Fine penetration tests for fine websites

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

Pentest-Report Mullvad VPN & Servers 11.-12.2020

Cure53, Dr.-Ing. M. Heiderich, J. Larsson, M. Kinugawa, MSc. R. Peraglie, BSc. B. Walny

Index

Introduction

"Mullvad VPN AB is owned by parent company Amagicom AB. The name Amagicom is derived from the Sumerian word ama-gi – the oldest word for "freedom" or, literally, "back to mother" in the context of slavery – and the abbreviation for communication. Amagicom stands for "free communication"."

From https://mullvad.net/en/what-is-privacy/

This document is dedicated to a presentation of a security-centered project carried out by Cure53 for Mullvad. More specifically, the report describes the results of a thorough and comprehensive penetration test and source code audit against the Mullvad VPN servers, infrastructure and related web applications and other exposed services. The project was completed in late 2020.

It needs to be clarified that this testing exercise marks the third time Cure53 assessed the Mullvad software, with *MUL-01* back in 2018 initiating the cooperation and focusing on the client software. Then, *MUL-02* covered API, client software and mobile apps in May 2020. The reported investigation, which took place in November and December 2020, is labeled as *MUL-03* accordingly. For the first time, servers and infrastructure were placed in the primary scope.

As for the resources, five members of the Cure53 team were involved in this project. They participated in preparing, executing and finalizing the tests and audits. The team spent a total of twenty-two days on the scope. The work was split into two different work packages for better structuring. These were:

- **WP1**: Mullvad Websites, Web Applications & API Backend
- **WP2**: Mullvad Infrastructure, Servers, VPN Bridge & Bastion Host

Following best practices established in previous cooperation, Cure53 relied on white-box methods. Access to sources, configuration files, servers and everything else that was required for good coverage levels was provided. Further shared with Cure53 were test-users with relevant credentials, as well as other test-supporting material.

The project started on time and progressed efficiently. Communications were done using the usual, shared Slack channel which enabled the two workspaces of Mullvad and Cure53 to be glued together. Everyone involved in the auditing and testing could take part in the discussions. Communications were frequent and excellent, as was the test preparations done by the Mullvad team. The communications during the test helped with optimizing coverage, while Cure53 was also able to give regular status updates about the emerging findings. As a consequence, the Mullvad team had a chance to promptly address some of the issues.

In terms of findings, the Cure53 team managed to spot and document a total of twelve security-relevant issues. Six were classified to be security vulnerabilities of varying severity levels and six represent general weaknesses marked by lower exploitation potential. Two findings were given *High* severity ratings and the rest were ranked *Medium* and lower, with some items just informational in nature. None of the issues received *Critical* scores, which is a good sign for Mullvad's server-side security posture. One of the high severity issues describes a VPN DoS in case OpenVPN is used, the other is a possible Docker container breakout that might be used for privilege escalation purposes.

In the following sections, the report will first shed light on the scope and key test parameters, including the structure of WPs. Next, all findings will be discussed in the specifically grouped vulnerability and miscellaneous categories, then following a chronological order within issue-types. Alongside technical descriptions, PoC and mitigation advice are supplied when applicable, so as to help the development team in promptly addressing the issues. Finally, the report will close with broader conclusions about this 2020 project. Cure53 elaborates on the general impressions and offers a verdict based on the testing team's observations and collected evidence. Tailored hardening recommendations for the Mullvad VPN complex are also incorporated into the final section.

Scope

- **Penetration-Tests & Source Code Audits against Mullvad VPN Software & Servers**
  - **WP1**: Mullvad Websites, Web Applications & API Backend
    - The sources for the API and web were shared with Cure53
    - **Mullvad Web Application**
      - https://mullvad.net
    - **Mullvad Web API**
      - https://problemreports.mullvad.net
      - https://ipn.mullvad.net
      - https://api.mullvad.net
  - **WP2**: Mullvad Infrastructure, Servers, VPN Bridge & Bastion Host
    - **WireGuard:**
      - se99-wireguard.mullvad.net (185.195.233.75)
    - **OpenVPN:**
      - se-sto-999.mullvad.net (185.195.233.76)
    - **Bridge:**
      - se-sto-br-999.mullvad.net (185.195.233.77)
    - **Bastion:**
      - se-mma-bst-002.mullvad.net (193.138.218.139, 10.1.0.1)
  - **Mullvad IDs used for testing**
    - tester1=8440405519083188
    - tester2=2051846157067350
    - tester3=6601574417833605
    - tester4=1164334873125326
    - tester5=4026893288017628
    - tester6=5218111127694357
  - **Sources were shared with Cure53**
  - **Test-supporting material was shared with Cure53**

Identified Vulnerabilities

The following sections list both vulnerabilities and implementation issues spotted during the testing period. Note that findings are listed in 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-03-001*) for the purpose of facilitating any future follow-up correspondence.

