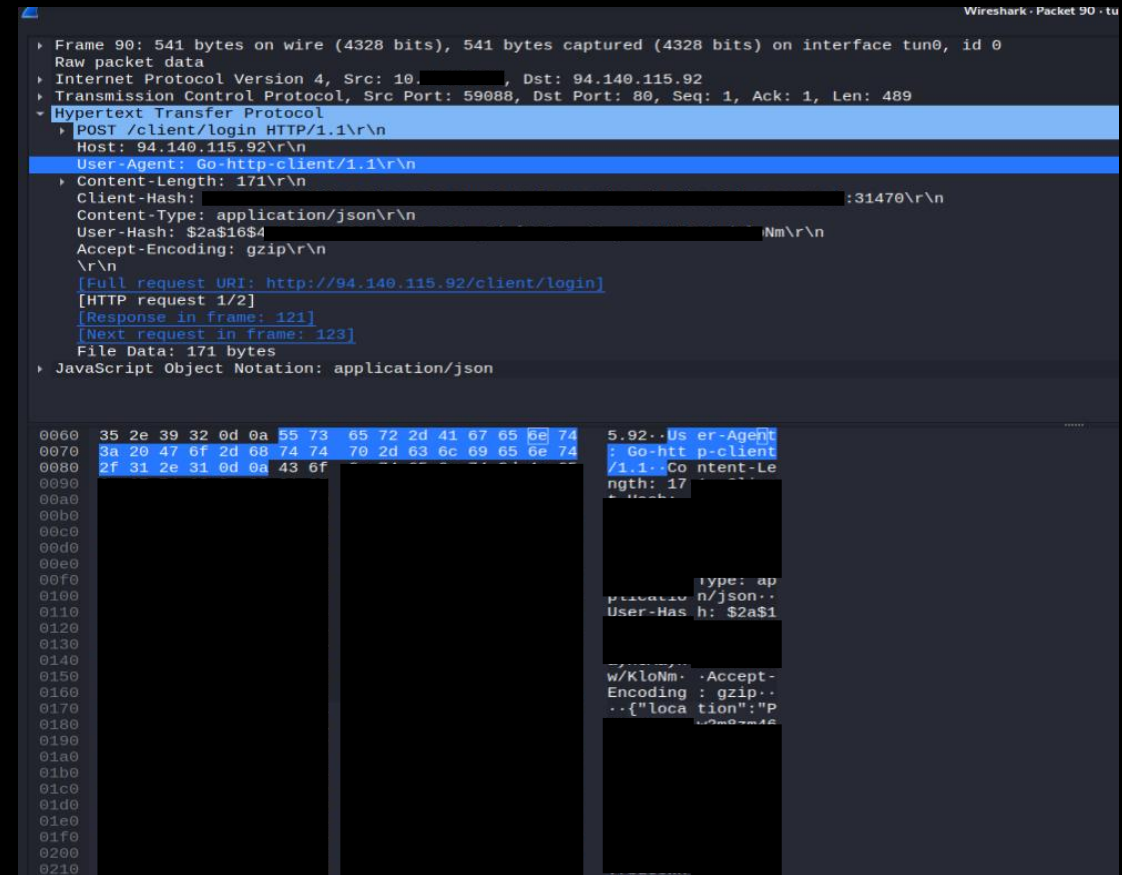
```
// Append the User-Hash and the Token
size_NewString = fmt.Sprintf(2LL, 2LL, v10, &v28);
runtime_stringtoslicebyte(2LL, 2LL, v12, 4LL);
if ( size_NewString <= 32 )
    sizeToRead = 0LL;
else
    sizeToRead = size_NewString - 32;
if ( sizeToRead > size_NewString )
    sub_465C20(2LL, 2LL, sizeToRead, 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);
}
```

