


# DDosia Project: How NoName057(16) is trying to improve the efficiency of DDoS attacks

 [decoded.avast.io/martinchlumecky/ddosia-project-how-noname05716-is-trying-to-improve-the-efficiency-of-ddos-attacks/](https://decoded.avast.io/martinchlumecky/ddosia-project-how-noname05716-is-trying-to-improve-the-efficiency-of-ddos-attacks/)

April 18, 2023



by [Martin Chlumecký](#) April 18, 2023 14 min read

Through their DDosia project, pro-Russia hacktivist group NoName057(16) is still conducting DDoS attacks, mostly with the goal to take offline websites of institutions and companies in European countries. On its Telegram channels, the group openly communicates the fact that they perform their actions in support of Russia in the war against Ukraine, and it's apparent that their activities will further continue during the war. The group has been offering payments in cryptocurrencies to people who install their DDosia tool in order to participate in their attacks. We want to create awareness that people who have NoName057(16)'s DDoS tool installed on their computer not only participate in cybercrime, but also support the groups' warfare activities.

We detect and block DDosia to make the internet a safer place, and we continue to track DDoS victims and configurations of the DDosia botnet because such information helps to mitigate the impact of DDoS attacks.

Since the first Python version needed to be more efficient, the group released a new Go variant of bots in late 2022. [SentinelLabs](#) has described the first variant of the Go implementation, including the C2 servers at that time active. A few days later, [Team Cymru](#) published an investigation about the botnet architecture describing the DDoS attacks as a largely static infrastructure.

Given the above findings, it is apparent that the C2 structure is still evolving. The primary purpose of the following analysis is to explore the C2 architecture and current communication process between the botnet and C2 servers. Therefore, we have been

actively monitoring the DDosia botnet and have found several innovations in the bot implementation and the botnet infrastructure. The C2 infrastructure is composed of one central server and two proxies forwarding bot requests. This, combined with an update mechanism, makes the botnet rather resilient to disruptions. The latest versions also include a bot authentication mechanism for all the C2 communication along with IP address blocklisting, presumably to hinder tracking of the project.

## Implementation Overview

---

The first implementation of DDosia came into the world around July 2022. Being authored by the NoName057(16) group, there was interestingly a brief coexistence with the Bobik botnet before the botnet was dismantled, presumably in favor of DDosia. It was written in Python using threads as a means of parallelism; nevertheless, it was still lacking in terms of efficacy. Since the first version, DDosia relied on HTTP protocol for C2 communication, with JSON configs distributed by the servers.

The lack of efficacy presumably motivated changes in DDosia, namely the move from Python to Go that we saw in late 2022, with [SentinelLabs](#) describing the first Go variants. The main advantage of Go in comparison to Python is direct compilation into native code along with the absence of Python's GIL that may severely affect the performance of threaded code in Python. Interestingly, these new versions are also multi-platform, as we've seen variants for all major operating systems (Windows, macOS, Linux, Android). Evidently, the bot development is still in progress, as we see new functionalities, such as HTTP authentication, being added to DDosia along with slight changes in the configuration file.



```
Authorization completed successfully
Targets received: 58
Successful responses (http code 200): 602 | Total responses received: 943 | Total requests sent: 1057
Successful responses (http code 200): 1014 | Total responses received: 1434 | Total requests sent: 2345
Successful responses (http code 200): 1331 | Total responses received: 1874 | Total requests sent: 4229
Successful responses (http code 200): 1807 | Total responses received: 2483 | Total requests sent: 6354
Successful responses (http code 200): 6749 | Total responses received: 7651 | Total requests sent: 14034
Successful responses (http code 200): 8450 | Total responses received: 9321 | Total requests sent: 15703
Successful responses (http code 200): 13068 | Total responses received: 14022 | Total requests sent: 20420
Successful responses (http code 200): 13146 | Total responses received: 14015 | Total requests sent: 25448
2023/04/10 17:32:42 Goals update failed. It's okay, let's continue with the current
Successful responses (http code 200): 13132 | Total responses received: 14002 | Total requests sent: 26953
Successful responses (http code 200): 14257 | Total responses received: 15631 | Total requests sent: 31650
Successful responses (http code 200): 15909 | Total responses received: 17510 | Total requests sent: 36333
Successful responses (http code 200): 16170 | Total responses received: 17789 | Total requests sent: 37715
Successful responses (http code 200): 16575 | Total responses received: 18209 | Total requests sent: 40948
Successful responses (http code 200): 16915 | Total responses received: 18612 | Total requests sent: 44262
Successful responses (http code 200): 17348 | Total responses received: 19033 | Total requests sent: 47737
2023/04/10 17:43:15 Goals update failed. It's okay, let's continue with the current
Successful responses (http code 200): 17516 | Total responses received: 19188 | Total requests sent: 49421
Successful responses (http code 200): 17734 | Total responses received: 19429 | Total requests sent: 51213
Successful responses (http code 200): 18183 | Total responses received: 19892 | Total requests sent: 54623
Successful responses (http code 200): 18698 | Total responses received: 20347 | Total requests sent: 58091
Successful responses (http code 200): 21624 | Total responses received: 23318 | Total requests sent: 64309
Successful responses (http code 200): 23826 | Total responses received: 25527 | Total requests sent: 66503
Successful responses (http code 200): 25277 | Total responses received: 26972 | Total requests sent: 68433
2023/03/24 19:13:17 Goals update failed. It's okay, let's continue with the current
Successful responses (http code 200): 34351 | Total responses received: 36020 | Total requests sent: 77559
Successful responses (http code 200): 36810 | Total responses received: 38521 | Total requests sent: 79990
Successful responses (http code 200): 39005 | Total responses received: 40691 | Total requests sent: 82190
```

