# VPN Clients Security Testing

## macOS – Validation Retest

### NordVPN

**Monday, March 30, 2021**

**VerSprite OffSec Team**

# Table of Contents

## *Validation Retest Matrix*

| Vulnerability | Status | Summary |
|---|---|---|
| *Real IP Leakage via Local Socket Binding (CWE-200)* | Remediated | The updated NordVPN macOS application implements a "Kill Switch" option that prevents programs from using the local network interface. Therefore, Internet hosts cannot be reached from other than the VPN interface. |
| *RealmDB Hardcoded Encryption Key (CWE-321)* | Remediated | The issue was remediated. After running the same application build on two different macOS computers, and inspecting the parameters passed to the encryption/decryption functions we observed they were using different keys. |
| *Information Disclosure in Binary Files (CWE-615)* | Remediated | The issue was remediated. We did not find internal or sensitive information in the application's binaries files. |

***Table 1 – Validation Retest Matrix***

## Technical Details

### (Remediated) Real IP Leakage via Local Socket Binding (CWE-200) – High

**Validation Retest Notes**

During the retest we found that this vulnerability was remediated through the implementation of a “Kill Switch” functionality that can be manually enabled by users to fully protect their real IP address in *NordVPN_[redacted]_6.1.1.pkg* version, by preventing programs to use the local network interface of the host to reach Internet hosts.