Image Source: Sekoia.io



# DDOSIA CLIENT - WIRESHARK

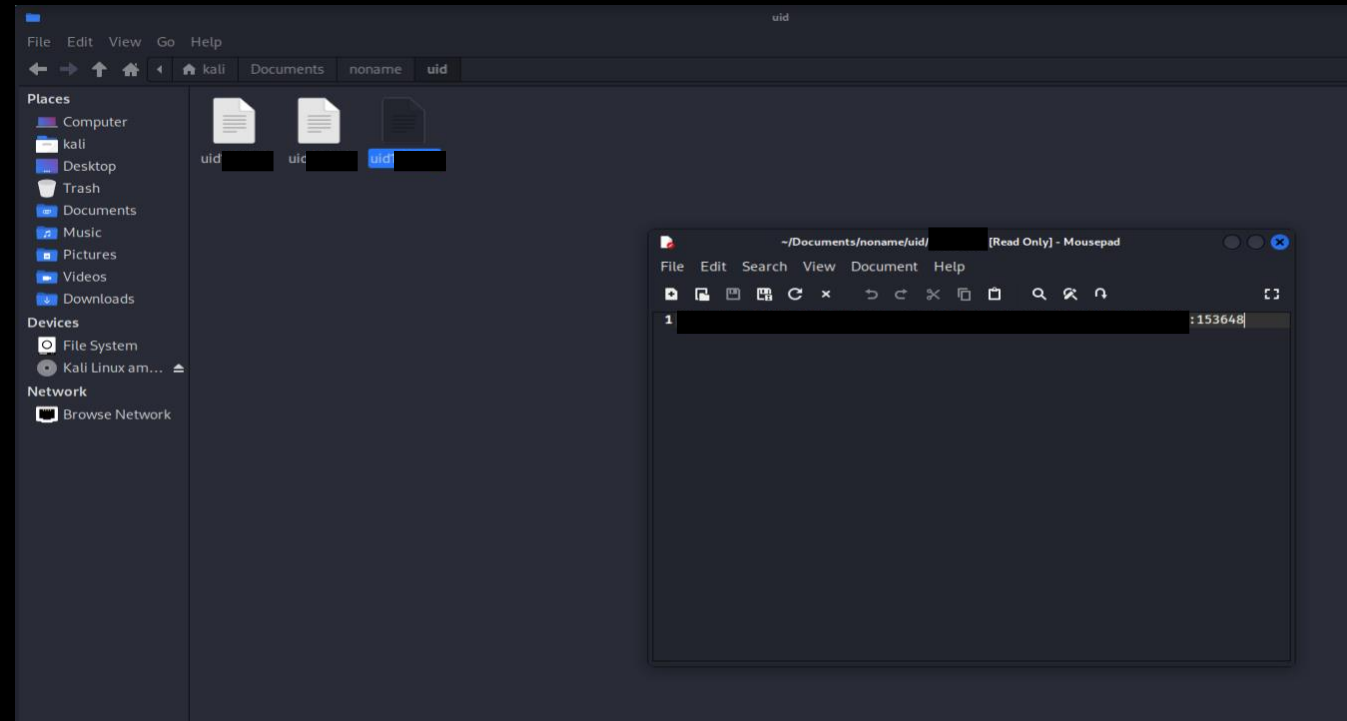
- Client does a POST to '/client/login' on the C2 server, with User-Agent: "Go-http-client/1.1"
- The Client-Hash is a generated hash of the OS UUID/GUID. They use the "Go MachineID" from GitHub (<https://github.com/denisbrodbeck/machineid>)



```
Wireshark - Packet 90 - tu
  > Frame 90: 541 bytes on wire (4328 bits), 541 bytes captured (4328 bits) on interface tun0, id 0
  Raw packet data
  > Internet Protocol Version 4, Src: 10.10.10.10, Dst: 94.140.115.92
  > Transmission Control Protocol, Src Port: 59088, Dst Port: 80, Seq: 1, Ack: 1, Len: 489
  > Hypertext Transfer Protocol
    > POST /client/login HTTP/1.1\r\n
      Host: 94.140.115.92\r\n
      User-Agent: Go-http-client/1.1\r\n
      Content-Length: 171\r\n
      Client-Hash: [REDACTED]:31470\r\n
      Content-Type: application/json\r\n
      User-Hash: $2a$16$4[REDACTED]Nm\r\n
      Accept-Encoding: gzip\r\n
      \r\n
      [Full request URI: http://94.140.115.92/client/login]
      [HTTP request 1/2]
      [Response in frame: 121]
      [Next request in frame: 123]
      File Data: 171 bytes
    > JavaScript Object Notation: application/json
      {
        "location": "P[REDACTED]"
      }
  0060  35 2e 39 32 0d 0a 55 73 65 72 2d 41 67 65 0e 74  5.92..Us er-Agent
  0070  3a 20 47 6f 2d 68 74 74 70 2d 63 6c 09 65 0e 74  : Go-htt p-client
  0080  2f 31 2e 31 0d 0a 43 6f                               /1.1. Content-Le
  0090  [REDACTED]
  00a0  [REDACTED]
  00b0  [REDACTED]
  00c0  [REDACTED]
  00d0  [REDACTED]
  00e0  [REDACTED]
  00f0  [REDACTED]
  0100  [REDACTED]
  0110  [REDACTED]
  0120  [REDACTED]
  0130  [REDACTED]
  0140  [REDACTED]
  0150  [REDACTED]
  0160  [REDACTED]
  0170  [REDACTED]
  0180  [REDACTED]
  0190  [REDACTED]
  01a0  [REDACTED]
  01b0  [REDACTED]
  01c0  [REDACTED]
  01d0  [REDACTED]
  01e0  [REDACTED]
  01f0  [REDACTED]
  0200  [REDACTED]
  0210  [REDACTED]
```