MUL-03-002 WP2: OpenVPN user-authentication can be bypassed *(Medium)*

It was found that attackers could make use of Mullvad's OpenVPN service without paying for it. The OpenVPN plugin used to handle the user authentication for OpenVPN allows arbitrary users to connect to the service as long as the Mullvad API is either unavailable by connection timeout or throws an erroneous HTTP status code (usually 4xx-5xx). However, a 429 HTTP error can always be induced by attackers who simply exceed the rate-limit for the OpenVPN server responsible for sending the requests to the API.

Affected File:
/home/mad/vpnserver/vpnauth_auth.py

Affected Code:
```
try:
       api = vpnauth_utils.MullvadAPI(common_name, username)
except vpnauth_utils.InvalidCID as e:
       [...]
       report_deny('invalid_cid')
       sys.exit(1)

try:
       client_auth_result = api.client_auth()
except requests.HTTPError as e:
       log.error('HTTP error, allowing: %s', e)
       vpnauth_utils.report_http_error('client_auth', e.response.status_code)
       report_allow('client_auth_http_error')
       sys.exit(0) # Allow without port forward, may be accounts bruteforce
except (ValueError, KeyError, requests.ConnectionError,
       requests.Timeout) as e:
       log.error('Master connection failed, allowing: %s', e)
       vpnauth_utils.report_error('client_auth')
       report_allow('client_auth_error')
       Sys.exit(0) # Allow any client if mullvad-api is down
```

Steps to reproduce:

1. Update a Mullvad OpenVPN configuration file and only allow se-*sto-999* as *remote*
2. Update the *mullvad_userpass.txt* file to hold a non-valid username like 1111
3. Run the following command in terminal:
4. `for i in `seq 1 21`; do sudo openvpn mullvad_se_sto.conf | tee /tmp/openvpn"$i".log & done`
5. If successful, at least a single remote connection will be established for a short time (~1-5 minutes).

It is recommended for access to OpenVPN to be forbidden in case of HTTP errors. However, at the same time, the rate-limit for the *internal/openvpn-client-auth/* endpoint should be removed. By doing so, the risk for a Denial-of-Service vulnerability is mitigated to the API while still granting service availability for the Mullvad's OpenVPN users.

MUL-03-005 WP2: OpenVPN users can be disconnected by attackers *(High)*

It was found that attackers can close the connection of other OpenVPN peers who are using the same entry gate. An OpenVPN *monitor.py* script frequently connects to the OpenVPN management interface of an OpenVPN-configured server instance. The script intends to fetch all connected OpenVPN users and check through the API if any of these accounts are expired. Finally, expired accounts are disconnected via the *client-kill* command issued through the OpenVPN management interface.

In order to retrieve all currently connected clients, the *monitor.py* script issues the *status 2* command and parses the output line-by-line. Every line that contains a single connected client is parsed by splitting the line at all commas and parsing the individual fields with Python variables. However, the script does not properly handle commas that are encapsulated within the *username* or *common_name* variable returned by the management interface.

This increases the risk of attackers successfully authenticating with a malicious expired *username* containing a comma *(,)* either with MUL-02-001 or by changing the username on a re-negotiation. That allows controlling the *client_id* parameter that will be used in the following *client-kill* command. Since the *client_id* is a sequential number, this could be abused to disconnect clients from the Mullvad OpenVPN server. At the same time, the *monitor.py* script will fail to disconnect the client on account-expiry.

Affected File:
/home/mad/monitor/monitor.py

Affected Code:

```
def _parse(self, status_line):
columns = status_line.split(',')
common_name = columns[1]
virtual_address = columns[3]
virtual_ipv6_address = columns[4]
username = columns[9]
client_id = columns[10]
[...]
self.client_id = client_id
```

Steps to reproduce:

1. Authenticate with the username *abcdef,1234* with MUL-03-002
2. After a while, the targeted connection to the client with the client-id *1234* is killed.

Since the OpenVPN management interface performs insufficient output sanitization on the *common_name* and the *username,* these variables have to be controlled before a connection can be successfully established. Therefore, it is advisable the OpenVPN plug-in deployed by Mullvad checks and verifies these variables before allowing the connection by applying an allow-list of alphanumeric characters only.