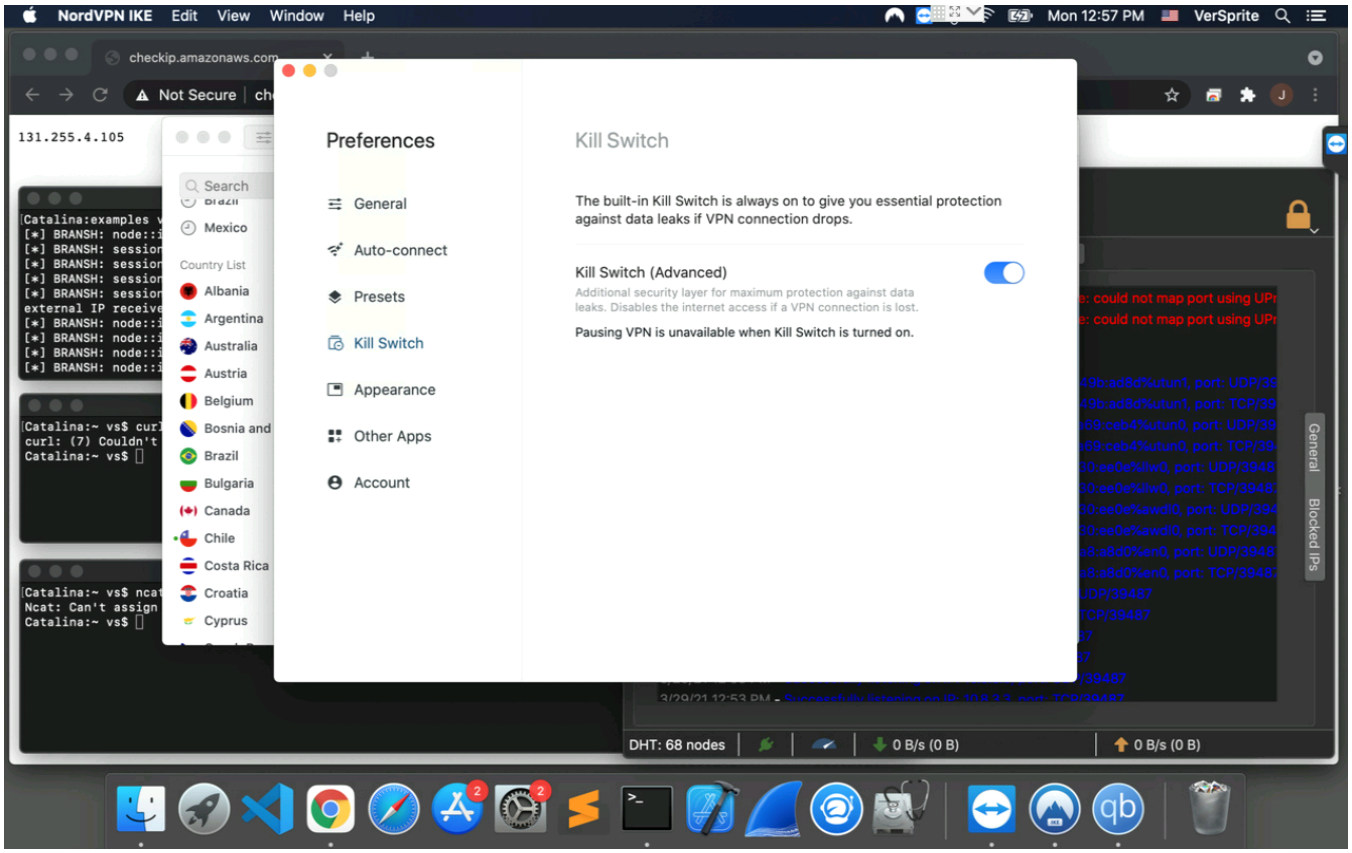


***Figure 1 – Kill Switch functionality enabled.***

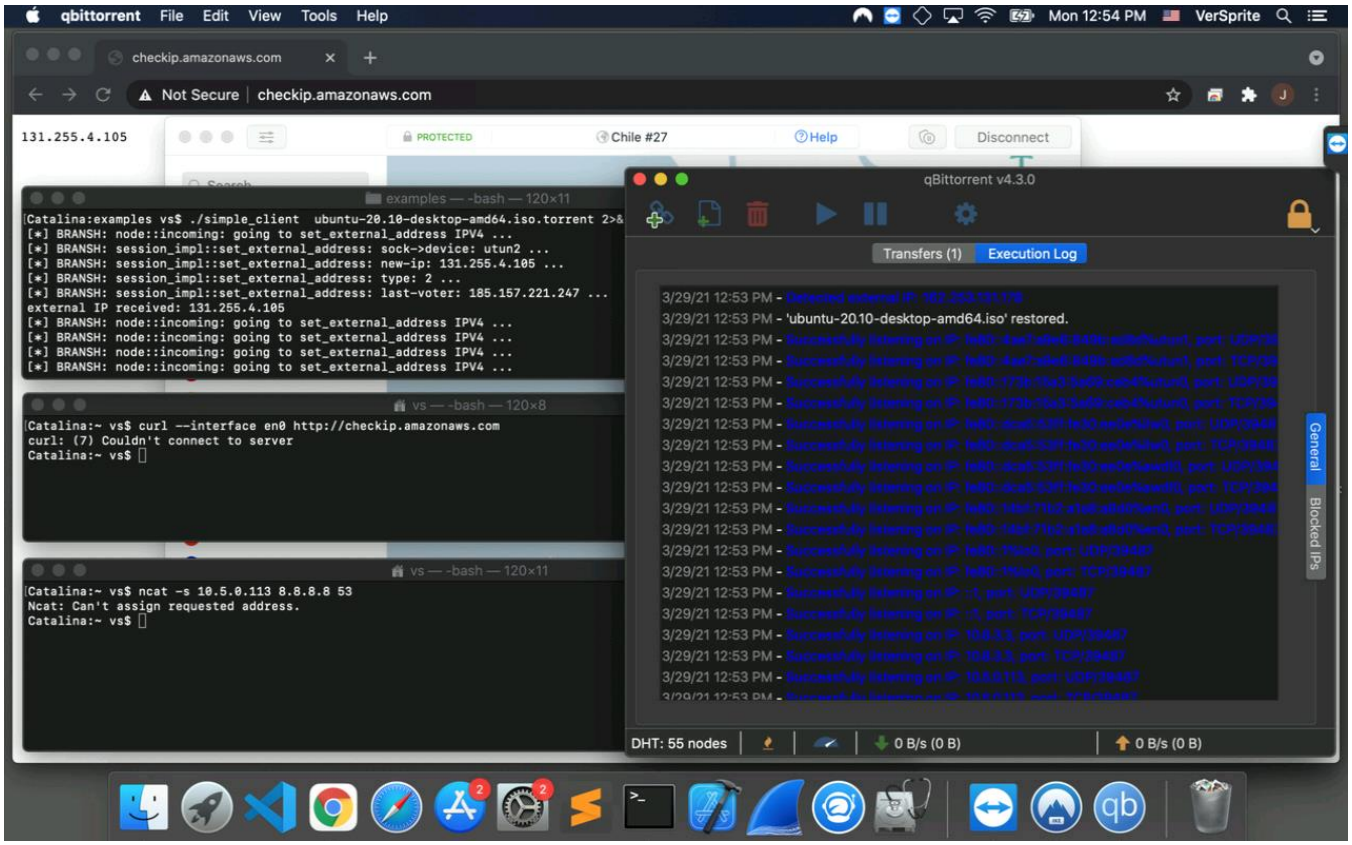


***Figure 2 – Could not use the local interface to leak the real IP address of the host.***

*Note: During the tests, our first attempts successfully leaked the real IP address of the host, even with the Kill Switch enabled. However, after testing it again, along with NordVPN personnel, we could not leak the IP using the same methods we were using initially. The first result may be related to the version of the plugin used by NordVPN solution staying old during the application removing and reinstallation procedures.*