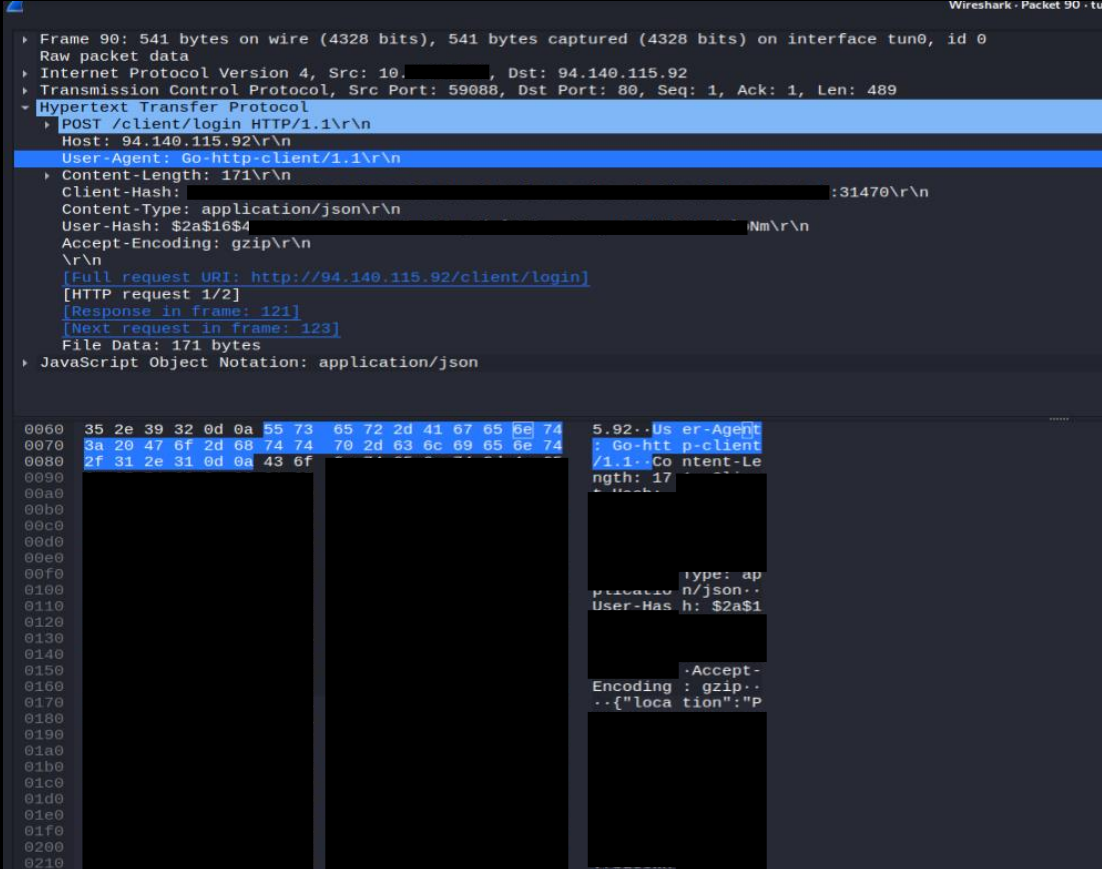
# DDOSIA CLIENT - WIRESHARK

- On the right you see the contents of such a uid file (Client-Hash).
- Right behind the machine ID, the PID of the client is appended, in this case 153648.



