Penetration Test Report

Mullvad VPN AB

V 1.1
Amsterdam, August 7th, 2023
Public

Document Properties

Client	Mullvad VPN AB
Title	Penetration Test Report
Targets	Wireguard VPN OpenVPN
Version	1.1
Pentester	Stefan Grönke
Authors	Stefan Grönke, Marcus Bointon
Reviewed by	Marcus Bointon
Approved by	Melanie Rieback

Version control

Version	Date	Author	Description
0.1	May 26th, 2023	Stefan Grönke	Initial draft
0.2	June 5th, 2023	Marcus Bointon	Review
1.0	June 8th, 2023	Marcus Bointon	1.0
0.4	July 7th, 2023	Stefan Grönke	Re-test findings
0.5	July 12th, 2023	Marcus Bointon	Re-test review
1.1	August 7th, 2023	Marcus Bointon	1.1

Contact

For more information about this document and its contents please contact Radically Open Security B.V.

Name	Melanie Rieback
Address	Science Park 608 1098 XH Amsterdam The Netherlands
Phone	+31 (0)20 2621 255
Email	info@radicallyopensecurity.com

Table of Contents

1 Executive Summary

1.1 Introduction

Between May 8, 2023 and May 26, 2023, Radically Open Security B.V. carried out a penetration test for Mullvad VPN AB.

This report contains our findings as well as detailed explanations of exactly how ROS performed the penetration test.

1.2 Scope of work

The scope of the penetration test was limited to the following targets:

- Wireguard VPN
- OpenVPN

The scoped services are broken down as follows:

- Host assesment of VPN servers: 10 days
- Reporting/PM/Review: 3-4 days
- (Optional) Retest: 1-3 days
- **Total effort: 14 - 17 days**

1.3 Project objectives

ROS will perform a host assessment of two production-alike VPN relay servers with Mullvad in order to assess the security of the WireGuard and OpenVPN relay servers and verify privacy objectives. To do so ROS will access two deployed systems via SSH and guide Mullvad in attempting to find vulnerabilities, exploiting any such found to try and gain further access and elevated privileges.

1.4 Timeline

The security audit took place between May 8, 2023 and May 26, 2023.

1.5 Results In A Nutshell

During this crystal-box penetration test we found 1 High, 6 Elevated, 4 Moderate, 10 Low and 4 Info-severity issues.

ROS audited two production-alike VPN relays, which were supposed to be isolated from user traffic. However, we found our WireGuard test server was listed in the production environment as a multihop server, which users have apparently been using MLL-024 (page 14). Compromised administrators potentially have access to user traffic through root access on production machines MLL-045 (page 24) without audit logging MLL-044 (page 30) or sudo password MLL-047 (page 44). Even with audit logging enabled, multiple local privilege escalation vulnerabilities in Netfilter MLL-022 (page 22) and misconfigured permissions on a systemd timer script MLL-019 (page 20) can be used to become root without leaving traces. In this area we also found misconfigured home folder permissions, allowing arbitrary system accounts to compromise another service account MLL-008 (page 17).

Misconfiguration in the IPMI interface can break isolation between VPN and management networks MLL-031 (page 23) when the IPMI interface is down. Interestingly, IPMI can be used on memory-only systems by adversaries to gain persistence MLL-009 (page 18).

Linux can be further hardened, for instance by confirming AppArmor profiles MLL-038 (page 29), deploying Microcode updates MLL-033 (page 27), minimizing the userspace MLL-012 (page 25), using a pre-shared key on WireGuard interfaces MLL-030 (page 39), adding binary hardening to services MLL-029 (page 37), ensuring the effectiveness of reverse-path filtering MLL-027 (page 36).

SSH agent forwarding should be disabled MLL-007 (page 33) to mitigate mistakes in administrator's SSH client configuration.

Some services on VPN servers use shared credentials; we recommend using unique per-system credentials for blocklist-service MLL-039 (page 42), SNMP MLL-021 (page 45), Slack notifications MLL-018 (page 46), and Telegraf MLL-016 (page 47). It might be feasible to switch to SSL client certificate authentication.

Overall, we found that significant effort had been put into preserving VPN clients privacy. We appreciate the in-memory OS approach and the absence of logging of user-related (meta)data, but recommend logging administrator activity on production machines. Ideally administrators should not be able to access production systems, but instead boot a server from a debug image with login capability which does not serve production traffic, so that admin access and production traffic are never active at the same time. The two WireGuard and OpenVPN servers accessed by ROS were appropriately hardened against client side attacks, so a majority of the findings and remarks refer to further OS hardening, administrative access, and misconfigured permissions.

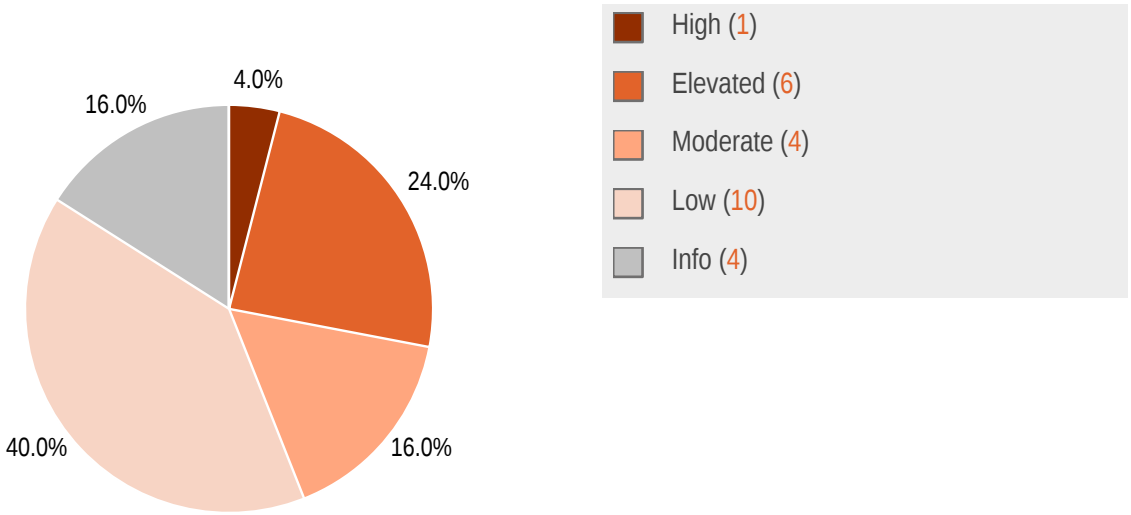
1.6 Summary of Findings

ID	Type	Description	Threat level
MLL-024	Information Disclosure	The VPN server used for testing processes multihop traffic for production VPN users.	High
MLL-008	Local Privilege Escalation	The monitor user's home directory can be hijacked by unprivileged system accounts.	Elevated
MLL-009	Persistence	Although VPN servers run from memory only, adversaries with root access can persist access by manipulating IPMI configuration and firmware.	Elevated