### Console output of the Go-Stresser version 1.0 – variant 2

#### Build Package

The aforementioned variant has support for multiple architecture as well as multiple platforms; unsurprisingly, it is also written in Go. The builds are distributed via the Telegram channel “Project DDosia” in the form of a zip file with the builds as follows:

- Windows x64 and arm64
- Linux x64 and arm64
- macOS x64 and arm64

The names of the executable are changed sometimes; there is a list of the captured names:

- dosia\_app\_(windows|macos|linux)\_(x64|arm64)
- d\_(win|mac|linux)\_(x64|arm64)
- pd\_(win|mac|linux)\_(x64|arm64)
- dosia\_(win|mac|linux)\_(x64|arm64)

#### Execution Workflow

A working dir of the bot executable must contain a text file `client_id.txt` with the **User-Hash** of the registered user. The form of the **User-Hash** is a BCrypt hash with these parameters `$2a$16$`.



```
GET /client/get_targets HTTP/1.1
Host: 94.140.114.239
User-Agent: Go-http-client/1.1
Client-Hash: 7d70614da685365d9d2fd609ddde0c6aeda77157a858e080a0a8747d4afd72e9:5481
Content-Type: application/json
Time: 16782xxxxxxxxxxxxxxxx
User-Hash: $2a$16$xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Accept-Encoding: gzip, deflate
Connection: close
```

### Getting targets from C2

The returned JSON file is similar to the first variant, with the difference that the original JSON configuration is wrapped up in a `data` key. Additionally, the new key `token` is included in each response of `GET /client/get_targets`. The `token` is a fresh token for further communication.

```
HTTP / 1.1 200 OK
Server: nginx / 1.18.0(Ubuntu)
Date: Wed, xx Mar 2023 xx:xx:xx GMT
Content-Type: text / plain; charset=utf-8
Content-Length: xxxxx
Connection: close
Vary: Origin
Access-Control-Allow-Origin:
Access-Control-Allow-Credentials: true
Access-Control-Expose-Headers: Link

{
  "token": 16782xxxxxxxxxxxxxxxx,
  "data": {
    "targets": [{ "id": "631b5ef7bc2f571ac64f4835", "ratio": "1", "type": "http", .....
    "randoms": [{"name": "Телефон", "id": "62d8286fd.....
  }
}
```

### The new form of returned configuration

The new configuration supports four attack types: `http`, `http2`, `nginx_loris`, and `tcp`. The rest of the items are the same as SentinelLabs, and we described previously; see [C&C Communication](#) and [SentinelLabs](#).

