Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Pentest-Report Mullvad VPN Clients 09.2018

Cure53 & Assured AB, Dr.-Ing. M. Heiderich; J. Larsson; A. Alasjö; Dr. J. Magazinius, MSc.; D. Weißer, BSc.; J. Hector, MSc.; N. Krein, BSc.; T.-C. “Filedescriptor” Hong

Index

Introduction

This report documents the results of a security assessment targeting the newly developed Mullvad VPN client software. Carried out in September 2018, this project comprised both a penetration test and a source code audit, benefitting from a collaboration of two security teams. Specifically, two teams from the Assured AB in Gothenburg, Sweden, and from the Cure53 in Berlin, Germany, were tasked with the completion of this assessment.

The assessment took a total of eighteen days to complete and involved a total of eight testers, three from the Assured AB and five from the Cure53 teams, respectively. Under the white-box premise, the testers were granted access to latest binaries and the relevant sources via GitHub. The methodological approach was a self-explanatory choice because everything related to the tested Mullvad VPN scope is available in an open source software format. In terms of the division of tasks, the Assured AB team focused on investigating the Electron available for three operating systems. In comparison, Cure53 audited the code for several components that fuel the Electron application and glue the UIs to the OS-level interactions, communications and settings.

Throughout the project, all three involved parties, meaning the Mullvad, Assured AB and Cure53 teams, communicated using a shared Slack channel, making the coordination of the tests quite pleasant in general. All communications were quick, accurate and productive. In addition to this, Cure53 further performed live-reporting for two issues which were considered to carry greater risks. Mullvad commented on those and one problem, namely the "*Critical*"-risk documented in MUL-01-004, has been fixed in lightning speed while the tests were still ongoing, allowing for a successful fix verification The other live-reported issue, namely MUL-01-006 was conversely communicated as an expected behavior and flagged as such. The same was decided for MUL-01-002 after discussing this report with the maintainers. However, the Mullvad team announced that a discussion is ongoing to see if it is possible to add a better defense-in-depth mechanism in the realm of this finding.

In sum, the assessment yielded a total of seven issues, which an exceptionally small number given the complex field of the VPN software and the connected, vast attack surface. It needs to be noted, however, that in this instance only applications and supporting features were checked, meaning that no verdict can be made on the security posture of the server-side and backend.

In the following sections, the report will first briefly detail the scope and then moves on to a case-by-case analysis of findings, accompanying the technical backgrounds for the aforementioned two vulnerabilities and five general weaknesses with the envisioned mitigation approaches. In the final section, the Assured AB and Cure53 teams share some broader conclusions about the tested Mullvad VPN scope. In light of the findings, a summary pertaining to the security premise and properties of the Mullvad project is offered.

Scope

- **Mullvad VPN software for OSX, Linux and Windows**
  - https://github.com/mullvad/mullvadvpn-app
  - Sources were made available to Cure53.

Identified Vulnerabilities

The following sections list both vulnerabilities and implementation issues spotted during the testing period. Note that findings are listed in a chronological order rather than by their degree of severity and impact. The aforementioned severity rank is simply given in brackets following the title heading for each vulnerability. Each vulnerability is additionally given a unique identifier (e.g. *MUL-01-001*) for the purpose of facilitating any future follow-up correspondence.

MUL-01-004 Windows: Privilege escalation by replacing executables (*Critical*)