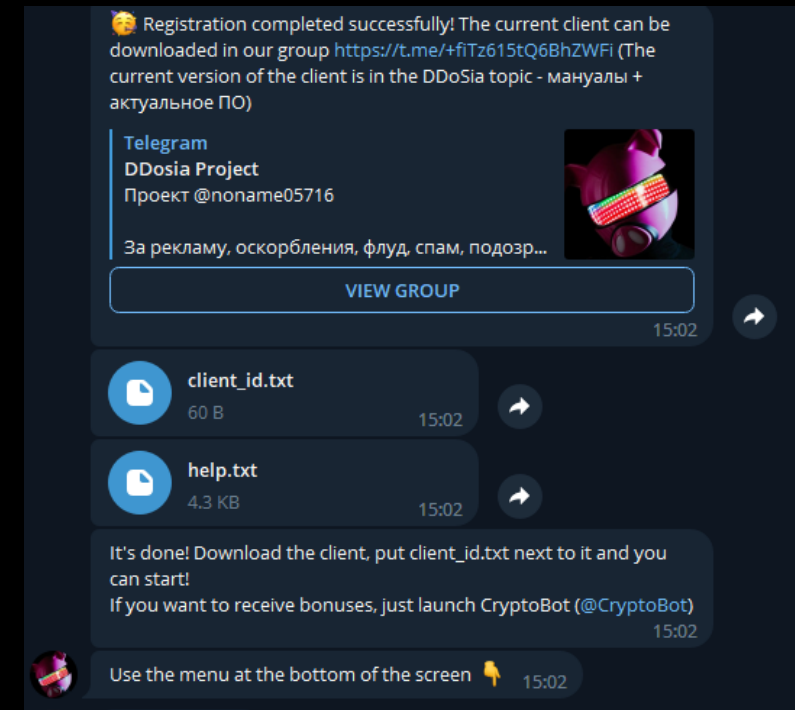
# DDOSIA CLIENT - WIRESHARK

- Also there: Content-Type: 'application/json' and the User-Hash.
- The User-Hash is also unique. This is taken from the 'client\_id' file, which has to be in the same folder as the DDoSia Client.
- How to get this unique Client\_ID? Through the official DDoSia Telegram bot (@ddosia\_bot).
- How you get to this bot? Only through the semi-private DDoSia Telegram group.



```
Wireshark - Packet 90 - tu
  Frame 90: 541 bytes on wire (4328 bits), 541 bytes captured (4328 bits) on interface tun0, id 0
  Raw packet data
  Internet Protocol Version 4, Src: 10.0.0.1, Dst: 94.140.115.92
  Transmission Control Protocol, Src Port: 59088, Dst Port: 80, Seq: 1, Ack: 1, Len: 489
  Hypertext Transfer Protocol
    POST /client/login HTTP/1.1\r\n
    Host: 94.140.115.92\r\n
    User-Agent: Go-http-client/1.1\r\n
    Content-Length: 171\r\n
    Client-Hash: [REDACTED]:31470\r\n
    Content-Type: application/json\r\n
    User-Hash: $2a$16$4[REDACTED]Nm\r\n
    Accept-Encoding: gzip\r\n
    \r\n
    [Full request URI: http://94.140.115.92/client/login]
    [HTTP request 1/2]
    [Response in frame: 121]
    [Next request in frame: 123]
    File Data: 171 bytes
  JavaScript Object Notation: application/json
  0060  35 2e 39 32 0d 0a 55 73 65 72 2d 41 67 65 0e 74  5.92..Us er-Agent
  0070  3a 20 47 6f 2d 68 74 74 70 2d 63 6c 09 65 0e 74  : Go-htt p-client
  0080  2f 31 2e 31 0d 0a 43 6f  \r\n  /1.1. Co ntent-Le
  0090  \r\n  ngth: 17
  00a0  \r\n
  00b0  \r\n
  00c0  \r\n
  00d0  \r\n
  00e0  \r\n
  00f0  \r\n
  0100  \r\n
  0110  \r\n
  0120  \r\n
  0130  \r\n
  0140  \r\n
  0150  \r\n
  0160  \r\n
  0170  \r\n
  0180  \r\n
  0190  \r\n
  01a0  \r\n
  01b0  \r\n
  01c0  \r\n
  01d0  \r\n
  01e0  \r\n
  01f0  \r\n
  0200  \r\n
  0210  \r\n
```

# DDOSIA BOT



# DDOSIA CLIENT – WIRESHARK CONTD.

- So, there has been done a POST to '/client/login' with our unique values. You then get a 'HTTP 200 OK' in return, along with the data and time in Epoch format.
- Finally it does a 'GET /client/get\_targets', where the Epoch time value is added to the request. This is necessary, else you will receive a 'HTTP 401 Unauthorized'.