When the login and get targets operation are successful, the bot creates approximately 20 working threads that perform the DDoS attack using a synchronized method since the bot counts the number of successful connections. Therefore, the bot waits for an attacked server response; if the response is `200`, the attempt is counted as a successful attack. Subsequently, the newest bot implementation is eight times faster than the initial implementation in Python.

Continuous statistics are sent each ~four minutes back to the C2 server through `POST /set_attack_count`.

```
POST /set_attack_count HTTP/1.1
Host: 94.140.114.239
User-Agent: Go-http-client/1.1
Content-Length: xxxxxx
Client-Hash: 7d70614da685365d9d2fd609ddde0c6aeda77157a858e080a0a8747d4afd72e9:5481
Content-Type: text/plain; charset=utf-8
Time: 16782xxxxxxxxxxxxxxxx
User-Hash: $2a$16$xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Accept-Encoding: gzip, deflate
Connection: close

[{"id":"7958b75b18f2aa7ec","total":86,"success":86},{"id":"c2bc0b7adafe3019f129", ...}]
```

### *Sending the attacks' statistics back to C2*

These statistics help the attacker track the target configuration's effectiveness and respond in time with a new configuration. If everything goes as expected, a new target configuration is requested every ~10 minutes. However, sometimes the C2 server is unable to handle requests, and a connection cannot be established. In this case, the bot continues on the latest target configuration and tries to contact C2 later. **Figure 1** provides an overview of the communication between the C2 server and a bot.