## (Remediated) RealmDB Hardcoded Encryption Key (CWE-321) – Low

**Validation Retest Notes**

During the retest we found that this vulnerability is no longer affecting the application. We installed the same build (NordVPN_[redacted]_6.1.1.pkg) on two different macOS computers, run the NordVPN software and then hooked the calls to the *setEncryptionKey* method of the class *RLMRealmConfiguration* to inspect the content of the arguments passed as parameters to the method. The inspection of arguments allowed us to see in plaintext the password for the RealmDB database used by the NordVPN installation. This time, we observed that both instances were using different keys for encrypting/protecting the information in the database.

*Inspecting macOS computer #1:*

```
sh-3.2# ifconfig | grep inet
        inet 127.0.0.1 netmask 0xff000000
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
        inet6 fe80::4d2:31c2:aba0:c965%en0 prefixlen 64 secured scopeid 0x5
        inet 10.5.0.113 netmask 0xffff0000 broadcast 10.5.255.255
        inet6 fe80::ac30:33ff:fe5c:616b%awdl0 prefixlen 64 scopeid 0x7
        inet6 fe80::ac30:33ff:fe5c:616b%llw0 prefixlen 64 scopeid 0x8
        inet6 fe80::b513:f20e:6585:bdd0%utun0 prefixlen 64 scopeid 0x9
        inet6 fe80::1a7f:c691:5875:5b63%utun1 prefixlen 64 scopeid 0xa
        inet 10.7.2.4 --> 10.7.2.4 netmask 0xffffff00
        inet 10.8.0.10 --> 10.8.0.10 netmask 0xffffff00
        inet6 fe80::dea9:4ff:fe95:5e3e%utun3 prefixlen 64 scopeid 0xc
        inet6 fd00::1 prefixlen 64

sh-3.2# python3 macos_inspector.py "NordVPN" find_args RLMRealmConfiguration "- setEncryptionKey:"
[!] Ctrl+D or Ctrl+Z to detach from instrumented program.


[*] About to hook RLMRealmConfiguration->- setEncryptionKey: ...
[+] Detected call to "setEncryptionKey:" of the class:RLMRealmConfiguration {
        fileURL = file:///[redacted]/Library/Application%20Support/com.nordvpn.osx/default.encrypted_v2.realm;
        inMemoryIdentifier = (null);
        encryptionKey = (null);
        readOnly = 0;
        schemaVersion = 0;
        migrationBlock = (null);
        deleteRealmIfMigrationNeeded = 0;
        shouldCompactOnLaunch = (null);
        dynamic = 0;
        customSchema = (null);
}
[+] Arguments passed as parameters:
        [*] arg2 Objective C type: Foundation.__NSSwiftData
        [*] arg2: {length = 64, bytes = 0x36306562 63306565 38323932 61653665 ... 65323635 37616237 }
        [*] arg2: 60ebc0ee82[redacted]
```

*Inspecting macOS computer #2:*

```
sh-3.2# ifconfig | grep inet
        inet 127.0.0.1 netmask 0xff000000
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
        inet6 fe80::1849:6033:94d6:25d3%en0 prefixlen 64 secured scopeid 0x7
```

```
        inet 10.5.0.116 netmask 0xffff0000 broadcast 10.5.255.255
        inet6 fe80::992:23eb:1ede:90fc%utun0 prefixlen 64 scopeid 0x8
        inet6 fe80::8693:f3a3:df12:da60%utun1 prefixlen 64 scopeid 0x9
        inet 10.8.1.9 --> 10.8.1.9 netmask 0xffffff00
        inet6 fe80::20c:29ff:fe8f:a47e%utun2 prefixlen 64 scopeid 0xa
        inet6 fd00::1 prefixlen 64

sh-3.2# python3 macos_inspector.py "NordVPN" find_args RLMRealmConfiguration "- setEncryptionKey:"
[!] Ctrl+D or Ctrl+Z to detach from instrumented program.


[*] About to hook RLMRealmConfiguration->- setEncryptionKey: ...
[+] Detected call to "setEncryptionKey:" of the class:RLMRealmConfiguration {
        fileURL                                                                                                 =
file:///[redacted]/Library/Application%20Support/com.nordvpn.osx/default.encrypted_v2.realm;
        inMemoryIdentifier = (null);
        encryptionKey = (null);
        readOnly = 0;
        schemaVersion = 0;
        migrationBlock = (null);
        deleteRealmIfMigrationNeeded = 0;
        shouldCompactOnLaunch = (null);
        dynamic = 0;
        customSchema = (null);
}
[+] Arguments passed as parameters:
        [*] arg2 Objective C type: Foundation.__NSSwiftData
        [*] arg2: {length = 64, bytes = 0x31306138 32633034 32643062 39383330 ... 38666565 35656437 }
        [*] arg2: 10a82c042d[redacted]
```