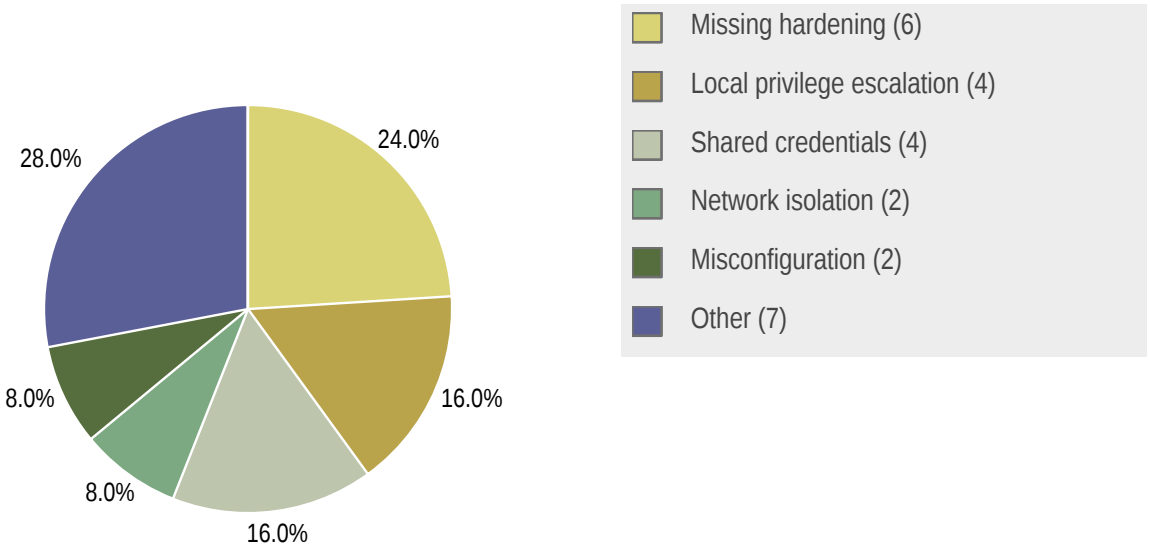
MLL-019	Local Privilege Escalation	Improper filesystem permissions on /home/mad allows other accounts to manipulate script files executed as root through a periodic systemd timer.	Elevated
MLL-022	Local Privilege Escalation	VPN servers use a custom Linux Kernel 6.3.0, which is vulnerable to CVE-2023-32233, a local privilege escalation in netfilter.	Elevated
MLL-031	Network Isolation	The IPMI network interface is configured in failover mode, so the IMPI can be accessed by system accounts or processes with network access, possibly through local Dante proxy server.	Elevated
MLL-044	Audit Logs	Administrator logins or executed commands are not logged remotely to provide an audit trail.	Elevated
MLL-045	Insufficient Access Controls	Administrators can log in to VPN servers through a jumphost via SSH (and become root), so that staff members can, in theory, silently tap into users VPN traffic.	Moderate
MLL-012	Unnecessary System Complexity	VPN servers are based on Ubuntu and have additional tools installed that are not required to provide the services.	Moderate
MLL-033	Missing Hardening	Hardware servers were missing microcode updates, typically used to mitigate vulnerabilities (like Spectre and Meltdown) in modern CPUs.	Moderate
MLL-038	Missing Hardening	Most services on both VPN servers (OpenVPN and Wireguard) have no AppArmor profiles loaded.	Moderate
MLL-003	Local Privilege Escalation	The GPG-Agent running as root reads configuration files from the home directory of an unprivileged account.	Low
MLL-007	Misconfiguration	The SSH daemon allows agent forwarding, which can expose the credentials of an administrator connecting to the machine.	Low
MLL-027	Misconfiguration	Loose reverse path forwarding was enabled on all interfaces, but without directionality has no effect due to default routes.	Low
MLL-029	Missing Hardening	The tcp2udp and blocklist-service binaries, which are self-maintained by Mullvad, are missing binary hardening.	Low
MLL-030	Missing Hardening	Internal WireGuard connections do not use a pre-shared key as an additional security layer to defend against cryptographic attacks.	Low
MLL-039	Shared Credentials	The blocklist service shares the same credentials across all VPN servers.	Low
MLL-047	Missing Hardening	Sudoers on VPN servers do not require password authentication, so a compromised SSH key could allow immediate access to privileged accounts.	Low
MLL-021	Shared Credentials	SNMP daemon credentials are encrypted but are re-used across servers.	Low
MLL-018	Shared Credentials	Slack notification credentials can be used to push misleading messages to the alerting Slack channel.	Low

		Credentials are re-used across VPN servers, so that identifying the origin for revocation may be hindered.	
MLL-016	Shared Credentials	Telegraf credentials are re-used across VPN servers.	Low
MLL-013	Missing Compartmentalization	Network namespaces (ip netns) or Linux containers are not used on VPN servers.	Info
MLL-014	Network Isolation	Outgoing iptables SNAT rules do not specify the outgoing interface, allowing traffic to flow in unexpected directions.	Info
MLL-037	DNS Hijacking	When using DNS blocklisting on the VPN server, firewall NAT rules apply to any connection to UDP port 53. Consequently, all outgoing DNS requests are hijacked and forced to use the VPN server's local DNS resolver.	Info
MLL-043	Missing Hardening	Blocklist-service is listening on 0.0.0.0, but it is not allowlisted in the firewall configuration.	Info

1.6.1 Findings by Threat Level



1.6.2 Findings by Type



1.7 Summary of Recommendations

ID	Type	Recommendation
MLL-024	Information Disclosure	• Filter production-alike systems (used for pentesting) from public relay lists.
MLL-008	Local Privilege Escalation	• Do not nest home directories. • Fix misconfigured filesystem permission of `/home/mad`.
MLL-009	Persistence	• Consider using servers without IPMI modules. • Consider using OpenBMC and disable mgmt from the server. • Disable IPMI kernel modules to prevent malicious root users from persisting access on hardware servers without access to the IPMI management network, which can be bypassed by attackers. • Consider using secure boot.
MLL-019	Local Privilege Escalation	• Set secure filesystem permissions on `/home/mad`. • Consider running periodic timer scripts as an unprivileged user.
MLL-022	Local Privilege Escalation	• Upgrade the Linux Kernel to a version newer than 6.3.1.
MLL-031	Network Isolation	• Disable IPMI failover mode.
MLL-045	Insufficient Access Controls	• Disable SSH and user login on production VPN hosts. • Reboot servers in an isolated debug state before allowing authenticated remote access over SSH. • Disable local console login via account password (e.g. access through IPMI).
MLL-012	Unnecessary System Complexity	• Minimize installed Ubuntu packages.

		• Install additional tools only when required, and reset the server after finding the cause of an issue. • In case of issues, consider resetting and deploying a server with additional administrative tools. This may lose the remote error state, but allows disconnecting the host system from the Internet so that only incoming VPN traffic can access the Internet.
MLL-033	Missing Hardening	• Deploy CPU Microcode updates on VPN servers.
MLL-038	Missing Hardening	• Enable AppArmor for exposed network and local services.
MLL-044	Audit Logs	• Log SSH authentication and executed commands on a remote system. • VPN server administrators must not have permission to delete/modify logged messages.
MLL-003	Local Privilege Escalation	• Ensure that the permissions for the GPG-agent config folder only allow write access by the user running the daemon.
MLL-007	Misconfiguration	• Set `AllowAgentForwarding no` in `/etc/ssh/sshd_config` or a file in `/etc/ssh/sshd_config.d`.
MLL-013	Missing Compartmentalization	• Use Linux network namespaces to ensure no traffic leaks between various services, or production and management networks.
MLL-014	Network Isolation	• Specify the outgoing interface in iptables NAT rules.
MLL-027	Misconfiguration	• Use RPF strict mode (`rp_filter=1`).
MLL-029	Missing Hardening	• Ensure the build process includes the build flags for hardening, such as `-Z stack-protector`, and that in release builds which are deployed to servers, that `strip = true` is set.
MLL-030	Missing Hardening	• Set WireGuard pre-shared keys for additional hardening.
MLL-037	DNS Hijacking	• Redirect DNS traffic (UDP port 53) only for DNS servers in private network ranges, so that clients can still connect to custom DNS servers.
MLL-039	Shared Credentials	• Ensure that unique credentials are generated per server. • Consider using client certificates to authenticate towards internal services.
MLL-043	Missing Hardening	• Bind blocklist-service to a specific interface to prevent public exposure in case of a firewall misconfiguration.
MLL-047	Missing Hardening	• Consider requiring further authentication (password, OTP) by administrator accounts before allowing them to become root on VPN servers.
MLL-021	Shared Credentials	• Use unique SNMP credentials per server.
MLL-018	Shared Credentials	• Use unique Slack notification credentials per VPN server. • Consider using a pull approach instead of allowing VPN servers to push unfiltered alert messages.
MLL-016	Shared Credentials	• Use unique credentials per VPN server appliance.