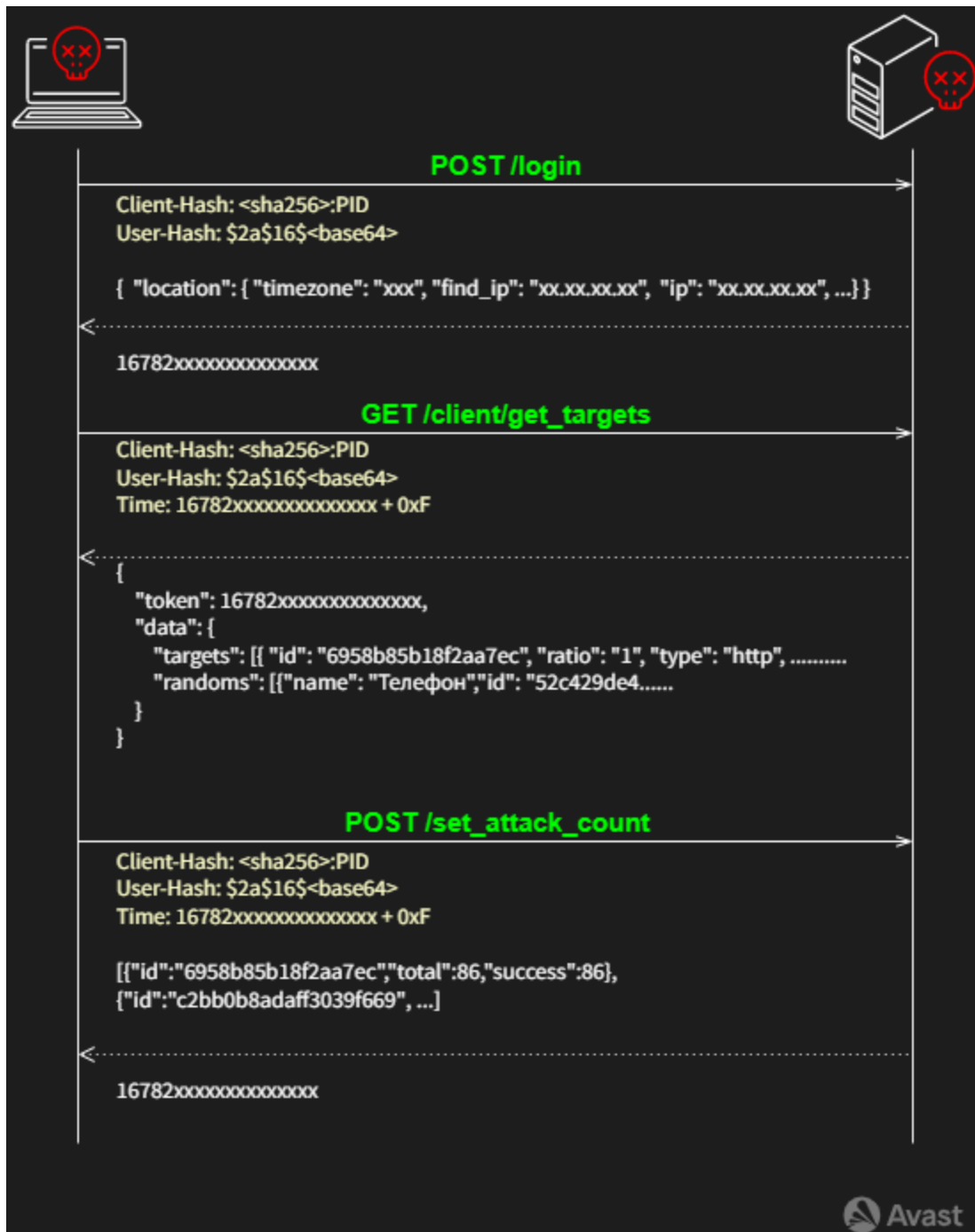


Figure 1. C2

communication workflow

## Bot Updater

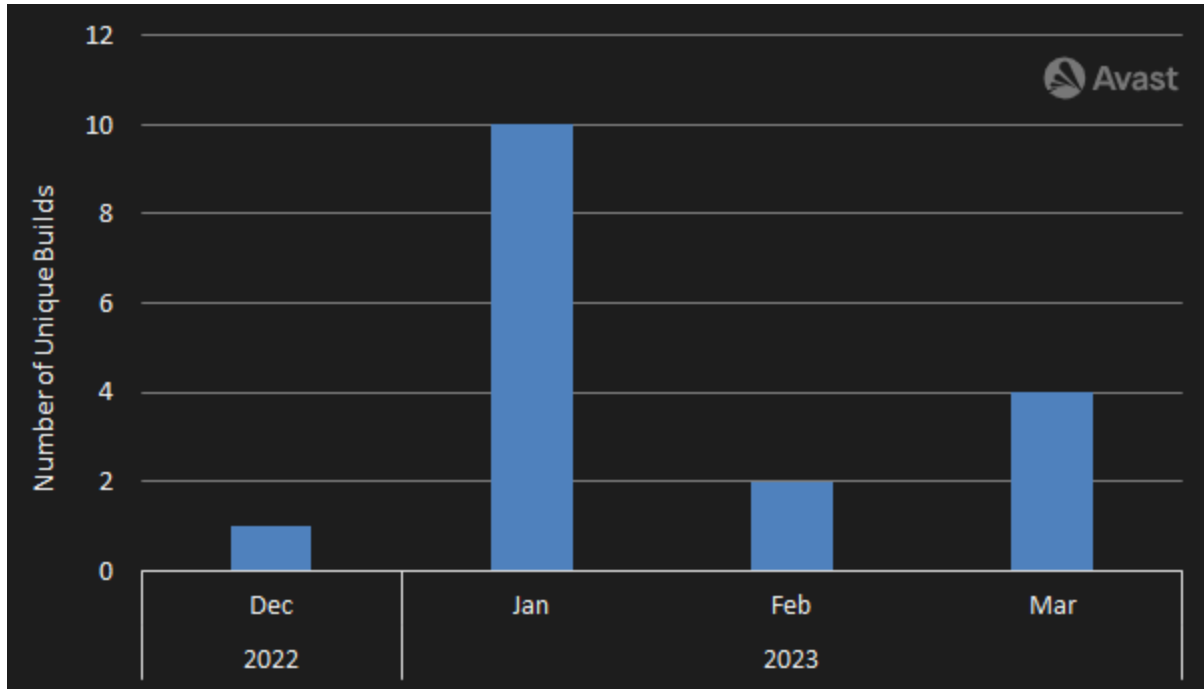
One of the unknowns remains the question of the bot updates. Our investigations into this area are still in progress, and we are trying to confirm our hypothesis that there is an automatic bot updater.

We've observed a few takedowns of C2 servers and new build releases in the last months. We expected a delay of several days between the bot updates and further DDoS attacks. However, the time between the C2 takedown and the new DDoS attacks was several hours.