MUL-03-007 WP2: Socks proxy allows connection to Telegraf StatsD *(Low)*

It was found that the Socks proxy allows to establish a connection to all services that run on the local interface, including the StatsD Telegraf service. This induces the risk of attackers maliciously connecting via the StatsD protocol and polluting malicious data or triggering unauthorized events.

The configuration file of the Socks proxy should define strict rules that only allow a connection using the Internet interface. By doing so, potentially dangerous services listening on the local loopback interface are protected from remote abuse.

MUL-03-008 WP2: Race condition potentially allows port forwarding *(Low)*

It was found that the OpenVPN port forwarding suffers from a race condition vulnerability. Port forwarding is performed with a rule-set deployed with *iptables* within the *vpnauth_portforward.py* script. The script will only be executed at the start of the connection removing port forwarding rules of other users and adding as users own rules for a given OpenVPN IP. However, this script is executed detached in a separate thread and returns immediate success to OpenVPN giving attackers the slim chance of connecting to a user’s port when the rules are yet not removed via *iptables*.

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

Affected File:
vpnauth-master/src/openvpn.rs

Affected Code:
```
fn handle_client_connect(handle: &Handle, env: HashMap<String, String>) {
        scriptrunner::exec_detached(&handle.connect_script[..], env, None);
}
```

It is recommended that the port forwarding rules are not detached and handled in a separate thread. As a result, OpenVPN will stop completing the connection before the port forwarding rules of the previous user are removed.

MUL-03-009 WP1: DOM-based XSS via outdated Swagger UI (*Medium*)

It was found that an XSS vulnerability exists in the used Swagger UI[1]. Swagger UI loads the URL specified in the *url* parameter and uses the included contents to render the HTML. The HTML written there is sanitized with the DOMPurify library[2] and then rendered. However, since the DOMPurify library is not deployed in its latest version, XSS can occur through a known bypass[3]. The problem can be reproduced via the following URLs.

PoCs:

- https://api.mullvad.net/app/documentation/?url=data:;base64,b3BlbmFwaTogIjMuMC4wIgoKaW5mbzoKICB0aXRsZTogWFNTIHZpYSBET01QdXJpZnkgQnlwYXNzCiAgZGVzY3JpcHRpb246IAogICAgPGg0PlRFU1Q8L2g0PgogICAgPGZvcm0%2BPG1hdGg%2BPG10ZXh0PjwvZm9ybT48Zm9ybT48bWdseXBoPjxzdmc%2BPG10ZXh0PjxzdHlsZT48cGF0aCBpZD0iPC9zdHlsZT48aW1nIG9uZXJyb3I9YWxlcnQoZG9jdW1lbnQuZG9tYWluKSBzcmM%2BIj4%3D
- https://api.mullvad.net/public/documentation/?url=data:;base64,b3BlbmFwaTogIjMuMC4wIgoKaW5mbzoKICB0aXRsZTogWFNTIHZpYSBET01QdXJpZnkgQnlwYXNzCiAgZGVzY3JpcHRpb246IAogICAgPGg0PlRFU1Q8L2g0PgogICAgPGZvcm0%2BPG1hdGg%2BPG10ZXh0PjwvZm9ybT48Zm9ybT48bWdseXBoPjxzdmc%2BPG10ZXh0PjxzdHlsZT48cGF0aCBpZD0iPC9zdHlsZT48aW1nIG9uZXJyb3I9YWxlcnQoZG9jdW1lbnQuZG9tYWluKSBzcmM%2BIj4%3D

Since the affected page is hosted on the *api.mullvad.net* domain, which is different from the main application's domain, the impact is limited. Consequently, the severity of the

[1] https://swagger.io/tools/swagger-ui/
[2] https://github.com/cure53/DOMPurify
[3] https://vovohelo.medium.com/from-svg-and-back-yet-another-mutation-xss-via-na...9ae8b1878f

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

flaw has been set to *Medium.* It is recommended to update Swagger UI to the latest version.

Fix Note: *This issue was confirmed as fixed and the fix was verified by Cure53.*

MUL-03-011 WP2: Publicly exposed Promtail service *(Info)*

It was pointed out to Cure53 before this assessment that Mullvad discovered and mitigated a misconfiguration issue attached to their Promtail service. This issue was confirmed to be present on the test-servers provisioned to Cure53. The Promtail service used by Mullvad to aggregate metric data was configured to be publicly reachable through the test services IPv6 interfaces. This could lead to potential service abuse or act as a potential attack vector for an external attacker.

