VPN Clients Security Testing

Linux Client Validation Retest

NordVPN

Friday, March 19, 2021

VerSprite OffSec Team

VERSPRITE

Table of Contents

Validation Retest Matrix

Vulnerability	Status	Summary
Lack of Authentication on nordvpnd.sock leads to DoS (CWE-248)	Remediated	We verified that proper permissions are being set on the NordVPN daemon's Unix Domain Socket which prevents any unauthorized user to interact with it. In addition, we verified that the Denial of Service and RACE condition exploits are no longer working.
Transport layer security (TLS) v1.0 and v1.1 supported (CWE-326)	Remediated	We verified that protocols TLSv1.0 and TLSv1.1 are no longer used.
Lack of Memory Protections (CWE-693)	Mitigated	We found that some of the NordVPN client binaries were still missing some of the available compile time protections. However, the new binaries made use of more protections than the previous version, indicating good progress in this direction.

Figure 1 - Validation Retest Matrix

(Remediated) Lack of Authentication on nordvpnd.sock leads to DoS (CWE-248) – Medium

Validation Retest Notes

During the validation retest, we verified that it is no longer possible for any unauthorized user on the system to interact with the NordVPN daemon's Unix Domain Socket due to proper access permissions being set during the setup as can be seen in the following excerpt.

```
uid0@0x75696430:~$ sudo ls -lha /run/nordvpn*
total 0
drwxrwx---  2 root nordvpn  60 Mar 16 11:18 .
drwxr-xr-x 36 root root    860 Mar 17 00:11 ..
srwxrwx---  1 root nordvpn   0 Mar 16 11:18 nordvpnd.sock
```

Another change that we can spot above is the use of a new group named *nordvpn*. This would prevent any user outside of that group to interact with the socket as shown below.

```
uid0@0x75696430:~$ nordvpn settings
Whoops! Permission denied accessing /run/nordvpn/nordvpnd.sock
```

In order to use the client, the user has to be added to the group, as follows.

```
uid0@0x75696430:~$ sudo usermod -aG nordvpn $USER
[sudo] password for uid0:
*After reboot*
uid0@0x75696430:~$ nordvpn settings
Technology: OpenVPN
Protocol: UDP
Kill Switch: disabled
CyberSec: disabled
Obfuscate: disabled
Notify: enabled
Auto-connect: disabled
DNS: disable
```

In addition, it is still possible for users -within the *nordvpn* group- to perform changes such as:

- Change the VPN technology and protocol in use
- Disconnect the NordVPN client

```
uid0@0x75696430:~$ sudo usermod -aG nordvpn pepito
uid0@0x75696430:~$ su pepito
Password:
pepito@0x75696430:/home/uid0$ nordvpn status
Status: Connected
Current server: ar43.nordvpn.com
Country: Argentina
City: Buenos Aires
Your new IP: 131.255.4.232
Current technology: OpenVPN
Current protocol: UDP
Transfer: 126 B received, 5.98 KiB sent
Uptime: 6 minutes 1 second
pepito@0x75696430:/home/uid0$ nordvpn disconnect
You are disconnected from NordVPN.
How would you rate your connection quality on a scale from 1 (poor) to 5 (excellent)? Type 'nordvpn rate [1-5]'.
```

Nevertheless, we consider the nature of the aforementioned a mere design decision without a security impact due to the newly access permissions set on the Unix Domain Socket.

Finally, we tested both exploits, the Denial of Service and the reconnect RACE condition on two fresh NordVPN installs on Kali 2020.3 and Ubuntu 20.04.2.0 LTS and verified that the NordVPN daemon behave as expected without presenting any error conditions or crashes. The following screenshots show an example of the tests.

