

OSINT & Reverse Engineering Report

Prepared by: Narendra Singh Naruka

OSINT Analysis

Objective:

- Identify the email address of Monkey D. Tuffy.

Methodology:

- Tool Used: WhatsMyName (<https://whatsmyname.app/>)
- Using **WhatsMyName** Web, we tried multiple usernames and we got a GitHub repository.

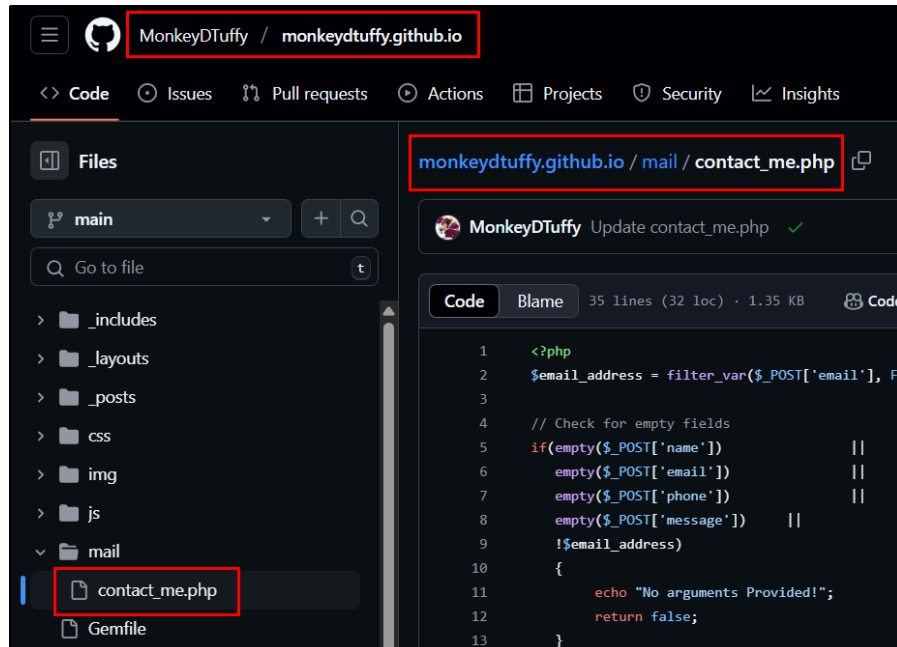
The screenshot shows the 'WhatsMyName Web' interface. At the top, there's a search bar with 'MonkeyDTuffy' entered. Below the search bar, it says 'Active Filter: All (exclude NSFW)'. The results are displayed in a grid of cards for various platforms: GitHub, Linktree, MyAnimeList, scratch, and X. Each card shows the username 'MonkeyDTuffy' and the category 'Account Found'. To the right, there's a table titled 'Filter by Username: MonkeyDTuffy' with columns for SITE, USERNAME, CATEGORY, and LINK. The first row in the table is for GitHub, with the link <https://github.com/MonkeyDTuffy> highlighted in red.

SITE	USERNAME	CATEGORY	LINK
GitHub	MonkeyDTuffy	coding	https://github.com/MonkeyDTuffy
Linktree	MonkeyDTuffy	social	https://linktr.ee/MonkeyDTuffy
MyAnimeList	MonkeyDTuffy	social	https://myanimelist.net/profile/MonkeyDTuffy
scratch	MonkeyDTuffy	coding	https://scratch.mit.edu/users/MonkeyDTuffy/
X	MonkeyDTuffy	social	https://x.com/MonkeyDTuffy

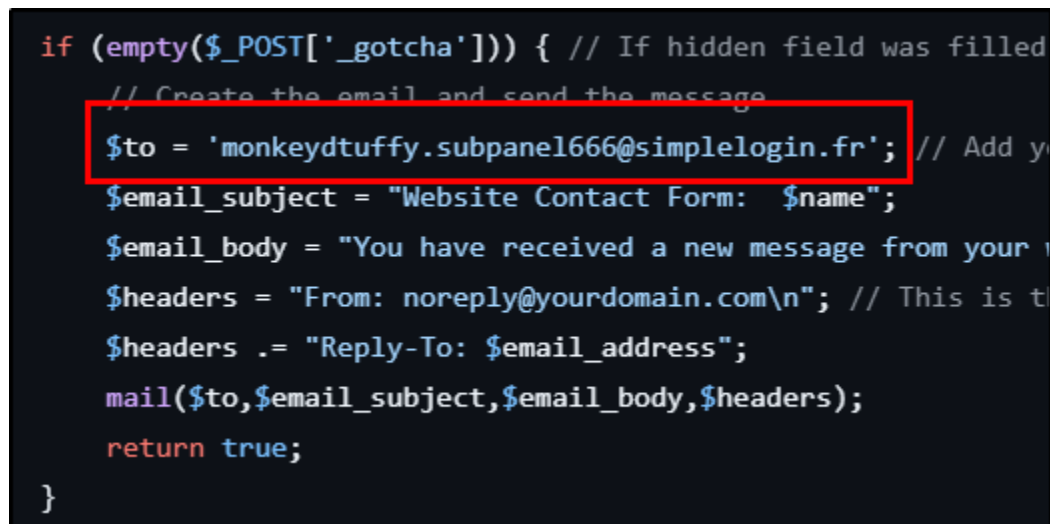
- Active GitHub profile: <https://github.com/MonkeyDTuffy?tab=repositories>