Therefore, our hypothesis is that there is an automatic updater since it is improbable to manually update approximately 7,200 independent clients within several hours.

The count of new bot releases was considerable in the last four months, as **Figure 2** illustrates. So, there should be some automatic updater.



**Figure 2.** DDosia executable hits

## C2 Protection

All C2 servers have used HTTP protocol to communicate, which was unencrypted. So, it was only a matter of time before the DDosia authors tried to implement a mechanism to protect the target configurations.

### Temporary DNS Records

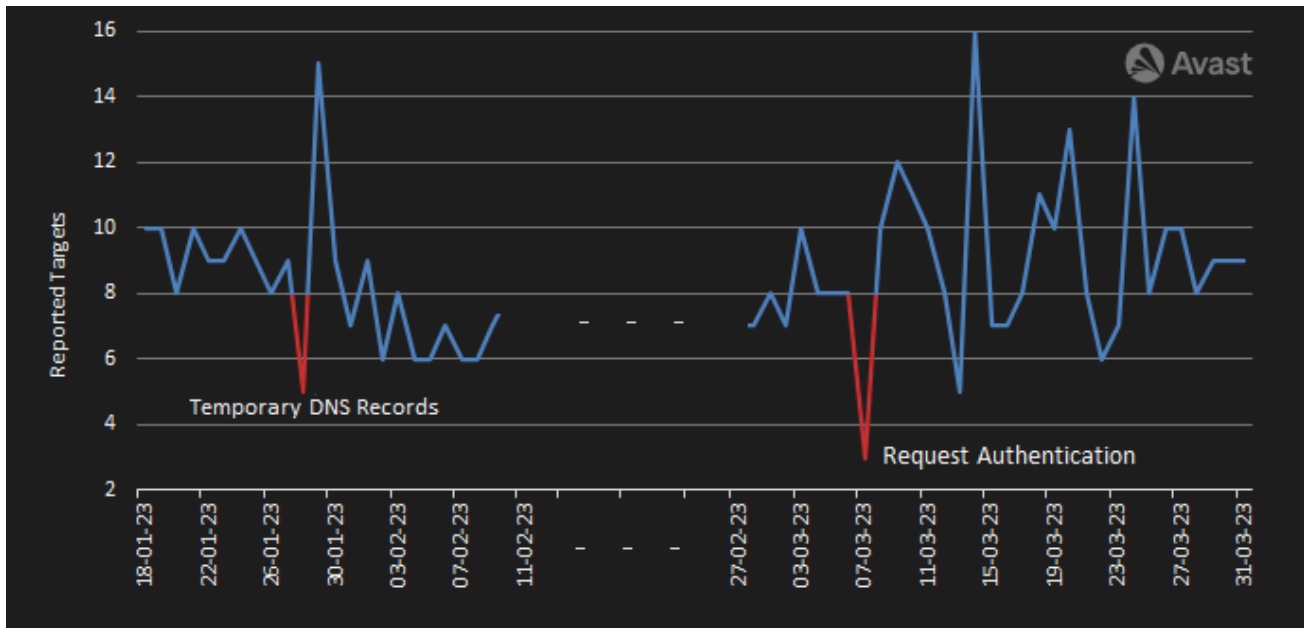
The first attempt to implement the protection mechanism was around January 28, 2023. The main idea was to use temporary DNS records, which are rotated every midnight. The DNS record is then reconfigured to a non-existent record. As a result of the 24-hour period, the initial DNS record is not captured by any online monitoring services, so the history of DNS records includes only the non-existent or invalid records. Consequently, the valid IP address of C2 server is not recorded anywhere, and it would not be easy to find them.

This mechanism has been seen in the cases on January 28-29, 2023. Two builds with hardcoded DNS records were set to non-existent IPs after midnight. The next day, the new builds with new DNS records were released.

For example, `deac48f968269bb5e75ceb9417f6680d8787a03ba0768e25d8716f189a079574` build has two DNS records (`pkcds2cas7.ignorelist.com`, `pqw2vsi21mx74.twilightparadox.com`) that led to `212.73.134.208`. However, the DNS records were reconfigured to `127.0.0.2` from midnight on January 27-28, 2023. So, if you resolve the DNS records today, you cannot resolve the initial IP since the address is already untraceable.