2 Methodology

2.1 Planning

Our general approach during penetration tests is as follows:

1. **Reconnaissance**
   We attempt to gather as much information as possible about the target. Reconnaissance can take two forms: active and passive. A passive attack is always the best starting point as this would normally defeat intrusion detection systems and other forms of protection afforded to the app or network. This usually involves trying to discover publicly available information by visiting websites, newsgroups, etc. An active form would be more intrusive, could possibly show up in audit logs and might take the form of a social engineering type of attack.
2. **Enumeration**
   We use various fingerprinting tools to determine what hosts are visible on the target network and, more importantly, try to ascertain what services and operating systems they are running. Visible services are researched further to tailor subsequent tests to match.
3. **Scanning**
   Vulnerability scanners are used to scan all discovered hosts for known vulnerabilities or weaknesses. The results are analyzed to determine if there are any vulnerabilities that could be exploited to gain access or enhance privileges to target hosts.
4. **Obtaining Access**
   We use the results of the scans to assist in attempting to obtain access to target systems and services, or to escalate privileges where access has been obtained (either legitimately though provided credentials, or via vulnerabilities). This may be done surreptitiously (for example to try to evade intrusion detection systems or rate limits) or by more aggressive brute-force methods. This step also consist of manually testing the application against the latest (2017) list of OWASP Top 10 risks. The discovered vulnerabilities from scanning and manual testing are moreover used to further elevate access on the application.

2.2 Risk Classification

Throughout the report, vulnerabilities or risks are labeled and categorized according to the Penetration Testing Execution Standard (PTES). For more information, see: http://www.pentest-standard.org/index.php/Reporting

These categories are:

- **Extreme**
  Extreme risk of security controls being compromised with the possibility of catastrophic financial/reputational losses occurring as a result.

- **High**

  High risk of security controls being compromised with the potential for significant financial/reputational losses occurring as a result.
- **Elevated**

  Elevated risk of security controls being compromised with the potential for material financial/reputational losses occurring as a result.
- **Moderate**

  Moderate risk of security controls being compromised with the potential for limited financial/reputational losses occurring as a result.
- **Low**

  Low risk of security controls being compromised with measurable negative impacts as a result.
- **Info**

  Observations worth mentioning. No direct risk of security controls being compromised with measurable negative impacts as a result.

3 Reconnaissance and Fingerprinting

We were able to gain information about the software and infrastructure through the following automated scans. Any relevant scan output will be referred to in the findings.

- nmap – https://nmap.org
- tcpdump – https://www.tcpdump.org/
- checksec – https://github.com/slimm609/checksec.sh
- tcpdump – https://www.tcpdump.org/

4 Findings

We have identified the following issues:

4.1 MLL-024 — Production multihop traffic on test system

Vulnerability ID: MLL-024 **Status:** Not Retested

Vulnerability type: Information Disclosure

Threat level: High

Description:

The VPN server used for testing processes multihop traffic for production VPN users.

Technical description:

Servers that ROS was given access to for testing purposes should be isolated from production data, but we found that the Wireguard host was receiving production user traffic via multihop configuration:

```
17:07:16.717652 wg-multihop In  IP 10.[REDACTED].64.47926 > 10.[REDACTED].84.1080: Flags [.], ack
 25, win 507, options [nop,nop,TS val 274642009 ecr 3376830347], length 0
17:07:16.717666 wg-multihop In  IP 10.[REDACTED].64.47926 > 10.[REDACTED].84.1080: Flags [P.], seq
 21:49, ack 25, win 507, options [nop,nop,TS val 274642009 ecr 3376830347], length 28
17:07:16.717666 wg-multihop In  IP 10.[REDACTED].64.47926 > 10.[REDACTED].84.1080: Flags [P.], seq
 49:92, ack 25, win 507, options [nop,nop,TS val 274642009 ecr 3376830347], length 43
17:07:16.717666 wg-multihop In  IP 10.[REDACTED].64.47926 > 10.[REDACTED].84.1080: Flags [P.], seq
 92:107, ack 25, win 507, options [nop,nop,TS val 274642009 ecr 3376830347], length 15
17:07:16.717777 wg-multihop Out IP 10.[REDACTED].84.1080 > 10.[REDACTED].64.47926: Flags [.], ack
 107, win 83, options [nop,nop,TS val 3376830365 ecr 274642009], length 0
17:07:16.717806 eth2  Out IP6 2a03:[REDACTED]::a99d.47926 > 2a00:[REDACTED]::200e.80: Flags [P.],
 seq 1:87, ack 1, win 85, options [nop,nop,TS val 1360333362 ecr 1998213920], length 86: HTTP: GET /
generate_204 HTTP/1.0
17:07:16.719104 eth2  In  IP6 2a00:[REDACTED]::200e.80 > 2a03:[REDACTED]::a99d.47926: Flags [.], ack
 87, win 256, options [nop,nop,TS val 1998213939 ecr 1360333362], length 0
17:07:16.719648 eth2  In  IP6 2a00:[REDACTED]::200e.80 > 2a03:[REDACTED]::a99d.47926: Flags [P.],
 seq 1:128, ack 87, win 256, options [nop,nop,TS val 1998213940 ecr 1360333362], length 127: HTTP:
 HTTP/1.0 204 No Content
17:07:16.719648 eth2  In  IP6 2a00:[REDACTED]::200e.80 > 2a03:[REDACTED]::a99d.47926: Flags [F.],
 seq 128, ack 87, win 256, options [nop,nop,TS val 1998213940 ecr 1360333362], length 0
17:07:16.719684 eth2  Out IP6 2a03:[REDACTED]::a99d.47926 > 2a00:[REDACTED]::200e.80: Flags [.], ack
 128, win 85, options [nop,nop,TS val 1360333364 ecr 1998213940], length 0
17:07:16.719765 wg-multihop Out IP 10.[REDACTED].84.1080 > 10.[REDACTED].64.47926: Flags [P.], seq
 25:152, ack 107, win 83, options [nop,nop,TS val 3376830367 ecr 274642009], length 127
17:07:16.719825 wg-multihop Out IP 10.[REDACTED].84.1080 > 10.[REDACTED].64.47926: Flags [F.], seq
 152, ack 107, win 83, options [nop,nop,TS val 3376830367 ecr 274642009], length 0
17:07:16.737567 wg-multihop In  IP 10.[REDACTED].64.47926 > 10.[REDACTED].84.1080: Flags [.], ack
 152, win 507, options [nop,nop,TS val 274642028 ecr 3376830367], length 0
```

```
17:07:16.737606 wg-multihop In  IP 10.[REDACTED].64.47926 > 10.[REDACTED].84.1080: Flags [F.], seq
 107, ack 153, win 507, options [nop,nop,TS val 274642028 ecr 3376830367], length 0
17:07:16.737635 wg-multihop Out IP 10.[REDACTED].84.1080 > 10.[REDACTED].64.47926: Flags [.], ack
 108, win 83, options [nop,nop,TS val 3376830385 ecr 274642028], length 0
17:07:16.737780 eth2  Out IP6 2a03:[REDACTED]::a99d.47926 > 2a00:[REDACTED]::200e.80: Flags [F.],
 seq 87, ack 129, win 85, options [nop,nop,TS val 1360333382 ecr 1998213940], length 0
17:07:16.738897 eth2  In  IP6 2a00:[REDACTED]::200e.80 > 2a03:[REDACTED]::a99d.47926: Flags [.], ack
 88, win 256, options [nop,nop,TS val 1998213959 ecr 1360333382], length 0
17:07:17.472983 eth2  In  IP 94.[REDACTED].178 > 185.[REDACTED].75: ICMP 94.[REDACTED].178 udp port
 51821 unreachable, length 184
17:07:17.472983 eth2  In  IP 94.[REDACTED].194 > 185.[REDACTED].75: ICMP 94.[REDACTED].194 udp port
 51821 unreachable, length 184
17:07:17.613631 eth2  M   IP6 fe80::827f:f800:b5b:a870 > ff02::1:ff00:20e: ICMP6, neighbor
 solicitation, who has 2a03:[REDACTED]::20e, length 32
17:07:18.247613 eth2  In  IP 94.[REDACTED].146 > 185.[REDACTED].75: ICMP 94.[REDACTED].146 udp port
 51821 unreachable, length 184
17:07:18.613408 eth2  M   IP6 fe80::827f:f800:b5b:a870 > ff02::1:ff00:20e: ICMP6, neighbor
 solicitation, who has 2a03:[REDACTED]::20e, length 32
17:07:18.843387 eth2  In  IP 89.[REDACTED].210 > 185.[REDACTED].75: ICMP 89.[REDACTED].210 udp port
 51821 unreachable, length 184
17:07:18.901111 eth2  In  IP 138.[REDACTED].169 > 185.[REDACTED].75: ICMP host 37.[REDACTED].14
 unreachable, length 184
17:07:19.137399 eth2  In  IP 45.[REDACTED].194.443 > 185.[REDACTED].75.43588: Flags [P.], seq
 4634:4874, ack 1, win 126, options [nop,nop,TS val 2991649552 ecr 920503678], length 240
17:07:19.137426 eth2  Out IP 185.[REDACTED].75.43588 > 45.[REDACTED].194.443: Flags [.], ack 4874,
 win 1190, options [nop,nop,TS val 920506675 ecr 2991649552], length 0
17:07:19.233496 lo    In  IP 127.0.0.1.52070 > 127.0.0.1.8125: UDP, length 27
17:07:19.572223 eth2  In  IP 62.[REDACTED].98 > 185.[REDACTED].75: ICMP host 185.156.46.130
 unreachable, length 184
17:07:19.618292 eth2  In  IP 168.[REDACTED].769 > 185.[REDACTED].251.38286: Flags [P.], seq 92:138,
 ack 93, win 7020, options [nop,nop,TS val 3200266737 ecr 448630235], length 46
17:07:19.618331 eth2  Out IP 185.[REDACTED].251.38286 > 168.[REDACTED].769: Flags [.], ack 138, win
 1182, options [nop,nop,TS val 448635038 ecr 3200266737], length 0
17:07:19.618436 wg-multihop Out IP 10.[REDACTED].84.1080 > 10.[REDACTED].24.38286: Flags [P.], seq
 92:138, ack 93, win 2883, options [nop,nop,TS val 922629694 ecr 3822465962], length 46
17:07:19.657042 eth2  In  IP 95.[REDACTED].150.52171 > 185.[REDACTED].251.55735: Flags [S], seq
 3379644437, win 32767, options [mss 1460,sackOK,TS val 1190584076 ecr 0,nop,wscale 11], length 0
17:07:19.659037 wg-multihop In  IP 10.[REDACTED].24.38286 > 10.[REDACTED].84.1080: Flags [.], ack
 138, win 501, options [nop,nop,TS val 3822470720 ecr 922629694], length 0
17:07:19.660035 wg-multihop In  IP 10.[REDACTED].24.38286 > 10.[REDACTED].84.1080: Flags [P.], seq
 93:139, ack 138, win 501, options [nop,nop,TS val 3822470722 ecr 922629694], length 46
17:07:19.660139 eth2  Out IP 185.[REDACTED].251.38286 > 168.[REDACTED].769: Flags [P.], seq 93:139,
 ack 138, win 1182, options [nop,nop,TS val 448635080 ecr 3200266737], length 46
17:07:19.685140 eth2  In  IP 168.[REDACTED].769 > 185.[REDACTED].251.38286: Flags [.], ack 139, win
 7020, options [nop,nop,TS val 3200266804 ecr 448635080], length 0
17:07:19.702674 wg-multihop Out IP 10.[REDACTED].84.1080 > 10.[REDACTED].24.38286: Flags [.], ack
 139, win 2883, options [nop,nop,TS val 922629778 ecr 3822470722], length 0
17:07:19.777369 eth2  In  IP 94.[REDACTED].162 > 185.[REDACTED].75: ICMP 94.[REDACTED].162 udp port
 51821 unreachable, length 184
17:07:19.942381 wg-multihop In  IP 10.[REDACTED].114.49052 > 10.[REDACTED].84.1080: Flags [S], seq
 3432764985, win 64240, options [mss 1380,sackOK,TS val 1945450825 ecr 0,nop,wscale 7], length 0
17:07:19.942429 wg-multihop Out IP 10.[REDACTED].84.1080 > 10.[REDACTED].114.49052: Flags [S.],
 seq 2138078350, ack 3432764986, win 42408, options [mss 1380,nop,nop,TS val 4218433239 ecr
 1945450825,nop,wscale 9], length 0
17:07:19.960246 eth2  In  IP 45.[REDACTED].194.443 > 185.[REDACTED].75.43588: Flags [P.], seq
 4874:5118, ack 1, win 126, options [nop,nop,TS val 2991650375 ecr 920506675], length 244
17:07:19.960281 eth2  Out IP 185.[REDACTED].75.43588 > 45.[REDACTED].194.443: Flags [.], ack 5118,
 win 1190, options [nop,nop,TS val 920507497 ecr 2991650375], length 0
17:07:19.974609 wg-multihop In  IP 10.[REDACTED].114.49052 > 10.[REDACTED].84.1080: Flags [.], ack
 1, win 502, options [nop,nop,TS val 1945450864 ecr 4218433239], length 0
```

```
17:07:19.974643 wg-multihop In  IP 10.[REDACTED].114.49052 > 10.[REDACTED].84.1080: Flags [P.], seq
 1:4, ack 1, win 502, options [nop,nop,TS val 1945450865 ecr 4218433239], length 3
17:07:19.974651 wg-multihop Out IP 10.[REDACTED].84.1080 > 10.[REDACTED].114.49052: Flags [.], ack
 4, win 83, options [nop,nop,TS val 4218433271 ecr 1945450865], length 0
17:07:19.974824 wg-multihop Out IP 10.[REDACTED].84.1080 > 10.[REDACTED].114.49052: Flags [P.], seq
 1:3, ack 4, win 83, options [nop,nop,TS val 4218433271 ecr 1945450865], length 2
17:07:20.007097 wg-multihop In  IP 10.[REDACTED].114.49052 > 10.[REDACTED].84.1080: Flags [.], ack
 3, win 502, options [nop,nop,TS val 1945450897 ecr 4218433271], length 0
17:07:20.007110 wg-multihop In  IP 10.[REDACTED].114.49052 > 10.[REDACTED].84.1080: Flags [P.], seq
 4:27, ack 3, win 502, options [nop,nop,TS val 1945450897 ecr 4218433271], length 23
17:07:20.007712 lo    In  IP 127.0.0.1.40772 > 127.0.0.53.53: 49743+ [1au] A? gs-loc.apple.com. (45)
17:07:20.007722 lo    In  IP 127.0.0.1.40772 > 127.0.0.53.53: 25165+ [1au] AAAA? gs-loc.apple.com.
 (45)
17:07:20.007888 lo    In  IP 127.0.0.1.51051 > 127.0.0.1.53: 49462+ [1au] A? gs-loc.apple.com. (45)
17:07:20.007993 lo    In  IP 127.0.0.1.35313 > 127.0.0.1.53: 1953+ [1au] AAAA? gs-loc.apple.com.
 (45)
17:07:20.008295 eth2  Out IP6 2a03:[REDACTED]::a99d.51090 > 2600:[REDACTED]::83.53: 40699 [1au] A?
 _.com.akadns.net. (57)
17:07:20.008999 eth2  In  IP 45.[REDACTED].194.443 > 185.[REDACTED].75.43588: Flags [P.], seq
 5118:5357, ack 1, win 126, options [nop,nop,TS val 2991650423 ecr 920507497], length 239
17:07:20.009039 eth2  Out IP 185.[REDACTED].75.43588 > 45.[REDACTED].194.443: Flags [.], ack 5357,
 win 1190, options [nop,nop,TS val 920507546 ecr 2991650423], length 0
17:07:20.017745 eth2  In  IP6 2600:[REDACTED]::83.53 > 2a03:[REDACTED]::a99d.51090: 40699 NXDomain*-
 0/1/1 (111)
17:07:20.018101 eth2  Out IP6 2a03:[REDACTED]::a99d.40173 > 2600:[REDACTED]::83.53: 64383 [1au] A?
 _.ls-apple.com.akadns.net. (66)
17:07:20.027486 eth2  In  IP6 2600:[REDACTED]::83.53 > 2a03:[REDACTED]::a99d.40173: 64383 NXDomain*-
 0/1/1 (120)
17:07:20.027921 eth2  Out IP6 2a03:[REDACTED]::a99d.45163 > 2001:[REDACTED]::f.53: 17683% [1au]
 AAAA? a1-128.akadns.net. (58)
17:07:20.027921 eth2  Out IP6 2a03:[REDACTED]::a99d.41712 > 2600:[REDACTED]::83.53: 36046 [1au] A?
 gs-loc-new.ls-apple.com.akadns.net. (75)
17:07:20.027951 eth2  Out IP6 2a03:[REDACTED]::a99d.43390 > 2600:[REDACTED]::83.53: 60156 [1au]
 AAAA? gs-loc-new.ls-apple.com.akadns.net. (75)
17:07:20.028041 eth2  Out IP6 2a03:[REDACTED]::a99d.51532 > 2001:[REDACTED]::f.53: 7948% [1au] AAAA?
 a9-128.akadns.net. (58)
17:07:20.028054 eth2  Out IP6 2a03:[REDACTED]::a99d.48142 > 2001:[REDACTED]::f.53: 62837% [1au]
 AAAA? a11-129.akadns.net. (59)
```