Using the obtained key for macOS computer #1, we were able to open the realm database:

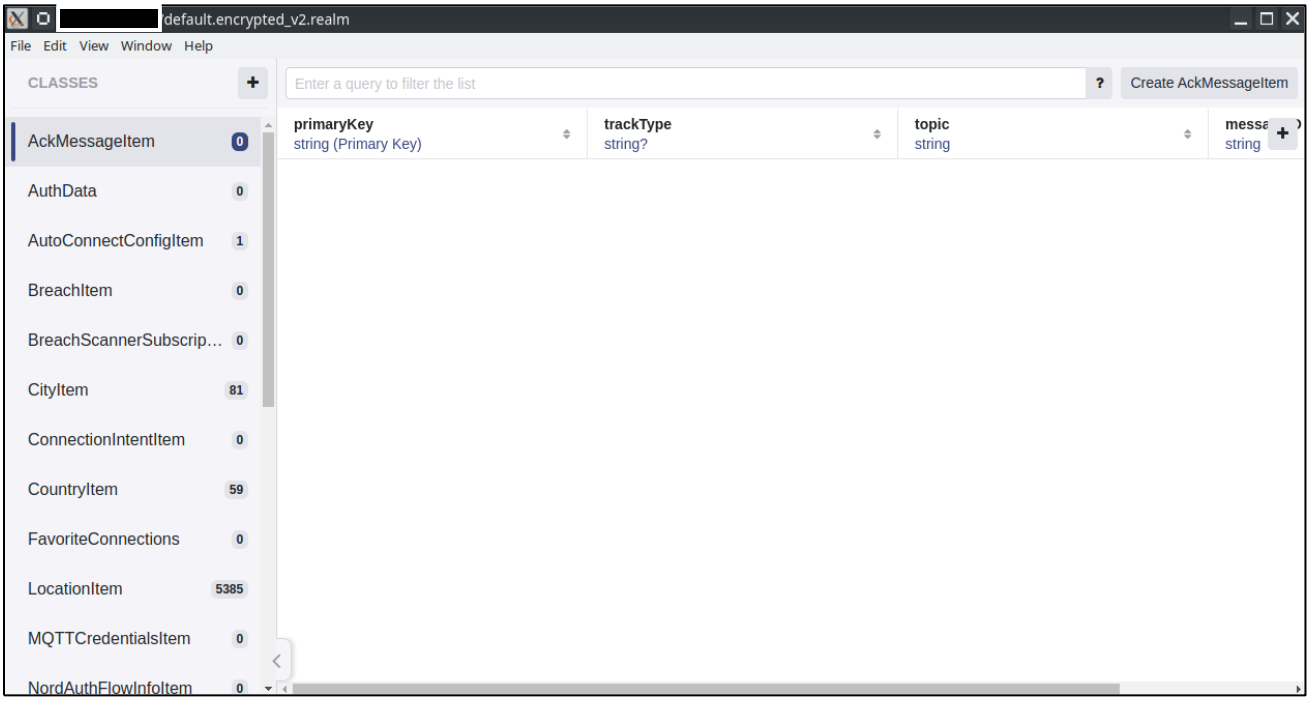


***Figure 3 – Accessing the encrypted database.***

In addition, it is worth mentioning that the encryption key is hold in memory in plaintext. In the following excerpt we show how it is possible for us to dump the memory process and then search for the key:

```
Catalina:fridump vs$ sudo python3 fridump.py "NordVPN IKE"


        ______   _     _
        |  ___|  (_)   | |
        | |_ _ __ _  __| |_   _ _ __ ___  _ __
        |  _| '__| |/ _` | | | | '_ ` _ \| '_ \
        | | | |  | | (_| | |_| | | | | | | |_) |
        \_| |_|  |_|\__,_|\__,_|_| |_| |_| .__/
                                          | |
                                          |_|

Current Directory: /[redacted]/tools/fridump
Output directory is set to: /[redacted]/tools/fridump/dump
Creating directory...
Starting Memory dump...
Progress: [##################################################] 99.42% Complete
Finished!


Catalina:fridump vs$ cd dump
Catalina:dump vs$ grep "60ebc0ee82[redacted]" *
Binary file 105553116266496_dump.data matches
^C
Catalina:dump vs$ strings 105553116266496_dump.data | grep
60ebc0ee82[redacted]
ZTUMZTUM60ebc0ee82[redacted]
60ebc0ee82[redacted]
60ebc0ee82[redacted]
60ebc0ee82[redacted]
```