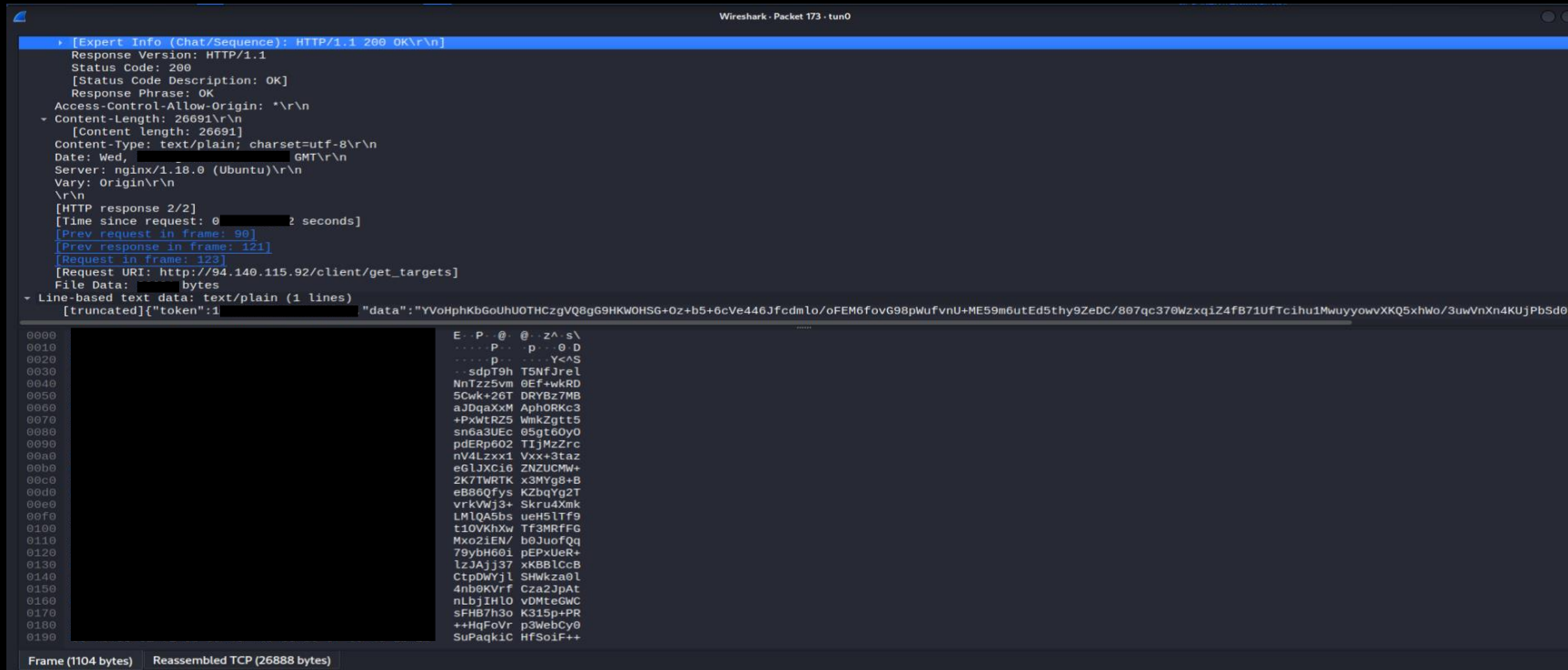
```
[Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n
Response Version: HTTP/1.1
Status Code: 200
[Status Code Description: OK]
Response Phrase: OK
Access-Control-Allow-Origin: *\r\n
Content-length: 28\r\n
Content-type: text/plain; charset=utf-8\r\n
Date: Wed, 07 Aug 2023 13:07 GMT\r\n
Server: nginx/1.18.0 (Ubuntu)\r\n
Vary: Origin\r\n
\r\n
[HTTP response 1/2]
[Time since request: 4.123 seconds]
[Request in frame: 90]
[Next request in frame: 123]
[Next response in frame: 173]
[Request URI: https://94.140.115.92/client/get_targets]
File Data: 19 bytes
- Line-based text data: text/plain (1 lines)
16
0000
0010
0020
0030
0040
0050
0060
0070
0080
0090
00a0
00b0
00c0
00d0
00e0
00f0
0100
0110
0120
0130
0140
0150
0160
0170
E...@...A s\
...P...pid 0
...N...Ye[
...] HTTP /1.1 200
...OK Acc-ess-cont
rol-Allo w-Origin
...: Con tent-Len
gth: 19 Content
-Type: t ext/plai
n; char s
...
Aug 2023 13 07
GMT: S ervers: n
ginx/1.1 8.0 (ubu
ntu) Va rv: Ori g
in: 16
8
```

```
Wireshark - Packet 123 - tun0
Frame 123: 381 bytes on wire (3048 bits), 381 bytes captured (3048 bits) on interface tun0, id 0
Raw packet data
Internet Protocol Version 4, Src: 10.0.0.1, Dst: 94.140.115.92
Transmission Control Protocol, Src Port: 59088, Dst Port: 80, Seq: 490, Ack: 214, Len: 329
Hypertext Transfer Protocol
  GET /client/get_targets HTTP/1.1\r\n
  [Expert Info (Chat/Sequence): GET /client/get_targets HTTP/1.1\r\n]
  Request Method: GET
  Request URI: /client/get_targets
  Request Version: HTTP/1.1
  Host: 94.140.115.92\r\n
  User-Agent: Go-http-client/1.1\r\n
  Client-Hash: 58a...:31470\r\n
  Content-Type: application/json\r\n
  Time: 1692...
  User-Hash: $2a$16s...
  Accept-Encoding: gzip\r\n
  \r\n
  [Full request URI: http://94.140.115.92/client/get_targets]
  [HTTP request 2/2]
  [Prev request in frame: 90]
  [Response in frame: 173]
0000
0010
0020
0030
0040
0050
0060
0070
0080
0090
00a0
00b0
00c0
00d0
00e0
00f0
0100
0110
0120
0130
0140
0150
0160
0170
E...}...@...[...
A s\...P...0...pJ
...N......pU
Y<[ GET /client/
get_targ ets HTTP
/1.1 Ho st: 94.1
40.115.9 2 User-
Agent: G o-http-c
lient/1. 1 Clie n
...:31470
Content- Type: ap
plicatio n/ison
Time: 16
69842421 3 User-
Hash: $2 a$16$4qL
Nm Acce pt-Encod
ing: gzi p...
```



# DDOSIA CLIENT – WIRESHARK CONTD.

- Finally you get the Targets back from the Server, along with a unique Token
- So, the Target data is encrypted by AES-GCM. We can decrypt this thanks to the info on Sekoia's blog