Affected service:
```
[...]
promtail 9709 promtail    7u     IPv6             107335       0t0      TCP *:9080
(LISTEN)
promtail 9709 promtail    8u     IPv6             107336       0t0      TCP *:36977
[...]

tcp6       0      0 :::36977                :::*                    LISTEN
9709/promtail
tcp6       0      0 :::9080                 :::*                    LISTEN
9709/promtail
-
```

In order to ensure transparency of the discovered issue pointed out and addressed by Mullvad, this ticket has been included in the final report.

Fix Note: *This issue was pointed out to Cure53 at the start of the test and a patch for the production servers was rolled out concurrently to notifying Cure53.*

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

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-03-001 WP1: Timing-unsafe comparison used in authentication *(Low)*

It was found that the web application uses the SQL timing-unsafe comparison operator to query the account identified by the *account* token from the database. This induces a linear relationship between the runtime of the query and the equivalent prefix-length of the queried account-token, which is compared with the *account* tokens of the stored data.

Affected File:
mullvad-api-master/core/api/authentication.py

Affected Code:
```
class AccountTokenAuthentication(authentication.BaseAuthentication):
      def authenticate(self, request):
      try:
            auth_data = request.META['HTTP_AUTHORIZATION'].split()
            if auth_data[0].lower() != 'token':
            raise Exception('The first word in the Authization
                  header should be: Token')
            account_token = auth_data[1]
      except Exception:
            raise exceptions.AuthenticationFailed(code='INVALID_AUTH_HEADER')
      try:
            account = Account.objects.get(token=account_token, kind='MULLVAD')
      except Account.DoesNotExist:
            raise exceptions.AuthenticationFailed(code='INVALID_ACCOUNT')
      request.account = account
      return account, None

      def authenticate_header(self, request):
      return 'Token'
```

It is recommended that the account is queried by looking for a cryptographical HMAC of the *account* token. The time-unsafe comparison operators of the database will only leak the HMAC of the *account* token, therefore rendering an extraction of the *account* token infeasible.

MUL-03-003 WP2: Log contents are not completely cleared *(Info)*

It was found that the *logging_clean_logs.sh* script on the servers does not fully clear all logs that were generated within a one-week period. The script removes only the current **.log* files, but leaves all *gzipped* log files available on the system. If a lot of content is written to the log file within a week, the system splits the file into the multiple files with the **.log* and **.gz* extensions. No personally identifiable information was found within those logs, making this issue purely informational.

Affected File:
/home/mad/logging_clean_logs.sh

Affected Code:
```
/bin/rm -rf /var/log/syslog
[...]
/bin/rm -rf /var/log/auth.log
[...]
/bin/rm -rf /var/log/kern.log
```

It is recommended to have the script remove all log files and their *gzipped* associates. As a result, potentially identifiable information will not be stored longer than a week on the filesystem of single nodes.

MUL-03-004 WP2: Additional hardening for container runtime *(Info)*

An analysis of the configuration attached to Docker containers used by Mullvad revealed that the production containers' configuration should be regarded as overly permissive. If an attacker would be able to gain an initial foothold inside any of the containers, it would be trivial to pivot from the contained runtime and break out to the host operating system.

The current implementation only leverages a small subset of performance and security enhancing abilities that can be deployed through a container ecosystem. The concerns raised by Cure53 during this assessment revolve around the added complexity of the current configuration, which ultimately renders traceability and auditability a complex task. This will inevitably lead to misconfiguration issues.

Going forward, it is recommended to clearly define how the container ecosystem will benefit the current infrastructure, both in terms of enabling security-enhancing features, and from a performance perspective. If neither of these features leads to any architectural benefits, Cure53 would recommend to abandon the container ecosystem and further develop the 'least-privileged' design already applied to the server nodes analyzed during this assessment.

MUL-03-006 WP2: *Shadowsocks-libev* outdated and runs as *root* (*Low*)

It was found that the *shadowsocks-libev* library version 3.1.3 is outdated and vulnerable to multiple known attack-vectors that led to the decryption of the encrypted traffic. This induces the risk of VPN-usage-detection by censors, next to a publicly known static key. At the same time, one of the *ss-server* instances runs as *root,* therefore exposing an unnecessarily elevated risk.