Testing nodes accessed by ROS were not available for users uplink, but were listed in the public relays list:

```
% curl -s https://api.mullvad.net/internal/relays | jq '.[] | select(.ipv4_addr_in=="185.
[REDACTED].75")'
{
  "hostname": "se-sto-wg-099",
  "hostname_socks": "se-sto-wg-socks5-099",
  "ipv4_addr_in": "185.[REDACTED].75",
  "ipv6_addr_in": "2a03:[REDACTED]::a99f",
  "ssh_public_key": "ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIJ9x/oJEfvFP76mNA1j9t9eo/
XSmVAoHJCgnok5LPj7F",
  "wg_public_key": "36P6tblwQ1TSlH3ZdW85p1+8gMPeo1pSMu+SarZRsTU=",
  "wg_multihop_public_key": "JzJk/7Z9Xo2H5k8TSHPS1HVtW65h+SgUXeGaxh18QhQ=",
  "wg_internal_public_key": "yhfx3j4E93/5RkDjyA12VNFke/IzT/kGCuOeEgDDkD8=",
  "multihop_ipv4": "10.[REDACTED].84",
  "multihop_ipv6": "fd00:aaaa::84"
}
```

This means that it's possible for users to find and use the system as a wg-multihop relay.

Impact:

Production user traffic is visible to pentest users.

Recommendation:

- Filter production-alike systems (used for pentesting) from public relay lists.

4.2 MLL-008 — Home directory of monitor can be hijacked

Vulnerability ID: MLL-008	**Status:** Resolved
Vulnerability type: Local Privilege Escalation	
Threat level: Elevated	

Description:

The `monitor` user's home directory can be hijacked by unprivileged system accounts.

Technical description:

The home directory of the `monitor` user account is nested in the `mad` user's home directory, which is group-writable, allowing any member of the `mad` group to take over the `monitor` user's home directory:

```
gronke@ch-zrh-ovpn-299:/home/mad$ cat /etc/passwd | grep monitor
monitor:x:997:997::/home/mad/monitor:/usr/sbin/nologin
gronke@ch-zrh-ovpn-299:~$ ls -al /home/ | grep mad
drwxrwx--x 16 mad mad 400 May 8 15:15 mad
gronke@ch-zrh-ovpn-299:/home/mad$ mv monitor monitor2
gronke@ch-zrh-ovpn-299:/home/mad$ ls -al | grep monitor
drwx------   3 monitor  root      240 May  8 19:44 monitor2
gronke@ch-zrh-ovpn-299:/home/mad$ mv monitor2/ monitor
```

To take advantage of this, the account attempting to hijack the `monitor` users home directory must be member of `mad` group (for write access to `/home/mad`). Such users are also members of the `sudoers` group (so they effectively have root access anyway), but local privilege escalation via takeover of this service account can be used to silently elevate privileges without leaving traces in sudo log events. The absence of those logs that we found in MLL-044 (page 30) does not change the need to harden the `monitor` user's home directory.

Separately we note that the `mad` home directory has world-executable permissions, which seems unnecessary.

Impact:

Any member of the `mad` group can hijack the `monitor` user's home directory and potentially escalate privileges to that Linux account.

Recommendation:

- Do not nest home directories.
- Fix misconfigured filesystem permission of `/home/mad`.

Update :

During the retest, we found that the `monitor` user's home directory is still within `/home/mad/monitor`.

```
gronke@ch-zrh-ovpn-299:~$ cat /etc/passwd | grep monitor
monitor:x:996:996::/home/mad/monitor:/usr/sbin/nologin
gronke@ch-zrh-ovpn-299:~$ ls -al /home/ | grep mad
drwxrwx--x 15 mad         mad         420 Jun 21 07:12 mad
gronke@ch-zrh-ovpn-299:~$ sudo ls -al /home/mad/ | grep monitor
drwx------   3 monitor  root      240 Jun 21 07:12 monitor
```

Unlike before, no other accounts are members of the `mad` group. With this change, the risk of administrators or other processes escalating privileges to monitor now only exists for the `mad` user itself through ownership of the parent directory. Although avoidable, the design decision to put the `monitor` account's home directory inside the `mad` user's home directory has no apparent security impact.

4.3 MLL-009 — IPMItool allows persistence on otherwise memory-only servers

Vulnerability ID: MLL-009

Vulnerability type: Persistence

Threat level: Elevated

Description:

Although VPN servers run from memory only, adversaries with root access can persist access by manipulating IPMI configuration and firmware.

Technical description:

Although VPN servers run from memory only, adversaries that obtain root access (for instance via MLL-019 (page 20) or MLL-022 (page 22)) can persist access by manipulating IPMI configuration and firmware:

```
root@ch-zrh-ovpn-299:~# ipmitool lan print
Set in Progress         : Set Complete
Auth Type Support       : NONE MD2 MD5 PASSWORD
Auth Type Enable        : Callback : MD2 MD5 PASSWORD
                        : User     : MD2 MD5 PASSWORD
                        : Operator : MD2 MD5 PASSWORD
                        : Admin    : MD2 MD5 PASSWORD
                        : OEM      : MD2 MD5 PASSWORD
IP Address Source       : Static Address
IP Address              : 192.168.40.3
Subnet Mask             : 255.255.0.0
MAC Address             : 3c:ec:ef:5b:2f:c3
SNMP Community String   : public
IP Header               : TTL=0x00 Flags=0x00 Precedence=0x00 TOS=0x00
BMC ARP Control         : ARP Responses Enabled, Gratuitous ARP Disabled
Default Gateway IP      : 172.21.0.1
Default Gateway MAC     : 00:00:00:00:00:00
Backup Gateway IP       : 0.0.0.0
Backup Gateway MAC      : 00:00:00:00:00:00
802.1q VLAN ID          : Disabled
802.1q VLAN Priority    : 0
RMCP+ Cipher Suites     : 1,2,3,6,7,8,11,12
Cipher Suite Priv Max   : XaaaXXaaaXXaaXX
                        :     X=Cipher Suite Unused
                        :     c=CALLBACK
                        :     u=USER
                        :     o=OPERATOR
                        :     a=ADMIN
                        :     O=OEM
Bad Password Threshold  : 3
Invalid password disable: yes
Attempt Count Reset Int.: 300
User Lockout Interval   : 300
```

The severity of this finding is defined by the absence of system disks. Control over IPMI allows an attacker with root access to persist reboots, which would otherwise not be possible. To exploit this finding, an attacker obviously needs to first escalate privileges to root, which attempting was a major concern when conducting this pentest.

Impact:

Adversaries gaining root access on a memory-only VPN server can persist access on the machine by manipulating host firmware and configuration.

Recommendation:

- Consider using servers without IPMI modules.
- Consider using OpenBMC and disable mgmt from the server.
- Disable IPMI kernel modules to prevent malicious root users from persisting access on hardware servers without access to the IPMI management network, which can be bypassed by attackers.
- Consider using secure boot.