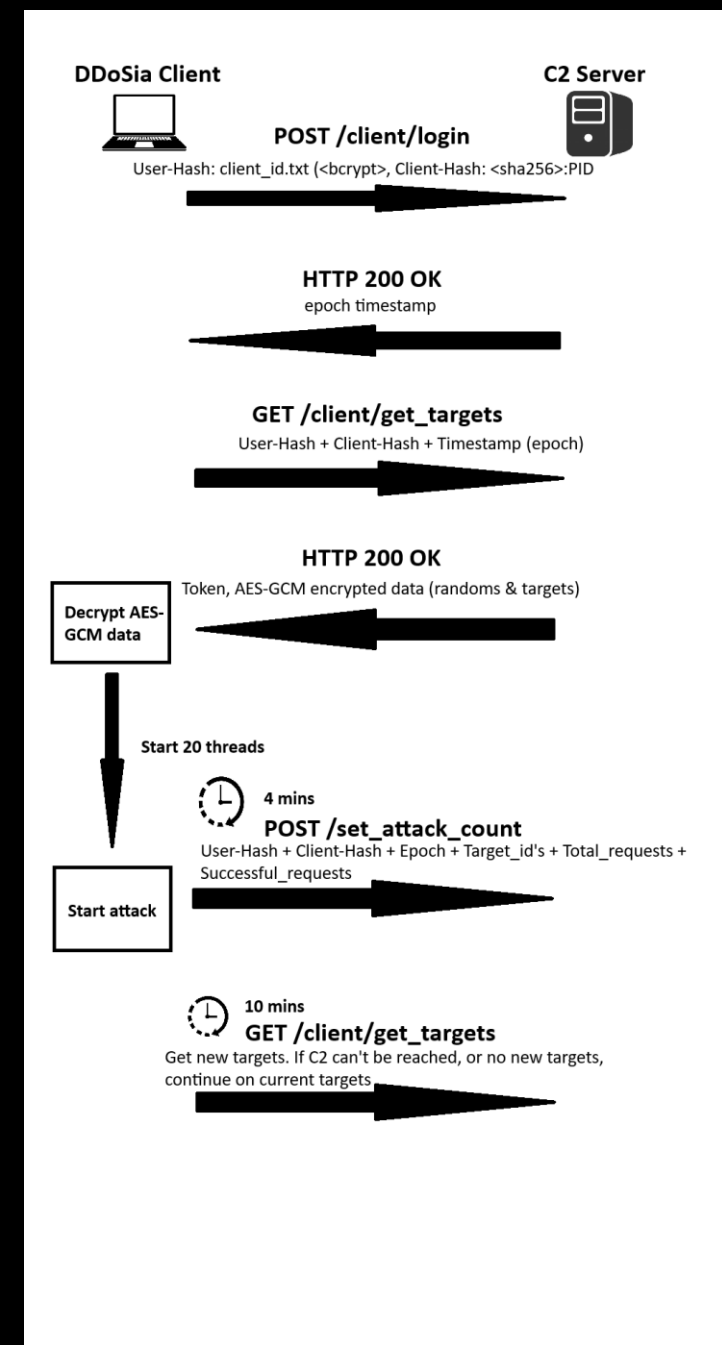


```
Wireshark - Packet 173 - tun0
[Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
Response Version: HTTP/1.1
Status Code: 200
[Status Code Description: OK]
Response Phrase: OK
Access-Control-Allow-Origin: *\r\n
Content-Length: 2688\r\n
[Content length: 2688]
Content-Type: text/plain; charset=utf-8\r\n
Date: Wed, 14 Jun 2017 10:00:00 GMT\r\n
Server: nginx/1.18.0 (Ubuntu)\r\n
Vary: Origin\r\n
\r\n
[HTTP response 2/2]
Time since request: 0.000 seconds
[Prev request in frame: 90]
[Prev response in frame: 121]
[Request in frame: 123]
[Request URI: http://94.140.115.92/client/get_targets]
File Data: 2688 bytes
- Line-based text data: text/plain (1 lines)
[truncated] {"token": "1", "data": "YVoHphKbGoUHU0THCzgvQ8g69HKW0HSG+Oz+b5+6cVe446Jfcdmlo/oFEM6fovG98pWufvntu+ME59m6utEd5thy9ZeDC/807qc370Wzqxq1Z4FB71UfTc1ihu1MwuyyowvXKQ5xhW0/3uwVnXn4KUJpBsd0r"}
0000  E..P.:@.:Z^s\
0010  ....P...p...0D
0020  ....p...Y<AS
0030  ..sdpT9h T5NfJrel
0040  NnTzz5vm 0Ef+wkRD
0050  5Cwk+26T DRyBz7MB
0060  a3DqaXxM AphORkC3
0070  +PxWtRZ5 WmkZgtt5
0080  sn6a3UEc 05gt60y0
0090  pdERp602 TijMzZrc
00a0  nV4Lzxx1 Vxx+3Taz
00b0  e6l3xC10 ZNZUCMw+
00c0  2K7TMRtK x3MYg8+B
00d0  eB86Qfys KZbqYq2T
00e0  vrkVmj3+ Skru4Xmk
00f0  LM1QA5bs ueH5lTf9
0100  t10Vkhxw TF3MRFFG
0110  Mxo2iEN/ b0Juf0q
0120  79ybh60i pEPxUeR+
0130  lzAj37 xKBBlccB
0140  CtpDMyj1 SHwkza0l
0150  4nb0KVrf Cza2JpAt
0160  nLbjIHl0 vDMte6wC
0170  sFH67h30 K319p+PR
0180  ++HqFovr p3Webcy0
0190  SuPaqkiC HfSo1E++
Frame (1104 bytes) Reassembled TCP (2688 bytes)
```

## DDOSIA CLIENT – WIRESHARK CONTD.

- If you don't quit the client at this time, it will start the DDoS-attack. After the previous step, the client starts up 20 threads simultaneously.
- Then it starts attacking all the targets all together (decrypted from AES-GCM).
- Every 4 minutes the Client does a 'POST /set\_attack\_count', with the total number of connections and how many were successful.
- New targets are then requested and obtained from the server every 10 minutes. If no connection can be made with the C2 anymore, the client keeps attacking the last known targets.