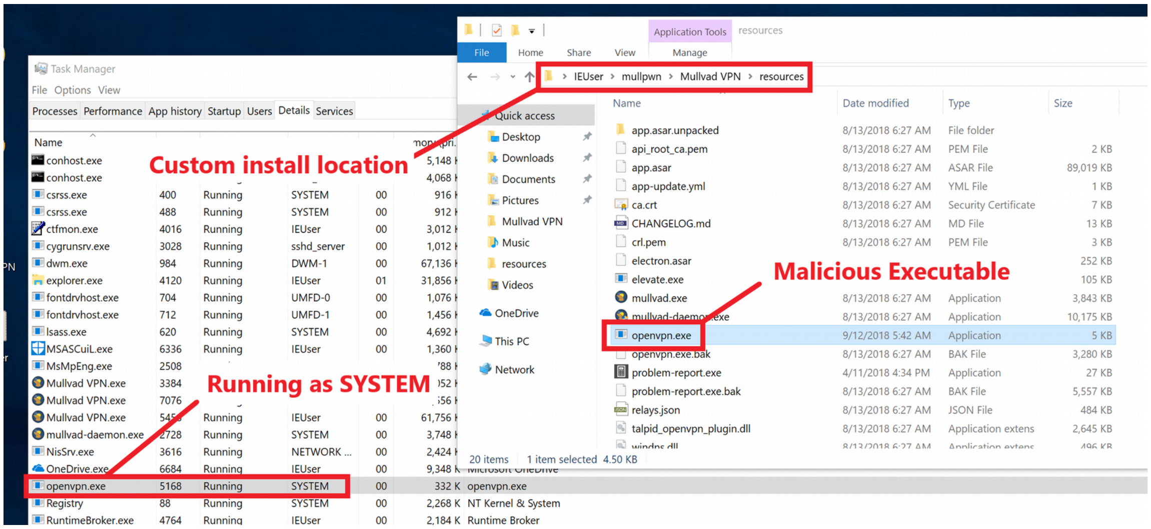
It was discovered that the Windows installer allows to install the application anywhere on the filesystem. If the application is installed in a location that is writeable by a regular user (e.g. an external hard drive in order to keep the system drive as small as possible), sensitive executables can be replaced. This, ultimately, can lead to a local privileges escalation to *SYSTEM*, which means the most powerful privileges on the Windows platform.

A central part of the application architecture is the *mullvad-daemon* service, which is responsible for establishing the actual VPN connection through the use of *openvpn.exe*. The *openvpn.exe* is supplied as part of the installation package and resides in the installation's location. Since the daemon service is running with *SYSTEM* privileges, any binary executed by it will thus be running with *SYSTEM* as well. Simply replacing the *openvpn.exe* with a malicious executable from the attacker allows to run any code with the highest privileges.

For demonstration purposes, a simple binary was compiled and used as replacement for the *openvpn.exe*. Listed below is the source of said executable. It is shown how it just prints something to *stdout* and waits for user-input (allowing for the execution to be observed in the *task manager*).

PoC executable source:
```
#include <stdio.h>
int main(int argc, char *argv[])
{
	printf("Give LPE!");
	getchar();
	return 0;
}
```

Upon replacing the executables and starting the Mullvad application, one can observe the execution of the malicious code using the *task manager*. Below is a screenshot showing the process in the *task manager*, as well as the custom install location.



Name					mory (pri.	
conhost.exe					5,148	
conhost.exe					4,068	
csrss.exe	400	Running	SYSTEM	00	916	
csrss.exe	488	Running	SYSTEM	00	912	
ctfmon.exe	4016	Running	IEUser	00	3,012	
cygrunsrv.exe	3028	Running	sshd_server	00	1,012	
dwm.exe	984	Running	DWM-1	00	67,136	
explorer.exe	4120	Running	IEUser	01	31,856	
fontdrvhost.exe	704	Running	UMFD-0	00	1,076	
fontdrvhost.exe	712	Running	UMFD-1	00	1,456	
lsass.exe	620	Running	SYSTEM	00	4,692	
MSASCuiL.exe	6336	Running	IEUser	00	1,360	
MsMpEng.exe	2508				'88	
Mullvad VPN.exe	3384				)52	
Mullvad VPN.exe	7076				)56	
Mullvad VPN.exe	545	Running	IEUser	00	61,756	
mullvad-daemon.exe	2728	Running	SYSTEM	00	3,748	
NisSrv.exe	3616	Running	NETWORK ...	00	2,424	
OneDrive.exe	6684	Running	IEUser	00	9,348 K	Microsoft OneDrive
openvpn.exe	5168	Running	SYSTEM	00	332 K	openvpn.exe
Registry	88	Running	SYSTEM	00	2,268 K	NT Kernel & System
RuntimeBroker.exe	4764	Running	IEUser	00	2,184 K	Runtime Broker