4.4 MLL-019 — LPE to root using systemd timers and insecure directory permissions

Vulnerability ID: MLL-019	**Status:** Resolved
Vulnerability type: Local Privilege Escalation	
Threat level: Elevated	

Description:

Improper filesystem permissions on `/home/mad` allows other accounts to manipulate script files executed as `root` through a periodic systemd timer.

Technical description:

A systemd timer `mullvad-check-local-resolver` runs a script `/home/mad/local_checks/check-local-resolver` periodically:

```
root@se-sto-wg-099:~# systemctl cat mullvad-check-local-resolver.timer
# /etc/systemd/system/mullvad-check-local-resolver.timer
[Unit]
# This ensures that this service is started when this timer is enabled
# Otherwise the service will never become inactive for this timer to start
Wants=mullvad-check-local-resolver.service
[Timer]
OnUnitInactiveSec=1h
```

```
root@se-sto-wg-099:~# systemctl cat mullvad-check-local-resolver.service
```

```
# /etc/systemd/system/mullvad-check-local-resolver.service
[Unit]
OnFailure=mullvad-rescue-local-resolver.service
StartLimitBurst=3
StartLimitInterval=5m
[Service]
ExecStart=/home/mad/local_checks/check-local-resolver
Restart=on-failure
```

Folder permissions on `/home/mad` allow other users to traverse the directory:

```
root@se-sto-wg-099:~# ls -al /home/mad
total 16
drwxrwx--x 13 mad                  mad                  340 May  9 20:10 .
drwxr-xr-x 11 root                 root                 220 May  2 11:01 ..
...
drwxr-xr-x  2 root                 root                 160 May  4 06:49 local_checks
...
```

With these permissions it is possible to rename the original `local_checks` directory and replace its contents, so that arbitrary code can be executed by any system account:

```
gronke@ch-zrh-ovpn-299:/home/mad$ mv local_checks local_checks.ori
gronke@ch-zrh-ovpn-299:/home/mad$ cp -r local_checks.ori local_checks
gronke@ch-zrh-ovpn-299:/home/mad$ echo "id > /tmp/ros.txt" >> local_checks/check-local-resolver
```

After the systemd timer has run, a file `/tmp/ros.txt` is created, showing the local privilege escalation to `root`:

```
root@ch-zrh-ovpn-299# cat /tmp/ros.txt
uid=0(root) gid=0(root) groups=0(root)
```

Impact:

Low-privileged system accounts can elevate their privileges to root by manipulating systemd timer script content.

Recommendation:

- Set secure filesystem permissions on `/home/mad`.
- Consider running periodic timer scripts as an unprivileged user.

Update :

The script has been moved to `/opt/local_checks/` with hardened filesystem permissions, resolving the issue.

```
gronke@ch-zrh-ovpn-299:~$ systemctl cat mullvad-check-local-resolver.service
# /etc/systemd/system/mullvad-check-local-resolver.service
[Unit]
OnFailure=mullvad-rescue-local-resolver.service
StartLimitBurst=3
```

```
StartLimitInterval=5m
[Service]
User=mullvad-local-checks
ExecStart=/opt/local_checks/check-local-resolver
Restart=on-failure
```

```
gronke@ch-zrh-ovpn-299:~$ ls -al /opt/local_checks/check-local-resolver
-rwxr-x--- 1 mullvad-local-checks mullvad-local-checks 285 Jun 21 07:01 /opt/local_checks/check-
local-resolver
```

4.5 MLL-022 — Netfilter flaw allows LPE to root

Vulnerability ID: MLL-022	**Status:** Resolved
Vulnerability type: Local Privilege Escalation	
Threat level: Elevated	

Description:

VPN servers use a custom Linux Kernel 6.3.0, which is vulnerable to CVE-2023-32233, a local privilege escalation in netfilter.

Technical description:

A recently published vulnerability CVE-2023-32233 affects the custom Linux Kernel 6.3.0 deployed to the testing VPN servers.

```
root@se-sto-wg-099:~# uname -a
Linux se-sto-wg-099 6.3.0-mullvad-gedd664ce9812 #1 SMP PREEMPT_DYNAMIC Mon Apr 24 20:27:13 CEST 2023
 x86_64 x86_64 x86_64 GNU/Linux
```

The vulnerability affects Linux Kernels <= 6.3.1 and was patched the following commit: https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/patch/?id=c1592a89942e9678f7d9c8030efa777c0d57edab.

Impact:

Low-privileged accounts can elevate their local privileges by exploiting CVE-2023-32233.

Recommendation:

- Upgrade the Linux Kernel to a version newer than 6.3.1.

Update :

Kernel has been updated to 6.3.2, which fixes the vulnerability.

4.6 MLL-031 — IPMI is configured in failover mode

Vulnerability ID: MLL-031	**Status:** Resolved
Vulnerability type: Network Isolation	
Threat level: Elevated	

Description:

The IPMI network interface is configured in failover mode, so the IMPI can be accessed by system accounts or processes with network access, possibly through local Dante proxy server.

Technical description:

The IPMI interface is not dedicated, but in failover mode. If it's disconnected, the interface is exposed to the LAN:

```
root@ch-zrh-ovpn-299:~# ipmitool raw 0x30 0x70 0x0c 0
 02
```

Impact:

Failover to the main network interface occurs when the IMPI's own LAN link is disconnected, so processes with network access may be able to read and write host firmware configuration by using known credentials, which is potentially possible through the Dante proxy service for VPN users.

Recommendation:

- Disable IPMI failover mode.

4.7 MLL-045 — Administrator access to production machines

Vulnerability ID: MLL-045	**Status:** Unresolved
Vulnerability type: Insufficient Access Controls	
Threat level: Moderate	

Description:

Administrators can log in to VPN servers through a jumphost via SSH (and become root), so that staff members can, in theory, silently tap into users VPN traffic.

Technical description:

Administrative access to memory-only VPN servers is possible. Remote accounts have to use allowlisted jumphosts to reach the SSH port, but the `sudoers` group membership would allow any such person to silently tap into VPN users' traffic.

We found that connection and command logging was not configured in MLL-044 (page 30), so activity in these accounts does not leave traces of malicious activity.

Remote access is an important tool for debugging networking issues, but when using ephemeral hosts, this level of access to a production system may not be required. The VPN server should be rebooted into a debug state instead, making the host unavailable for customer access. The operational risk of not having the ability to log in to production VPN servers with live user traffic appears to be fairly limited, so we recommend considering hardening the systems in this way.

Impact:

VPN servers accept remote logins from administrators, who technically have the ability to tap into production users' VPN traffic.

Recommendation:

- Disable SSH and user login on production VPN hosts.
- Reboot servers in an isolated debug state before allowing authenticated remote access over SSH.
- Disable local console login via account password (e.g. access through IPMI).

4.8 MLL-012 — Non-minimized Linux

Vulnerability ID: MLL-012	**Status:** Resolved
Vulnerability type: Unnecessary System Complexity	
Threat level: Moderate	

Description:

VPN servers are based on Ubuntu and have additional tools installed that are not required to provide the services.

Technical description:

Additional tools that are not required to provide VPN services to users are installed on the Ubuntu servers:

- ModemManager
- nmap
- tcpdump
- byobu
- cloud-guest-utils
- cryptsetup
- gcc-12-base
- git
- man-db
- manpages
- mdadm
- ntfs-3g
- netcat
- open-vm-tools
- open-iscsi
- pastebinit

Some tools are handy to debug issues on a remote machine, but also give administrators or adversaries who gain code execution on the remote system (for instance through another vulnerability) additional abilities or provide additional attack surface.

Without any modems present in the hardware setup, ModemManager has no particular use. Pre-installed tools like `tcpdump` or `pastebinit` may be used to compromise VPN users privacy.

Impact:

Additional tools installed on the VPN servers can increase the attack surface and provide remote accounts with pre-installed capabilities to compromise VPN users privacy.