# DDOSIA CLIENT – OVERVIEW



# DDOSIA CLIENT – DECRYPTING

Thanks to dynamic analysis of the client by Sekoia

## Key Calculation:

- Token value / 5
- Result is added to the User-Hash
- Takes last 32 chars of User-Hash and convert to Hex

## IV Calculation:

- Take the ciphertext (targets), decode from Base64 to ASCII
- Take the first 12 chars and convert to bytes

## TAG Calculation:

- Take the ciphertext (targets), decode from Base64 to ASCII
- Take the last 16 chars and convert to bytes

Now the targets can be decrypted...

```
// 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);
}
```

Image Source: Sekoia.io

# DDOSIA CLIENT – DECRYPTING

- After decryption we get a one-liner with all the target info (see below).

```
1 {"randoms":[{"name":"Телефон","id":"62d8286fddcbb37b0c77c87f","digit":true,"upper":false,"lower":false,"min":11,"max":11},{name":"Все символы 6-12","id":"62d8fccfb44b5774ee96ec0a","digit":true,"upper":true,"lower":true,"min":6,"max":12}]}
```

- Then we can use a JSON beautifier tool to get a better overview, i.e.:

<https://codebeautify.org/jsonviewer>



# DDOSIA CLIENT – DECRYPTING

- After 'beautifying' we get a great overview.
- Targets consist of 2 parts: 'randoms' and 'targets'.
- The 'randoms' part is used for randomizing the requests. This are defined values, such as numbers, min/max integer values and upper- or lowercase letters.

```
{  
  "randoms": [  
    {  
      "name": "Телефон",  
      "id": "62d8286fddcbb37b0c77c87f",  
      "digit": true,  
      "upper": false,  
      "lower": false,  
      "min": 11,  
      "max": 11  
    },  
  ],  
}
```

# DDOSIA CLIENT – DECRYPTING

```
"targets": [  
  {  
    "target_id": "644a281974c402c5b2cc1192",  
    "request_id": "644a281a74c402c5b2cc1193",  
    "host": "pz.gov.pl",  
    "ip": "185.41.93.77",  
    "type": "http",  
    "method": "POST",  
    "port": 443,  
    "use_ssl": true,  
    "path": "/dt/login/login",  
    "body": {  
      "type": "str",  
      "value": "loginForm=loginForm&loginForm%3Alogin=$_1%40gmail.com&loginForm%3Ahas%C5%82o=$_1&loginForm%3AloginButton=Zaloguj+si%C4%99&javax.faces.ViewState=7833864021476204279%3A5328956561244476557"  
    },  
    "headers": null  
  },  
],
```

# DDOSIA CLIENT – DECRYPTING

The 'targets' part indicates the current targets to be DDoS'ed by the client. These have atleast:

1. A unique ID (for tracking),
2. A type and method (i.e. HTTP POST)
3. A target host, IP and port
4. A path/URI (i.e. /register or /login)
5. The body, with the value type (i.e. String) and value (i.e. "user\_email=\$\_1", where '\$\_1' will be replaced by a random value from the 'randoms' part.
6. Timeout in ms (default 1000).
7. If it has to wait for a response (True / False).
8. Eventual headers to send along with the requests.

# DDOSIA CLIENT – STATS & FINANCIAL MOTIVATION

NoName057(16)

Friends! We are with a good update. As promised, we are introducing financial incentives for the leaders of our TOP of the most powerful DDoS fighters .

Prize fund after successful attacks:

For 1-3 places :

- 80 000 rubles for 1st place
- 50 000 rubles for 2nd place
- 20 000 rubles for 3rd place

Payment in cryptocurrency at the rate on the day of payment.

From 4th to 10th places - 50,000 rubles. divided proportionally according to the number of successful attacks.

Image Source: Telegram @NoName

The screenshot shows a Telegram chat interface with a dark theme. At the top, there's a header 'Stats' and a message 'Doing something...'. Below that, a message says 'Use the menu at the bottom of the screen'. A 'Daily top' section follows, with another 'Doing something...' message. The main content consists of two lists of 'TOP 10 DDoS'ers'. The first list is 'by total number of requests since' and the second is 'by the number of successful requests since'. Each list includes a rank, a name, and the number of attacks made. At the bottom, there's a 'My stats' section with the text 'You haven't launched the client yet on 24.08.2023' and another 'Use the menu at the bottom of the screen' message. The bottom of the chat shows a 'Write a message...' input field and a navigation bar with buttons for 'Daily top', 'My stats', 'Back', and 'Help'.

Stats

Doing something...

Use the menu at the bottom of the screen

Daily top

Doing something...

TOP 10 DDoS'ers by total number of requests since

- 1) Z.V.E.Z.D.A - 48 101 355 attacks made
- 2) Grizzly - 19 375 170 attacks made
- 3) Error 503 - 18 430 328 attacks made
- 4) Wonderboy - 17 769 882 attacks made
- 5) Russian\_Paradise - 17 485 521 attacks made
- 6) king\_of\_aces - 17 117 171 attacks made
- 7) Vladimir Putin - 16 981 840 attacks made
- 8) Cookiepie - 16 220 683 attacks made
- 9) Pizda\_hohlam - 15 601 051 attacks made
- 10) Bratishkam - 15 581 963 attacks made

TOP 10 DDoS'ers by the number of successful requests since

- 1) Z.V.E.Z.D.A - 4 955 167 attacks made
- 2) Vladimir Putin - 2 986 401 attacks made
- 3) Wonderboy - 2 746 602 attacks made
- 4) Grizzly - 2 743 891 attacks made
- 5) Russian\_Paradise - 2 595 424 attacks made
- 6) EXCOMMUNICADO - 2 337 616 attacks made
- 7) king\_of\_aces - 2 270 238 attacks made
- 8) Bratishkam - 2 266 701 attacks made
- 9) Pizda\_hohlam - 2 187 732 attacks made
- 10) Kalashnikov47 - 2 126 457 attacks made

My stats:  
You haven't launched the client yet on 24.08.2023

Use the menu at the bottom of the screen

Write a message...