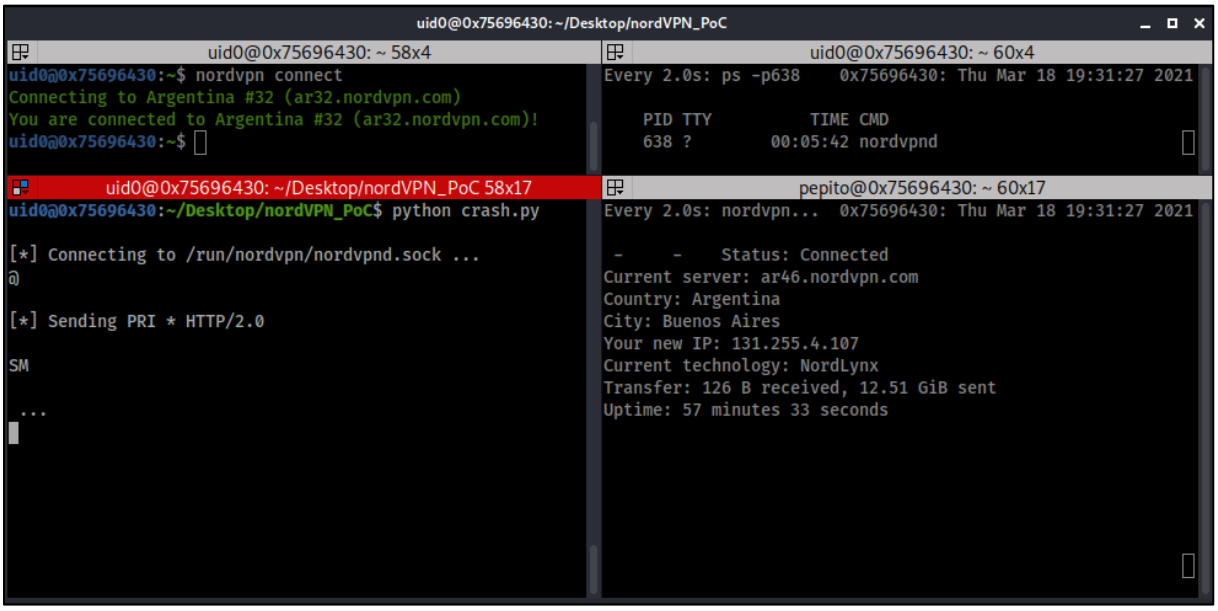

Figure 2 - Exploit attempts on Kali

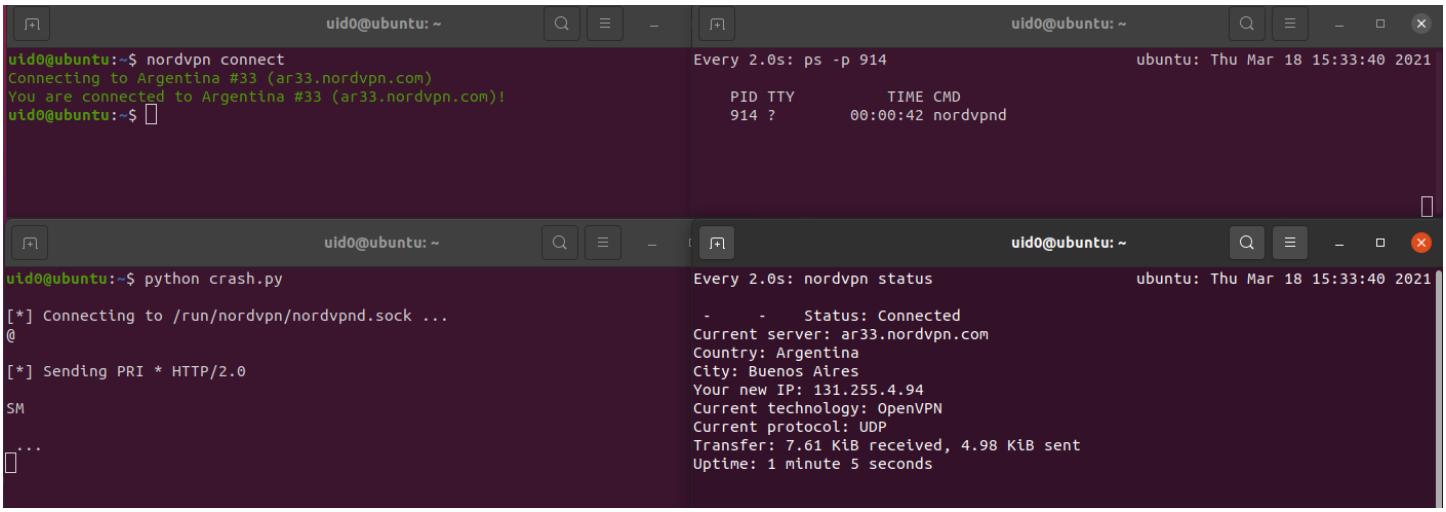

Figure 3 - Exploit attempts on Ubuntu

(Remediated) Transport layer security (TLS) v1.0 and v1.1 supported (CWE-326) – Low