Recommendation:

- Minimize installed Ubuntu packages.
- Install additional tools only when required, and reset the server after finding the cause of an issue.
- In case of issues, consider resetting and deploying a server with additional administrative tools. This may lose the remote error state, but allows disconnecting the host system from the Internet so that only incoming VPN traffic can access the Internet.

Update :

On retesting, we observed that the number of installed packages has been reduced. Only a few of the packages listed in the finding description are included in the updated VPN server image:

```
gronke@ch-zrh-ovpn-299:~$ dpkg --get-selections | grep -iE "(ModemManager|nmap|tcpdump|byobu|cloud-
guest-utils|cryptsetup|gcc-12-base|git|man-db|manpages|mdadm|ntfs-3g|netcat|open-vm-tools|open-
iscsi|pastebinit)"
cryptsetup                  install
cryptsetup-bin                  install
cryptsetup-initramfs                install
gcc-12-base:amd64               install
git                     install
git-man                     install
libcryptsetup12:amd64               install
libntfs-3g89                    install
man-db                      install
manpages                    install
mdadm                       install
netcat                      install
netcat-openbsd                  install
nmap                        install
nmap-common                 install
ntfs-3g                     install
open-iscsi                  install
tcpdump                     install
```

While tools like `NetworkManager` and `open-vm-tools` have been removed and reduce the attack surface, system administration tools like `tcpdump` and `nmap` are staples of a hacker's toolkit and can potentially be used to the disadvantage of connected VPN clients. It is understandable to leave such important debugging tools installed, until a solution is found to completely disable administrator access to production VPN servers, which would render those tools

useless on a production machine anyway. This topic is covered in other findings though, and the tools that have been removed from the base image are indeed the ones which had potential effect on the security posture of VPN servers.

4.9 MLL-033 — No microcode updates applied on servers

Vulnerability ID: MLL-033	**Status:** Resolved
Vulnerability type: Missing Hardening	
Threat level: Moderate	

Description:

Hardware servers were missing microcode updates, typically used to mitigate vulnerabilities (like Spectre and Meltdown) in modern CPUs.

Technical description:

Both hardware servers tested were found to be missing microcode updates, leaving the systems unprotected from publicly known vulnerabilities in their CPUs:

```
% dmesg | grep  -i microcode
[    7.203812] microcode: Microcode Update Driver: v2.2.
# hostnamectl
 Static hostname: ch-zrh-ovpn-299
       Icon name: computer-server
         Chassis: server
      Machine ID: c0a921a87d59449a9cb6554e41fdff54
         Boot ID: 8cace299fcb248bd954421d872112d12
Operating System: Ubuntu 22.04.2 LTS
          Kernel: Linux 6.3.0-mullvad-gedd664ce9812
    Architecture: x86-64
 Hardware Vendor: Supermicro
  Hardware Model: SYS-1029P-WTR
root@ch-zrh-ovpn-299:~#
```

```
% dmesg | grep -i microcode
# hostnamectl
 Static hostname: se-sto-wg-099
       Icon name: computer-server
         Chassis: server
      Machine ID: c0a921a87d59449a9cb6554e41fdff54
         Boot ID: 34edbfed0e4448a4b4d60481e542a7b9
Operating System: Ubuntu 22.04.2 LTS
          Kernel: Linux 6.3.0-mullvad-gedd664ce9812
    Architecture: x86-64
 Hardware Vendor: Supermicro
  Hardware Model: Super Server
```

```
root@se-sto-wg-099:/home/gronke#
```

Because the VPN servers do not execute arbitrary code from users, this vulnerability will not be exploitable by VPN users. Only administrators (who already have the ability to become root) can log in to the system. This makes exploitation unlikely, but we decided to report this issue anyway, and recommend applying patches, though this may lead to other issues, depending on which CPU features require mitigation.

Impact:

Adversaries with the ability to execute arbitrary code in unprivileged processes may exploit publicly known vulnerabilities in the system's CPUs.

Recommendation:

- Deploy CPU Microcode updates on VPN servers.

Update :

During the retest, we observed that Intel microcode updates have been installed and applied:

```
console
root@ch-zrh-ovpn-299:~# dpkg -s intel-microcode | grep Version
Version: 3.20230214.0ubuntu0.22.04.1
root@ch-zrh-ovpn-299:~# journalctl -b -k | grep "microcode"
Jun 21 06:41:58 production-stbuild-ubuntu kernel: microcode: Microcode Update Driver: v2.2.
root@ch-zrh-ovpn-299:~# dmesg | grep microcode
[    8.779213] microcode: Microcode Update Driver: v2.2.
root@ch-zrh-ovpn-299:~# dpkg --get-selections | grep microcode
intel-microcode                 install
root@ch-zrh-ovpn-299:~# grep -i micro /proc/cpuinfo
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
microcode   : 0x2006e05
```

4.10 MLL-038 — AppArmor is unconfined for most exposed services

Vulnerability ID: MLL-038	**Status:** Unresolved
Vulnerability type: Missing Hardening	
Threat level: Moderate	

Description:

Most services on both VPN servers (OpenVPN and Wireguard) have no AppArmor profiles loaded.

Technical description:

OpenVPN server

```
root@ch-zrh-ovpn-299:~# aa-unconfined
10359 /usr/sbin/snmpd not confined
11678 /usr/sbin/named confined by 'named (enforce)'
14754 /usr/bin/blocklist-service not confined
15158 /usr/sbin/danted not confined
15182 /usr/sbin/danted (danted: io-child: 0/32 (0 in progress)) not confined
19743 /usr/sbin/sshd (sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups) not confined
26770 /usr/bin/telegraf not confined
27965 /usr/lib/systemd/systemd-resolved (/lib/systemd/systemd-resolved) not confined
32778 /usr/sbin/openvpn not confined
32792 /usr/sbin/openvpn not confined
32806 /usr/sbin/openvpn not confined
32821 /usr/sbin/openvpn not confined
32835 /usr/sbin/openvpn not confined
32850 /usr/sbin/openvpn not confined
32904 /usr/sbin/openvpn not confined
32918 /usr/sbin/openvpn not confined
32933 /usr/sbin/openvpn not confined
32947 /usr/sbin/openvpn not confined
32961 /usr/sbin/openvpn not confined
32975 /usr/sbin/openvpn not confined
32989 /usr/sbin/openvpn not confined
```

WireGuard server

```
root@se-sto-wg-099:~# aa-unconfined
5411 /usr/sbin/sshd (sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups) not confined
11068 /usr/bin/telegraf not confined
11105 /usr/sbin/snmpd not confined
14119 /usr/sbin/named confined by 'named (enforce)'
14156 /usr/lib/systemd/systemd-resolved (/lib/systemd/systemd-resolved) not confined
14185 /usr/bin/blocklist-service not confined
16600 /usr/local/bin/tcp2udp confined by '/usr/local/bin/tcp2udp (enforce)'
16608 /usr/local/bin/tcp2udp confined by '/usr/local/bin/tcp2udp (enforce)'
16636 /usr/bin/wg-manager not confined
16661 /usr/sbin/danted not confined
16685 /usr/sbin/danted (danted: io-child: 0/32 (0 in progress)) not confined
```

```
16692 /usr/sbin/danted not confined
17054 /usr/sbin/danted (danted: io-child: 6/32 (0 in progress)) not confined
17057 /usr/sbin/danted (danted: io-child: 0/32 (0 in progress)) not confined
```

Impact:

Services without an AppArmor profile lack additional hardening of access control and security boundaries that could mitigate or limit potential damage from compromised services.

Recommendation:

- Enable AppArmor for exposed network and local services.

Update :

Unchanged on WireGuard server, not re-tested on OpenVPN due to `apparmor-utils` no longer being installed.

4.11 MLL-044 — No remote syslog and audit logging

Vulnerability ID: MLL-044	**Status:** Unresolved
Vulnerability type: Audit Logs	
Threat level: Elevated	

Description:

Administrator logins or executed commands are not logged remotely to provide an audit trail.

Technical description:

Syslog messages (already disabled for OpenVPN, WireGuard and DNS) are not logged remotely, so there is no record that can be used to identify malicious activity by remotely authenticated users. If remote access to production machines (as in MLL-045 (page 24)) is technically required, access events and executed commands should be logged remotely, to provide evidence for the absence of malicious activity.