› IEUser › mullpwn › Mullvad VPN › resources

Name	Date modified	Type	Size
app.asar.unpacked	8/13/2018 6:27 AM	File folder	
api_root_ca.pem	8/13/2018 6:27 AM	PEM File	2 KB
app.asar	8/13/2018 6:27 AM	ASAR File	89,019 KB
app-update.yml	8/13/2018 6:27 AM	YML File	1 KB
ca.crt	8/13/2018 6:27 AM	Security Certificate	7 KB
CHANGELOG.md	8/13/2018 6:27 AM	MD File	13 KB
crl.pem	8/13/2018 6:27 AM	PEM File	3 KB
electron.asar			252 KB
elevate.exe			105 KB
mullvad.exe	8/13/2018 6:27 AM	Application	3,843 KB
mullvad-daemon.exe	8/13/2018 6:27 AM	Application	10,175 KB
openvpn.exe	9/12/2018 5:42 AM	Application	5 KB
openvpn.exe.bak	8/13/2018 6:27 AM	BAK File	3,280 KB
problem-report.exe	4/11/2018 4:34 PM	Application	27 KB
problem-report.exe.bak	8/13/2018 6:27 AM	BAK File	5,557 KB
relays.json	8/13/2018 6:27 AM	JSON File	484 KB
talpid_openvpn_plugin.dll	8/13/2018 6:27 AM	Application extens	2,645 KB
windns.dll	8/13/2018 6:27 AM	Application extens	496 KB

20 items 1 item selected 4.50 KB

Fig.: PoC screenshot, showing executable running as SYSTEM

A first solution that may come to mind is to check binary signatures prior to execution, however this approach is susceptible to time-of-check & time-of-use (TOC & TOU) issues. In particular, the time in between the signature check and the actual execution of the binary can potentially be abused to swap out the executables, meaning that the execution of malicious code would remain possible. Hence, it is rather recommended to ensure that all executables run by the daemon (or any other higher-privilege process) are executed from a trusted location such as *Program Files* (only writeable by *admins*) of the *root* drive.

Fix Note: This issue was live-reported and then fixed during the testing phase. Cure53 reviewed the fix and found the mitigation appropriate. The fix verification was successful.

MUL-01-006 Daemon: Any user can issue WebSocket commands (*High*)

It was noticed that Mullvad suffers from a general design flaw in which any user on a system with Mullvad installed, can issue configurational changes to its runtime. This is due to the fact that the Mullvad-daemon service opens a WebSocket interface to listen to incoming commands and act accordingly.

Despite authentication being in place, the already reported weak file permission weakness (see MUL-01-003) makes it possible for any user to authenticate with the WebSocket, which effectively means they can issue arbitrary commands as well. This includes the option to completely disable the Mullvad daemon and thus deanonymize the user who runs the app. For that, an authenticated WebSocket user can issue the following command.

Example *shutdown* command:

```
{"jsonrpc":"2.0","id":"de857110-d52b-4ae4-bd8a-
6ccdc8db75d6","method":"shutdown","params":[]}
```

Log File:

```
[2018-09-13 15:09:46.227][info] The websocket connection closed with code: 1006
[2018-09-13 15:09:46.228][debug] Lost connection to daemon: Abnormal closure
```

While the weak file permissions, mentioned in MUL-01-003, allow ANY user on the system to access the daemon, this issue still persists in case access is restricted to the intended users.

The core problem at hand is the multi-user design chosen by Mullvad, given that any user (outside of the group allowed to access Mullvad) can issue any command supported by the daemon. If one user from the group has evil intentions, it is possible to, for example, shutdown the VPN service. This would, of course, deanonymize the other users in the group. Thus, it is recommended to redesign the Mullvad-Daemon user-model and implement a form of privilege segregation which restricts management endpoints to either a certain user or a certain user-group.

Note: This issue was flagged to be a false alert and the criticized behavior was announced to be expected and desired by design. This is because malicious local users are not part of the threat model.

Miscellaneous Issues

This section covers those noteworthy findings that did not lead to an exploit but might aid an attacker in achieving their malicious goals in the future. Most of these results are vulnerable code snippets that did not provide an easy way to be called. Conclusively, while a vulnerability is present, an exploit might not always be possible.

MUL-01-001 App: Missing *BrowserWindow* preferences allow RCE *(Info)*

It was found that the client application does not enforce the necessary separation between the *BrowserWindow* component and *Node*. This exposes critical *Node* API functions which would allow an attacker to leverage an XSS vulnerability to execute arbitrary commands on the system. To enforce proper separation between the *BrowserWindow* and the *Node*, two options need to be set in the *webPreferences* object when the *BrowserWindow* is instantiated.