It is recommended that the *ss-server* runs as a standard user on a port that requires no *root* privileges. Port-forwarding could be deployed via *iptables* to route all traffic destined to the *root* port(443) to the non-*root* port of the *ss-server*. Additionally, it should be considered to use a different Socks proxy that deploys user-authentication instead of a static key in order to circumvent potential censorship.

Note: *This issue is not applicable to VPN traffic that passes through. The integrity of these services is protected by the use of the encryption schemes offered by OpenVPN and WireGuard.*

MUL-03-010 WP2: Insecure Docker configuration leads to breakout (*High*)

It was found that the current Docker configuration, which is used to host the metric and Socks features used by Mullvad, offers neither meaningful compartmentalization nor separation from the host operating system. The current Docker deployment consists of four containers that provide services for both Mullvad users and the backend features used by Mullvad. Those handle session management and metrics collection. If any of these containers were to be breached, the overall integrity of the host machine should be considered as compromised.

Configuration attached to the Socks service(s):

```
Current capabilities:
cap_chown,cap_dac_override,cap_fowner,cap_fsetid,cap_kill,cap_setgid,cap_setuid,
cap_setpcap,cap_net_bind_service,cap_net_raw,cap_sys_chroot,cap_mknod,cap_audit_
write,cap_setfcap+i

Hostfs mounted:
"Mounts": [
            {
                "Type": "bind",
                "Source": "/etc/socks_multihop.conf",
                "Destination": "/etc/socksd.conf",
                "Mode": "rw",
                "RW": true,
                "Propagation": "rprivate"
            }
```

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

Configuration attached to *openvpnmonitor:*

```
Current capabilities:
cap_chown,cap_dac_override,cap_fowner,cap_fsetid,cap_kill,cap_setgid,cap_setuid,
cap_setpcap,cap_net_bind_service,cap_net_admin,cap_net_raw,cap_sys_chroot,cap_mk
nod,cap_audit_write,cap_setfcap+eip

Command triggered:
Command executed as root on the host OS, container user has read and write
access to start.sh.
"Cmd": [
       "/bin/sh",
       "-c",
       "/start.sh"

Hostfs mounted:
"Mounts": [
          {
              "Type": "bind",
              "Source": "/home/mad/openvpn_monitor_crypto",
              "Destination": "/etc/openvpn/crypto",
              "Mode": "rw",
              "RW": true,
              "Propagation": "rprivate"
          },
          {
              "Type": "bind",
              "Source": "/home/mad/openvpn_monitor",
              "Destination": "/etc/openvpn",
              "Mode": "rw",
              "RW": true,
              "Propagation": "rprivate"
          }
```

Configuration attached to Telegraf:

```
Current Capabilities:
cap_chown,cap_dac_override,cap_fowner,cap_fsetid,cap_kill,cap_setgid,cap_setuid,
cap_setpcap,cap_net_bind_service,cap_net_admin,cap_net_raw,cap_sys_chroot,cap_sy
s_ptrace,cap_mknod,cap_audit_write,cap_setfcap+eip

Command triggered:
Command executed as root on the host OS, container user has read and write
access to start.sh.
"Cmd": [
       "/bin/sh",
       "-c",
	"/start.sh"

Hostfs mounted:
Container user have read and write access to hostfs.
/hostfs/home/mad
	Cure53.sh
	Blocklist-service.conf
	Ipset.conf
	Logging_clean_logs.sh
	openvpn_monitor_crypto
	Promtail
	Telegraf
	Txqueue_checks
	wireguard-manager.conf
	Ipmi_monitoring
	Localhost:8125
	Openvpn_monitor
	Packages
	Secrets
	wireguard
```

The overall Docker configuration offers neither additional separation nor further security boundaries for the host environment. The configuration concerns addresses should be further investigated by Mullvad, in order to determine if the use of docker as a technology stack offers any meaningful performance or security enhancing features for the overall integrity of the hosts. If this results in a decision to further use docker an overall hardening project should be initiated in order to minimize the attack surface of the current configuration.

MUL-03-012 WP1: Unrestricted PayPal webhook could lead to DoS (*Info*)

While auditing the backend for potential personal information leaks or logical flaws in the payment process, an endpoint which receives payment notifications by PayPal was found. This endpoint first of all verifies the incoming request by issuing a request to the PayPal API, which in turn confirms or denies the legitimacy of the request. Due to missing enforcement of access control, a malicious user could spam requests to this endpoint, which then triggers the same number of requests to the PayPal API.