Validation Retest Notes

We verified that the reported TLS protocol versions are no longer in use, as can be seen in the following excerpt.

```
[redacted]:~$ sslscan https://zwyr157wwiu6eior.com
Version: 2.0.4-static
OpenSSL 1.1.1i-dev  xx XXX xxxx

Connected to 2606:4700::6811:cf66

Testing SSL server zwyr157wwiu6eior.com on port 443 using SNI name zwyr157wwiu6eior.com

  SSL/TLS Protocols:
SSLv2     disabled
SSLv3     disabled
TLSv1.0   disabled
TLSv1.1   disabled
TLSv1.2   enabled
TLSv1.3   enabled

  Supported Server Cipher(s):
Preferred TLSv1.3  128 bits  TLS_AES_128_GCM_SHA256        Curve 25519 DHE 253
Accepted  TLSv1.3  256 bits  TLS_AES_256_GCM_SHA384        Curve 25519 DHE 253
Accepted  TLSv1.3  256 bits  TLS_CHACHA20_POLY1305_SHA256  Curve 25519 DHE 253
Preferred TLSv1.2  256 bits  ECDHE-ECDSA-CHACHA20-POLY1305 Curve 25519 DHE 253
Accepted  TLSv1.2  128 bits  ECDHE-ECDSA-AES128-GCM-SHA256 Curve 25519 DHE 253
Accepted  TLSv1.2  128 bits  ECDHE-ECDSA-AES128-SHA        Curve 25519 DHE 253
Accepted  TLSv1.2  128 bits  ECDHE-ECDSA-AES128-SHA256     Curve 25519 DHE 253
Accepted  TLSv1.2  256 bits  ECDHE-ECDSA-AES256-GCM-SHA384 Curve 25519 DHE 253
Accepted  TLSv1.2  256 bits  ECDHE-ECDSA-AES256-SHA        Curve 25519 DHE 253
Accepted  TLSv1.2  256 bits  ECDHE-ECDSA-AES256-SHA384     Curve 25519 DHE 253
Accepted  TLSv1.2  256 bits  ECDHE-RSA-CHACHA20-POLY1305   Curve 25519 DHE 253
Accepted  TLSv1.2  128 bits  ECDHE-RSA-AES128-GCM-SHA256   Curve 25519 DHE 253
Accepted  TLSv1.2  128 bits  ECDHE-RSA-AES128-SHA          Curve 25519 DHE 253
Accepted  TLSv1.2  128 bits  ECDHE-RSA-AES128-SHA256       Curve 25519 DHE 253
Accepted  TLSv1.2  128 bits  AES128-GCM-SHA256
Accepted  TLSv1.2  128 bits  AES128-SHA
Accepted  TLSv1.2  128 bits  AES128-SHA256
Accepted  TLSv1.2  256 bits  ECDHE-RSA-AES256-GCM-SHA384   Curve 25519 DHE 253
Accepted  TLSv1.2  256 bits  ECDHE-RSA-AES256-SHA          Curve 25519 DHE 253
Accepted  TLSv1.2  256 bits  ECDHE-RSA-AES256-SHA384       Curve 25519 DHE 253
Accepted  TLSv1.2  256 bits  AES256-GCM-SHA384
Accepted  TLSv1.2  256 bits  AES256-SHA
Accepted  TLSv1.2  256 bits  AES256-SHA256
  SSL Certificate:
Signature Algorithm: ecdsa-with-SHA256
ECC Curve Name:      prime256v1
ECC Key Strength:    128

Subject:  sni.cloudflaressl.com
Altnames: DNS:zwyr157wwiu6eior.com, DNS:sni.cloudflaressl.com, DNS:*.zwyr157wwiu6eior.com
Issuer:   Cloudflare Inc ECC CA-3

Not valid before: Aug 14 00:00:00 2020 GMT
Not valid after:  Aug 14 12:00:00 2021 GMT
```

We performed another test using Qualys SSL Labs on every IP resolved by the domain which returned a A+ score on each of them.