*Note: In order to verify the remediation, we followed the same procedure we used to test and identifying the issue. Although we can now see that the keys used by the two different instances are different, we did not verify the source code changes. Therefore, we cannot ensure they cannot be inferred from another known data in the host machine.*

## (Remediated) Information Disclosure in Binary Files (CWE-615) – Low

**Validation Retest Notes**

During the retest we found that this vulnerability was no longer present. We performed some string searches in the binaries and did not find internal information. As an example:

*For the NordVPN_[redacted]_6.1.1 build:*

```
Catalina:/ vs$ strings
/Applications/NordVPN.app/Contents/Frameworks/NordKeychain.framework/NordKeychain | grep -i
'users\|[redacted]'
```

*For the NordVPN_[redacted]_6.1.1 build:*

```
Catalina:/ vs$ strings /Applications/NordVPN\
IKE.app/Contents/Frameworks/NordKeychain.framework/NordKeychain | grep -i 'users\|[redacted]'

Catalina:/ vs$ strings /Applications/NordVPN\
IKE.app/Contents/Frameworks/OpenVPNApple.framework/Versions/A/OpenVPNApple | grep -i
'users\|[redacted]'
compiler: /Applications/Xcode.app/Contents/Developer/usr/bin/gcc -fPIC -arch x86_64 -O3 -arch
x86_64 -isysroot
/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX11.1.sdk
-I/Users/builds/fRZHn_B1/0/low-level-hacks/openvpn/openvpn-
llh/platform/darwin/build/macos/x86_64/include -mmacosx-version-min=10.11  -
D__APPLE_USE_RFC_3542=1 -DL_ENDIAN -DOPENSSL_PIC -DOPENSSL_CPUID_OBJ -DOPENSSL_IA32_SSE2 -
DOPENSSL_BN_ASM_MONT -DOPENSSL_BN_ASM_MONT5 -DOPENSSL_BN_ASM_GF2m -DSHA1_ASM -DSHA256_ASM -
DSHA512_ASM -DKECCAK1600_ASM -DRC4_ASM -DMD5_ASM -DVPAES_ASM -DGHASH_ASM -DECP_NISTZ256_ASM -
DX25519_ASM -DPOLY1305_ASM -D_REENTRANT -DNDEBUG -O3 -arch x86_64 -isysroot
/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX11.1.sdk
-I/Users/builds/fRZHn_B1/0/low-level-hacks/openvpn/openvpn-
llh/platform/darwin/build/macos/x86_64/include -mmacosx-version-min=10.11
.cnf
openssl.cnf
/Users/builds/fRZHn_B1/0/low-level-hacks/openvpn/openvpn-
llh/platform/darwin/build/macos/x86_64/ct_log_list.cnf
OPENSSLDIR: "/Users/builds/fRZHn_B1/0/low-level-hacks/openvpn/openvpn-
llh/platform/darwin/build/macos/x86_64"
ENGINESDIR: "/Users/builds/fRZHn_B1/0/low-level-hacks/openvpn/openvpn-
llh/platform/darwin/build/macos/x86_64/lib/engines-1.1"
crypto/engine/eng_cnf.c
/Users/builds/fRZHn_B1/0/low-level-hacks/openvpn/openvpn-
llh/platform/darwin/build/macos/x86_64/lib/engines-1.1
crypto/evp/evp_cnf.c
/Users/builds/fRZHn_B1/0/low-level-hacks/openvpn/openvpn-
llh/platform/darwin/build/macos/x86_64/private
/Users/builds/fRZHn_B1/0/low-level-hacks/openvpn/openvpn-llh/platform/darwin/build/macos/x86_64
/Users/builds/fRZHn_B1/0/low-level-hacks/openvpn/openvpn-
llh/platform/darwin/build/macos/x86_64/certs
/Users/builds/fRZHn_B1/0/low-level-hacks/openvpn/openvpn-
llh/platform/darwin/build/macos/x86_64/cert.pem
ssl/ssl_mcnf.c
```