The same case was seen from midnight on January 28-29, 2023, on the `5c1be24a5fa75b70c50515a061d2d3ce911b785188328408edc0a79dfc5e3173` build. The other two DNS records (`trafficsearch.ddns.net`, `trafficanalyzer.bounceme.net`) led to `94.140.115.129`. The DNS records were also reconfigured to invalid IP addresses; namely `0.0.0.0`.

Implementing this mechanism was probably not successful because the count of reported targets on the group telegram was lower on January 28, as **Figure 3** demonstrates. Moreover, there were reported taken-down domains from the previous target configuration. Finally, the build that was released on January 30 contained hard-coded IP addresses of the C2 server (`94.140.114.239`).



Request Authentication

The second attempt to implement the protection mechanism was on March 7, 2023. The communication with the C2 server is also via HTTP, but a token mechanism was designed and realized. Therefore, anybody cannot download the target configuration (list of attacked domains) freely without authentication as before.



Figure 4.

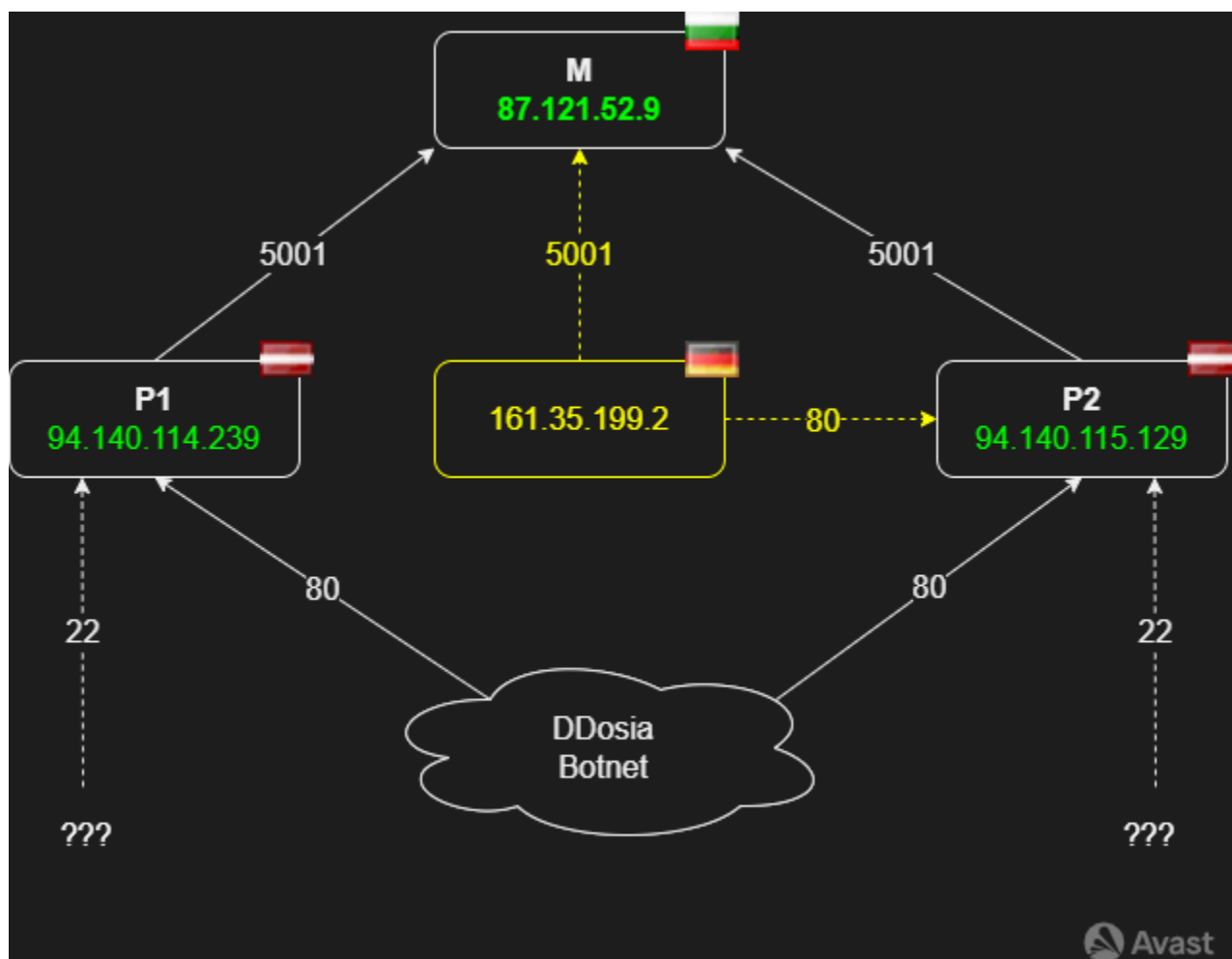
#### Authentication mechanism

The first communication with the C2 server is the login request, as described above in **Figure 4**. First, the request must include the header **User-Hash**, which users obtain during the registration process in the DDosia Project Telegram channel. The other necessary condition is data about the GeolP of the bot. If the IP address or ISP of the given bot is on the blacklist (e.g. Avast), the authentication process ends with **401 Unauthorized**. However, if the authentication is successful, the login request reruns the token in the string form.

The token is valid for approximately 15 minutes, and the constant **0xF** must be added each time the token is used for the following requests to the C2 servers. The adjusted token is included in the HTTP header as a **Time** entry, and each response then consists of a new fresh token value.

## C2 Architecture

The C2 architecture is dynamically changing. We noticed four IP addresses related to the DDosia project since the beginning of 2023. Three addresses are active web servers run on Ubuntu using **nginx/1.18.0**. More importantly, these web servers return the same target configurations and provide the services like logging into the botnet as well as reporting statistics to the attackers. The currently discovered C2 architecture of the DDosia project is shown in **Figure 5**.



**Figure 5.** C2 architecture

Using HTTP, the central C2 server (M) is contacted by proxy C2 servers P1 and P2 throughout port 5001. The DDosia bots reach the proxy servers also using HTTP via port 80, where requests are forwarded to the central server. Any suspicious outcome from the primary server has not been detected yet. However, one suspicious server or client communicates, especially with the primary and P2 servers. We recorded the most network activity of a suspicious IP (161.35.199.2) around February 14, 2023. The purpose of this suspicious machine is still unknown, but it can be a testing or monitoring service.

Besides the bots' communication over port 80, we detected connections on port 22 for both proxy servers. The transmission on port 22 is not implemented in the bot executables we analyze, but our telemetry indicates a higher communication volume. However, most captured IPs contacting port 22 are suspicious due to port scans or SSH brute force attacks.