This could lead to a block of the Mullvad web services. As a result, one could expect Denial-of-Service for legitimate customers seeking to purchase Mullvad services via PayPal. The relevant file and code parts, which are missing the enforcement of an allow-list, are shown below. The relevant parts of the request issued to PayPal have been highlighted.

Affected File:
pentest-mullvad/mullvad-api-master/payments/ipn.py

Affected Code:
```
def handle_request(request):
       statsd.increment('paypal.ipn_requested')

       if not verify_ipn_with_paypal(request):
       statsd.increment('paypal.ipn_error', tags=['reason:verification_failed'])
       return False

       data = request.POST
       [...]
```

Before undertaking any further action, it is recommended to ensure that the request was issued from services *a priori* known to be PayPal.

Fix Note: *This issue was fixed by Mullvad by implementing IP allow-listing for the PayPal IPN endpoint.*

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

Conclusions

The results of this 2020 project demonstrate that the Mullvad VPN complex is capable of fending off the most severe or *Critical* threats. At the same time, the involved team of five Cure53 testers has also proven, after twenty-two days on the scope in November and December, that the Mullvad complex suffers from certain flaws. In fact, twelve security issues, including two marked as *High,* have been highlighted in this report.

From a broad perspective, the overall attack surface of all assets in scope should be regarded as narrow and well-protected. Same goes for the security awareness and overall posture. As expected, Cure53 was unable to discover any leaks of Personally identifiable information (PII) attached to the Mullvad's end-users. However, the majority of findings were discovered during WP2, which centered on vulnerabilities and miscellaneous security-related items that reside within the infrastructure used by Mullvad. This part of the Mullvad compound has not been audited by Cure53 during any of the previous assessments, which perhaps explains the plethora of findings in this realm.

Cure53 raised concerns regarding defense-in-depth and topology design of the container ecosystem deployed throughout the Mullvad compound. The broad defense concepts adopted by Mullvad could be further improved in order to offer additional protection against post-exploitation vectors, especially local privilege escalation and potential lateral movement. In that sense, while the general security premise is solid, the assumed breach and defense-in-depth approaches warrant more work.

In regard to WP1, the Mullvad Websites, Web Applications & API Backend left the Cure53 team with a rather good impression, as evident by the overall low number of documented flaws. The attack surface for both front and backend is kept very slim, while the applications are based on the Django framework. The configuration and usage of features pertaining to the latter has been undertaken in a secure manner.

Starting from the HTTP server configuration, all relevant security headers, session parameters and cookie flags are being configured correctly. Then, security relevant Django-middleware has been configured and correctly implemented in the code, for instance with the CSRF protection through annotating relevant templates. For storage, the Django internal object storage is being used, which does not allow for SQL injection-like attacks. Other, potentially dangerous functions, such as providing *shell* access, have been avoided.

A miscellaneous issue documented in MUL-03-012 could be attributed to an oversight after refactoring, since the implementations for handling other payment service providers include an allow-list for IP addresses and offer good mitigation. Taking all the previous points into consideration, the overall state in terms of security of the Mullvad web applications and API is very good.

The frontend was checked by auditing the relevant JavaScript code. *Vue.js* framework is used in a safe way and no issues were found. The JavaScript libraries used were also examined and here an outdated Swagger UI library vulnerable to XSS was found (see MUL-03-009). The updates for the used libraries should be always checked and deployed quickly, as the effects of delays are illustrated by the aforementioned issue.

The majority of flaws during this audit were spotted within the OpenVPN server. Several design decisions that aim to maintain the availability of the Mullvad OpenVPN service (e.g. MUL-03-002) open up a window for attackers who seek to disconnect users from a targeted gateway (see MUL-03-005). At the same time, this allows using Mullvad's services without paying, introducing the risk of financial loss. A few doubts concern the value of the Shadow Socket proxies that aim to circumvent censorship. Despite MUL-03-006, a static and publicly known key could be insufficient to protect against state-funded efforts.

Besides the flaws in the newly examined area, the Mullvad service continues to be strong from a security perspective. It generally exposes a security-aware concept, even though issues concerning infrastructure should be tackled as soon as possible. Summarizing this late 2020 security assessment by Cure53, the overall results should be regarded as positive. PII and privacy leaks were not spotted on the scope, albeit being identified by Mullvad as the primary and main concern. The attack surface offered by the published services should be judged as successfully minimized. It is recommended for Mullvad to consider the advice linked to defense-in-depth concepts in the realm of user-exposed services.

Cure53 would like to thank Richard, Joshua and Simon from the Mullvad team for their excellent project coordination, support and assistance, both before and during this assignment.