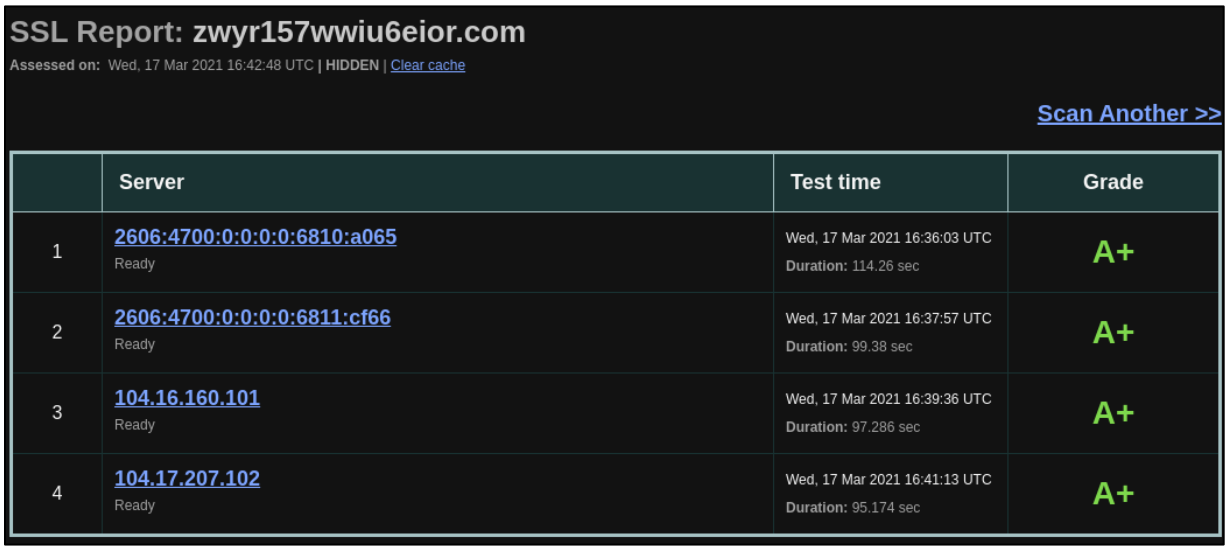

Figure 4 - SSL Labs results

(Mitigated) Lack of Memory Protections (CWE-693) – Low

Validation Retest Notes

During the validation retest, we found that some of the NordVPN client binaries were still missing some of the available compile time protections. However, the new binaries made use of more protections than the previous version, indicating good progress in this direction.

The following output of the *hardening-check* tool shows the improvements, as well as the protections that are still not in use:

```
uid0@0x75696430:~$ ./hardening-check /usr/bin/nordvpn
/usr/bin/nordvpn:
 Position Independent Executable: yes
 Stack protected: no, not found!
 Fortify Source functions: yes
 Read-only relocations: yes
 Immediate binding: yes

uid0@0x75696430:~$ ./hardening-check /usr/sbin/nordvpnd
/usr/sbin/nordvpnd:
 Position Independent Executable: yes
 Stack protected: no, not found!
 Fortify Source functions: yes (some protected functions found)
 Read-only relocations: yes
 Immediate binding: yes

uid0@0x75696430:~$ ./hardening-check /var/lib/nordvpn/openvpn
/var/lib/nordvpn/openvpn:
 Position Independent Executable: yes
 Stack protected: yes
 Fortify Source functions: yes (some protected functions found)
 Read-only relocations: yes
 Immediate binding: yes
```

We recommend enabling the stack protection mechanism by supplying the following flag to *go build*:

```
export CGO_CPPFLAGS="-D_FORTIFY_SOURCE=2 -fstack-protector-all"
```