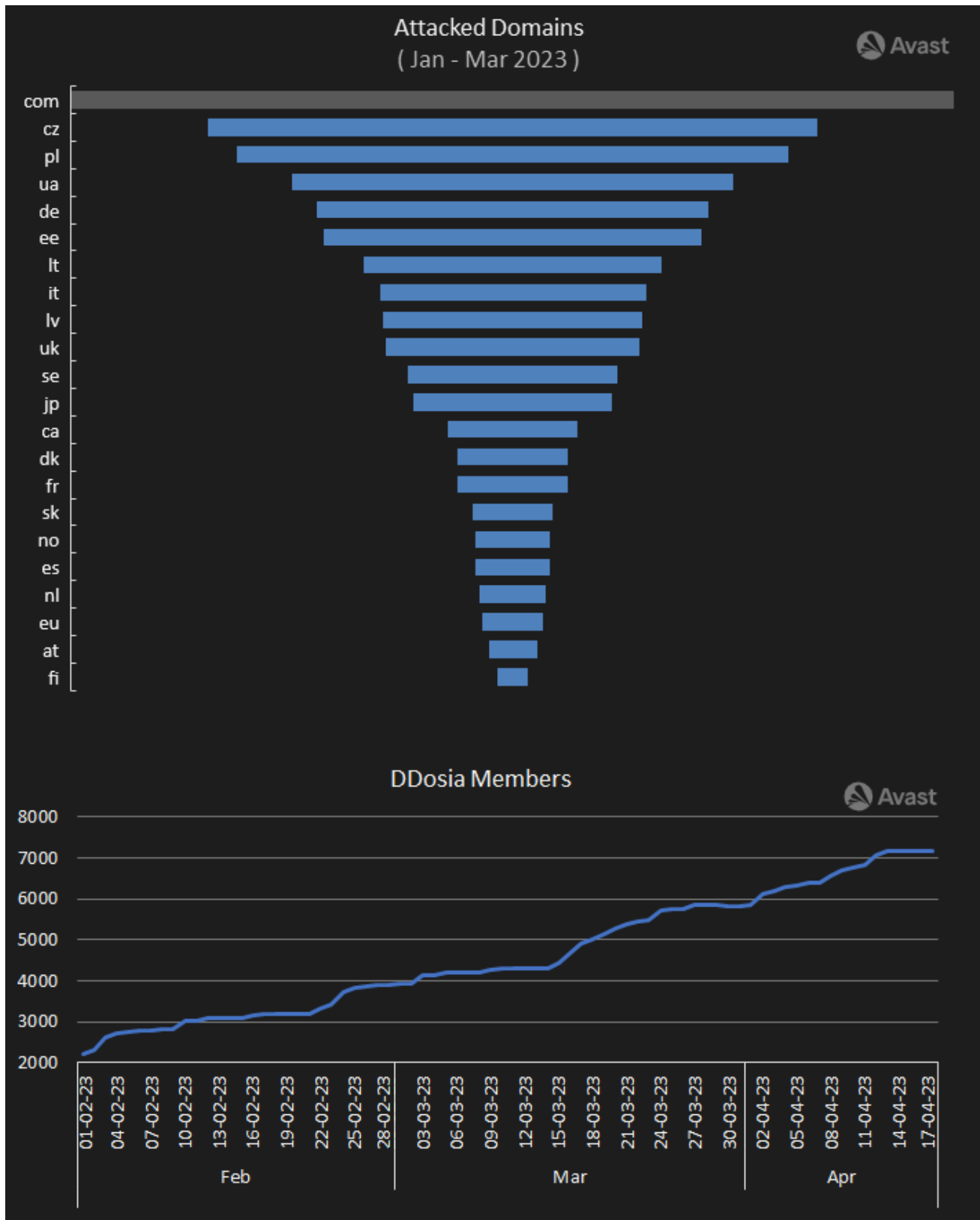
In addition, the C2 infrastructure relies heavily on proxy servers which contributes to the resilience of DDosia's infrastructure. Nevertheless, our monitoring revealed that outages indicated by 502 Bad Gateway error responses from the proxy servers. The first significant disruption occurred during the deployment of the authentication mechanism. The outage

lasted for several hours – the duration and the timing indicates that development issues may have been responsible. The root of the problem seems to be partially fixed as recent outages were resolved within one hour.

## DDosia Tracking

---

We still continue to monitor the DDosia project targets and the count of users that have joined the project. We will publish detailed information about the targets, configurations, and volunteers in a subsequent post. **Figure 6** illustrates a quick overview.



**Figure 6.** Attacked countries and trend of the joined users

We've also observed that DDosia's community is steadily growing, though there can be doubts about the capacity new members can contribute. Nevertheless, it seems that in this specific case, a volunteer-based model is rather efficient and easier to manage than a

malware-based botnet; however, its availability is probably enabled by the political circumstances.

## Conclusion

---

It is evident that the project is still in development, and NoName057(16) is trying to improve the efficiency of the DDoS attacks. They are trying to move to a more efficient Go platform because the pilot variant written in Python was lacking in performance.

Many of the changes seem to be motivated by protecting the target configuration and C2 architecture secrecy. Hence, the latest version of DDosia bots has realized the authentication mechanism for C2 communication.

Our most interesting observation was probably the implementation of an update mechanism in the client since previous updates caused only short-term disruptions to the project's effectiveness. This has also increased the resilience of the C2 mechanism, as it is no longer necessary to do a manual update after a server takedown. The update mechanism is still under our investigation. In a future blog post, we plan to release a more detailed analysis of the tracker's historical data.

## References

---

Tagged [asbotnet](#), [ddos](#)