The *nodeIntegration* option, by default set to *true*, exposes the Node API to the rendered app. Conversely, this option needs to be set to *false* to protect the system. The *contextIsolation* option, by default set to *false*, separates the context rendered app from the context of *preload* scripts and Electron's internal code. If *contextIsolation* is not enforced, the rendered app can coerce scripts running in the Electron context to execute arbitrary commands.

Recommendation for the updated preferences:

```
const options = {
  width: 320,
  minWidth: 320,
  height: contentHeight,
  minHeight: contentHeight,
  resizable: false,
  maximizable: false,
  fullscreenable: false,
  show: false,
  frame: false,
  webPreferences: {
    // prevents renderer process code from not running when window is hidden
    backgroundThrottling: false,
    // Enable experimental features
    blinkFeatures: 'CSSBackdropFilter',
    // Disable Node API in renderer
    nodeIntegration: false,
    // Enforce isolation between application and underlying electron code.
    contextIsolation: true
  },
};
```

While no XSS vulnerabilities were found in the application code itself, a general issue in Electron exists and pertains to a situation where the *BrowserWindow* navigates to a file which is dragged and dropped onto the application's window. This file inherits the access rights of the component, allowing any contained script's code to execute arbitrary system commands.

It is recommended to set the flags to a more secure state than the default and avoid attacks that lead to XSS to taking place. It is known that Electron is a hard-to-tame horse in terms of security and its security guidelines[1] should always be studied closely.

PoC HTML file:

```
<html>
  <head>
    <script>
    require('child_process').exec('/usr/bin/gnome-calculator');
    window.history.go(-1);
    </script>
  </head>
  <body></body>
</html>
```

MUL-01-002 App: WebSocket leaks real IP addresses and geolocation *(Medium)*

It was discovered that the *get_current_location* RPC call over WebSocket contains the geolocation and IP address of the system running the Mullvad daemon. If the WebSocket URI and authentication key is known to an attacker, the real IP address may be leaked via a rogue WebSocket client. This can happen via a web or a locally installed software as the VPN can be disconnected via an RPC call (see MUL-01-006).

Example RPC call and corresponding result:

```
{"jsonrpc":"2.0","id":"5b522711-52d8-4ced-98bd-
a445faa531c7","method":"get_current_location","params":[]}
{"jsonrpc":"2.0","result":
{"city":"Gothenburg","country":"Sweden","ip":"XXX.XXX.XXX.XXX","latitude":57.716
7,"longitude":11.9667,"mullvad_exit_ip":false},"id":"5b522711-52d8-4ced-98bd-
a445faa531c7"}
```

---

[1] https://github.com/electron/electron/blob/master/docs/tutorial/security.md

It is recommended not to expose sensitive RPC calls over WebSocket.

Note: This issue was flagged to be a false alert and technically a subset of what is described in MUL-01-006. The criticized behavior was announced to be expected and desired by design. This is because malicious local users are not part of the threat model.

MUL-01-003 Daemon: Weak permissions on *config* and *log* files *(Low)*

The Mullvad daemon and application were revealed to create *configuration* and *log* files with overly permissive *access* flags. In essence, any user with access to the machine where Mullvad is installed will be able to access the files' contents. In a scenario where the *host* runs an additional, vulnerable application that allows to leak files, this issue might even let an attacker elevate privileges. The following listing shows the permission flags of the affected files.

Mullvad's *configuration* and *log* files

```
# ls -al /etc/mullvad-vpn/
-rw-r--r--   1 root root   341 Sep 11 12:15 settings.json
# ls -al /var/log/mullvad-vpn/
-rw-r--r--  1 root root   3661923 Sep 11 12:15 daemon.log
-rw-r--r--  1 root root      6413 Sep 11 12:15 openvpn.log
-rw-r--r--  1 root root      6424 Sep 11 12:08 openvpn.old.log
# ls -la /tmp/.mullvad_rpc_address
-rw-r--r-- 1 root root 58 Sep 13 15:09 /tmp/.mullvad_rpc_address
```

In order to make sure that no potential attacker can read the files in question, it is recommended to make them only accessible for the *root* user. This can be achieved by setting the file's *access* flags to *600*[2].