The screenshot shows the GitHub profile page for 'MonkeyDTuffy'. The profile picture is a circular image of a character with red hair and a red bandana. The page shows three repositories: 'ABC-Corp' (Public, Forked from meta-llama/llama3, Python, Updated on Jun 7, 2024), 'MonkeyDTuffy' (Public, Config files for my GitHub profile, Updated on Jun 18, 2024), and 'monkeydttuffy.github.io' (Public, JavaScript, MIT License, Updated on Jun 19, 2024).

- **Repository Analysis:**
- Let's try some enumeration. After some enumeration, we got an interesting repository, [monkeydtuffy.github.io](https://github.com/monkeydtuffy/monkeydtuffy.github.io).
- We navigated to the [monkeydtuffy.github.io](https://github.com/monkeydtuffy/monkeydtuffy.github.io) repo and searched the HTML/JS source. In [contact.php](#).



- We found the email address embedded in the mail '**\$to**' variable.



- Email found: **monkeydtuffy.subpanel666@simplelogin.fr**
- So finally we got the email address of **Monkey D. Tuffy**. Objective achieved.

Reverse Engineering

Step 1: Setup

- Download the zip file
- Extract the zip

unzip DiffuseTheConfusion.zip

```
(root@kali)-[/home/kali/Downloads/reveng]
# unzip DiffuseTheConfusion.zip
Archive: DiffuseTheConfusion.zip
replace obf.js? [y]es, [n]o, [A]ll, [N]one, [r]ename: y
  inflating: obf.js
```

- There is an [obf.js](#) Javascript file that is obfuscated.
- Now we need to deobfuscate it.

Step 2: Deobfuscation

- For this task, we use **Obfuscator.io Deobfuscator**
 - Link: <https://obf-io.deobfuscate.io/>

- We Deobfuscated the JavaScript code to make it human-readable
- Now we need to understand how this code works.

Step 3: Execution

- Ran the code using Node.js

```
node obf.js
```

- After running this, we got an error for pass.txt not found.
- So we create a pass.txt and set up a web server that responds to JS code when it fetches pass.txt.

```
(root@kali)-[/home/kali/Downloads/reveng]
# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
192.168.21.129 - - [26/Jun/2025 13:09:41] "GET /pass.txt HTTP/1.1" 200 -
192.168.21.129 - - [26/Jun/2025 13:11:39] "GET /pass.txt HTTP/1.1" 200 -
```

- Again we run the code using the command

```
node obf.js
```

- Then we got another error that “prompt is not defined”
- So Replaced the `prompt()` call with a hardcoded password:
 - Like
 - `var _0x4ca3cf = "password123";`

```
fetch('http://192.168.21.129/pass.txt', _0x4a50fd).then(_0x2e50f9 => _0x2e50f9.json  
()).then(_0x5bb73f => {  
  var _0x4ca3cf = "password123";  
  if (_0x4ca3cf !== '' && _0x5bb73f !== '' && _0x4ca3cf === _0x5bb73f) {  
    _0x4ca3cf = _0x32a02b(_0x4ca3cf);  
    const _0x5bb2d4 = [0x68, 0x61, 0x72, 0x64, 0x63, 0x6f, 0x64, 0x65, 0x64, 0x5f,  
      0x70, 0x61, 0x73, 0x73, 0x77, 0x6f, 0x72, 0x64];  
    _0x4ca3cf += _0x22f02a(_0x5bb2d4);  
    _0x4ca3cf = new TextEncoder().encode(_0x4ca3cf);  
  }  
})
```

- Now we run the command and get the flag.
 - **Flag:** `flag{0h_Y0u_C3@Ked_M4}`

```
(root@kali)-[/home/kali/Downloads/reveng]  
# node copy_obf.js  
Flag: flag{0h_Y0u_C3@Ked_M4}
```

Conclusion

- Through a combination of open-source intelligence gathering and reverse engineering, we achieved both objectives:
 - a. Found the email address of the threat actor
 - b. Decrypted the obfuscated JavaScript to extract the flag
- Email Discovered: `monkeydtuffy.subpanel666@simplelogin.fr`
- Flag Recovered: `flag{0h_Y0u_C3@Ked_M4}`

Report Date: 26 June 2025
Prepared by: Narendra Singh Naruka