Daily top My stats

Back Help



The time has come, following the ranks and achievements (we wrote about them [here](#)) to talk about another important change in our volunteer DDoSia Project.

⚡⚡⚡ We are introducing the world's first electronic currency tied to real actions - dCoin ✨ ✨ ✨

Coins will be awarded to the soldiers of our volunteer army according to their contribution to the attacks (combat merit) and according to their rank (military rank). That is, the higher the volunteer's rank and the more attacks he made, the more dCoin he will be credited.

dCoin can be withdrawn - converted into TONcoin and sent to your crypto wallet.

✅ The dCoin rate at the first stage will be equal to: 1dCoin=1 Russian ruble. But it will change depending on the activity of volunteers (the more efficiently we work, the steeper the rate will be and vice versa).

Image Source: Telegram @DDoSia



# CONTENTS

- Hacktivism, Targets & Methods
- DDoS Attack Types & Botnets
- Hacktivist Groups, Attacks & Cyberwarfare
- Anonymous Sudan, Killnet & Dark Parliament
- NoName057(16)
- Technical Analysis of DDoSia Client
- **Red Cross Rules of Engagement for Hackers**
- What can you do?
- Questions



# RED CROSS ROE FOR CIVILIAN HACKERS

- Published on October 4th 2023
- A “Genova Code of Cyber-War”
- For civilian hackers involved in conflicts

## Some of the rules:

- Do not attack civilian targets
- Comply with these rules even if the enemy doesn’t
- Do not conduct any cyber-attack against medical and humanitarian facilities
- Do not threaten violence to spread terror among civilians

## Replies from Hactivist Alliances:

- IT Army of Ukraine: “Make best efforts to follow rules”
- Killnet: initially refused, few days later agreed to abide
- Anonymous: always operates on principles, but will not follow the rules
- Anonymous Sudan: not viable and breaking them for the group’s cause is unavoidable

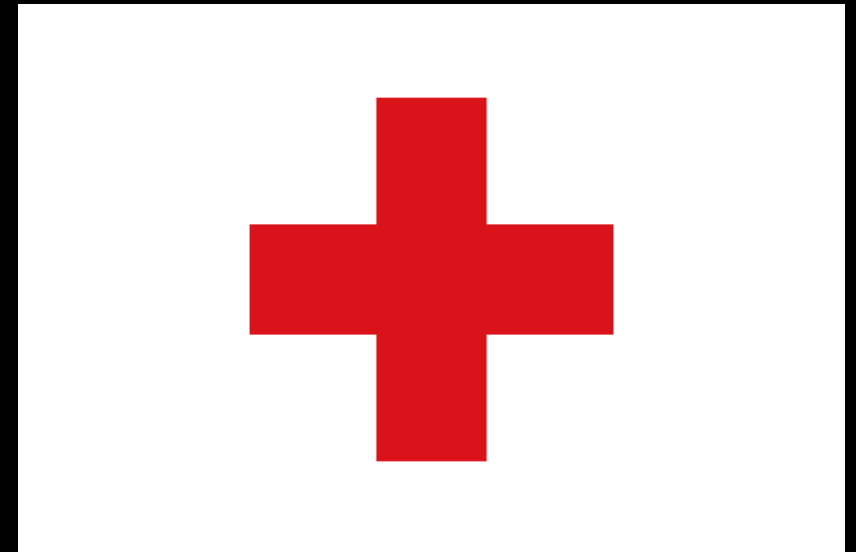


Image Source: Red Cross

# CONTENTS

- Hacktivism, Targets & Methods
- DDoS Attack Types & Botnets
- Hacktivist Groups, Attacks & Cyberwarfare
- Anonymous Sudan, Killnet & Dark Parliament
- NoName057(16)
- Technical Analysis of DDoSia Client
- Red Cross Rules of Engagement for Hackers
- **What can you do?**
- Questions



# WHAT CAN YOU DO?



- **Updating:** Keep everything up-to-date!
- **Shrink Attack Surface:** Disable unused services and ports
- **Network Segmentation:** So that a DDoS Attack doesn't take down all the components of your network
- **Mitigation:** Use a Content Distribution Network (CDN) or 'Washing Service' such as NaWas
- **Logging:** Log all attacks, enable notifications to detect anomalies.
- **OSINT & CTI:** Monitor these groups and potential upcoming attacks
- **Fingerprinting:** Make use of ADC's Clearing House and the fingerprint database to detect well-known attacks
- **Testing:** Participate in ADC's DDoS Test each year to test your resilience

... or else, just wait until your servers will go back up ...

**THE END**