MUL-01-005 OOS: CSRF on adding and removing forwarded ports *(Low)*

A CSRF vulnerability was spotted on the Mullvad website and allows an attacker to add and remove a forwarded port for a user. While this is technically out-of-scope for this assessment, it was decided to report this finding nevertheless for the sake of completeness.

Affected Endpoints:

- https://mullvad.net/en/account/add_port/
- https://mullvad.net/en/account/remove_port/

Steps to Reproduce:

1. Make sure to be logged in.
2. Navigate to *data:text/html,<img src=https://mullvad.net/en/account/add_port/>*
3. Notice that a forwarded port is added.

A PoC for forwarded removal can be done in a similar fashion. It is recommended to implement a proper CSRF protection that utilizes an unguessable nonce as a CSRF token and validates it against the affected endpoints.

---

[2] http://www.filepermissions.com/file-permission/600

MUL-01-007 App: Lax version requirements for *Node* dependencies (*Info*)

It was found that the *Node* applications have dependencies which are defined as "version compatible with X.X.X" in *package.json* files. This is done with the caret notation specified by *node-semver*[3]. This implies that updates of any dependency might contain malicious code or otherwise bad behavior, even though the *yarn.lock* file contains integrity checks for the installed modules.

An example dependency in the Desktop application's *package.json* is *"history": "^4.6.1"*, where any version of the history module compatible with 4.6.1 would be installed. If accepted version numbers are too loosely defined, any verification of dependencies becomes impractical. Mullvad should choose a stricter approach to define dependency versions to mitigate this issue, as well as only install modules from trusted developers.

Conclusions

After examining the scope of the Mullvad VPN client, the combined team comprising testers from the Assured AB and Cure53 teams only managed to find seven security-relevant issues in the application and its code. Pairing the positive indicator of very few findings with the fact that only one had been deemed to have "*Critical*" impact and was fixed before the test even finished, allows to draw a conclusion about a positive outcome of this particular project. In addition, it was great to see how quickly and elegantly the live-reported issue was fixed by the maintainers, again boding well for the security-affine development of the Mullvad VPN project.

Moving on to the specifics, it needs to be noted that the examined source code is of an overall high-quality and exhibits great readability. The project files are managed well and built-in scripts allow for efficient auditing. A major role in this rather secure implementation is played by the Rust language, which by itself offers limited attack surface and mitigates most of the low-level issues that similar software tends to suffer from. Aside from the privilege escalation issue described in MUL-01-004, the bindings to third-party software, such as *OpenVPN,* can be considered correctly handled. No forms of CLI injections seem to be possible in the investigated scope.

Next, it is clear the frontend applications were developed with cross-platform compatibility in mind, which makes auditing simpler but also requires the assumption that the underlying framework is secure. The core problem is that the underpinning is done with Electron, which is by now known to be prone to various vulnerabilities and poses major challenges. Therefore, security work in this realm should not be stopped. To protect the application and the daemon, it is recommended to investigate and follow best practices for the framework configuration, especially since missing configuration preferences may introduce security issues. This is evident for the Electron applications, as MUL-01-001 illustrates.

---

[3] https://github.com/npm/node-semver

Since the application is depending on third-party modules, it is recommended to establish a verification procedure for external dependencies, along with the currently used integrity checks. Notably, the WebSocket communication was replaced with IPC in a release after the audit started. It was verified to have mitigated several issues regarding the RPC calls.

As an exception to the above, a negative impression has been gained from some issues summed up in MUL-01-006. This shows that some questionable, general design decisions should be seen as concerning. Discussions between the developers have nevertheless demonstrated that the current approach of handling multiple users is the desired one, even though it is expected that another debate about handling this realm in safer ways is supposed to be internally held at Mullvad soon. In connection to this, Cure53 strongly believes that a stricter permission model for a multi-user setup is required in order to fully mitigate the presented issue.

In sum, despite this one concern, Cure53 and Assured AB are happy with the results of the audit and the software leaves an overall positive impression. With security dedication of the in-house team at the Mullvad VPN compound, the testers have no doubts about the project being on the right track from a security standpoint.

Cure53 and Assured AB would like to thank Fredrik Strömberg and Linus Färnstrand from the Mullvad team for their excellent project coordination, support and assistance, both before and during this assignment.