Impact:

If apparently malicious activity is detected, the absence of remote logs means that investigations cannot distinguish inadvertent (or malicious) operations by system admins from those of genuine attackers.

Recommendation:

- Log SSH authentication and executed commands on a remote system.
- VPN server administrators must not have permission to delete/modify logged messages.

4.12 MLL-003 — gpg-agent running as root reads config from an unprivileged account

Vulnerability ID: MLL-003 **Status:** Unresolved

Vulnerability type: Local Privilege Escalation

Threat level: Low

Description:

The GPG-Agent running as root reads configuration files from the home directory of an unprivileged account.

Technical description:

GPG-agent reads from `/home/mad/.gnupg`:

```
root@ch-zrh-ovpn-299# ps aux | grep mad
root        8158  0.0  0.0  78788  2700 ?        Ss   May04   0:00 gpg-agent --homedir /home/
mad/.gnupg --use-standard-socket --daemon
```

By manipulating the configuration file in `/home/mad/.gnupg` as user can elevate their priviledges by overwriting the path to external programs called by the GPG agent, which are under control of the user.

```
--scdaemon-program filename
Use program filename as the Smartcard daemon. The default is installation dependent and can be shown
 with the gpgconf command.

--pinentry-program filename
Use program filename as the PIN entry. The default is installation dependent. With the default
 configuration the name of the default pinentry is pinentry; if that file does not exist but a
 pinentry-basic exist the latter is used.

On a Windows platform the default is to use the first existing program from this list: bin
\pinentry.exe, ..\Gpg4win\bin\pinentry.exe, ..\Gpg4win\pinentry.exe, ..\GNU\GnuPG\pinentry.exe, ..
```

```
\GNU\bin\pinentry.exe, bin\pinentry-basic.exe where the file names are relative to the GnuPG
 installation directory.
```

Impact:

An attacker with write permissions to `/home/mad/.gnupg` can elevate their privileges to root. However, given the ephemeral boot of the systems tested, this would only apply on GPG agent restart.

Recommendation:

- Ensure that the permissions for the GPG-agent config folder only allow write access by the user running the daemon.

Update :

At the time of re-testing, `gpg-agent` is still run as `root` with `/home/mad/.gnupg` as the home directory.

```
gronke@ch-zrh-ovpn-299:~$ ps ax | grep /home/mad
  16648 ?        Ss     0:00 gpg-agent --homedir /home/mad/.gnupg --use-standard-socket --daemon
  81241 pts/0    S+     0:00 grep --color=auto /home/mad
```

Unlike before, administrators are no longer members of the `mad` group and have lost privileges:

```
gronke@ch-zrh-ovpn-299:~$ ls -al /home | grep mad
drwxrwx--x 15 mad         mad         420 Jun 21 07:12 mad
gronke@ch-zrh-ovpn-299:~$ ls -al /home/mad
ls: cannot open directory '/home/mad': Permission denied
```

Because of `o+x` permissions on `/home/mad`, and read permission for the gpg-agent's home directory, other users still have read access.

```
gronke@ch-zrh-ovpn-299:~$ ls -al /home/mad/.gnupg
total 44
drwxr-xr-x  3 root root   200 Jun 21 07:10 .
drwxrwx--x 15 mad  mad    420 Jun 21 07:12 ..
srwx------  1 root root     0 Jun 21 07:01 S.gpg-agent
srwx------  1 root root     0 Jun 21 07:01 S.gpg-agent.browser
srwx------  1 root root     0 Jun 21 07:01 S.gpg-agent.extra
srwx------  1 root root     0 Jun 21 07:01 S.gpg-agent.ssh
drwx------  2 root root    40 Jun 21 07:01 private-keys-v1.d
-rw-r--r--  1 root root 19730 Jun 21 07:01 pubring.kbx
-rw-r--r--  1 root root 18407 Jun 21 07:01 pubring.kbx~
-rw-r--r--  1 root root  1840 Jun 21 07:10 trustdb.gpg
```

In the absence of members of the `mad` group, parts of the issue have been mitigated. The remaining risk (that a process or user running as `mad` could attempt escalating to root) has not been addressed. Counter to our earlier recommendation, `mad` still owns the home directory accessed by gpg-agent running as root.

4.13 MLL-007 — SSH daemon allows agent forwarding

Vulnerability ID: MLL-007 **Status:** Resolved

Vulnerability type: Misconfiguration

Threat level: Low

Description:

The SSH daemon allows agent forwarding, which can expose the credentials of an administrator connecting to the machine.

Technical description:

We found that the SSH daemon allows clients to enable agent forwarding:

```
root@ch-zrh-ovpn-299# sshd -T | grep -i agent
allowagentforwarding yes
```

When connecting with a misconfigured client, this could cause other system users to share an administrator's SSH keys unintentionally.

Impact:

Misconfigured SSH clients could accidentally enable SSH agent forwarding, making the administrators' SSH keys available to other administrators or privileged processes.

Recommendation:

- Set `AllowAgentForwarding no` in `/etc/ssh/sshd_config` or a file in `/etc/ssh/sshd_config.d`.

Update :

The issue was resolved during the assignment, right after noticing:

```
root@ch-zrh-ovpn-299:~$ sshd -T | grep -i agent
```

```
allowagentforwarding no
hostkeyagent none
```

4.14 MLL-013 — No usage of network namespaces or containers

Vulnerability ID: MLL-013

Vulnerability type: Missing Compartmentalization

Threat level: Info

Description:

Network namespaces (`ip netns`) or Linux containers are not used on VPN servers.

Technical description:

Neither network namespaces (`ip netns`) nor Linux containers are used on the VPN servers, although both appear to be a great measure to complement the existing configuration with additional hardening. The configuration found on VPN servers is mostly protected by iptables firewall rules, but network namespaces in particular can provide an additional layer of hardening. One important objective here is to separate production operation from management access; one should not compromise the other.

Impact:

Although strict firewall rules prevent unfortunate/unexpected routing of network traffic, Linux compartmentalization features could express similar hardening using rules that are easier to comprehend and debug, helping to prevent administrators from making configuration changes that could inadvertently undermine security objectives.

Recommendation:

- Use Linux network namespaces to ensure no traffic leaks between various services, or production and management networks.

4.15 MLL-014 — iptables SNAT does not specify outgoing interface

Vulnerability ID: MLL-014 **Status:** Resolved

Vulnerability type: Network Isolation

Threat level: Info

Description:

Outgoing iptables SNAT rules do not specify the outgoing interface, allowing traffic to flow in unexpected directions.

Technical description:

Outgoing iptables POSTROUTING NAT rules allow source NAT from VPN client addresses on arbitrary interfaces:

```
root@ch-zrh-ovpn-299:~# iptables -t nat -L POSTROUTING -v -n
Chain POSTROUTING (policy ACCEPT 171K packets, 9751K bytes)
 pkts bytes target  prot opt in  out  source         destination
    0     0 SNAT    all  --  *   *    10.7.0.0/16    0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
    0     0 SNAT    all  --  *   *    10.6.0.0/16    0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
    0     0 SNAT    all  --  *   *    10.5.0.0/16    0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
    0     0 SNAT    all  --  *   *    10.8.0.0/16    0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
    0     0 SNAT    all  --  *   *    10.9.0.0/16    0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
    0     0 SNAT    all  --  *   *    10.10.0.0/16   0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
    0     0 SNAT    all  --  *   *    10.11.0.0/16   0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
    0     0 SNAT    all  --  *   *    10.14.0.0/16   0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
    0     0 SNAT    all  --  *   *    10.15.0.0/16   0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
    0     0 SNAT    all  --  *   *    10.16.0.0/16   0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
    0     0 SNAT    all  --  *   *    10.17.0.0/16   0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
    0     0 SNAT    all  --  *   *    10.21.0.0/16   0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
    0     0 SNAT    all  --  *   *    10.22.0.0/16   0.0.0.0/0    to:46.19.140.196-46.19.140.222
 persistent
```

Because FORWARD rules block such attempts, remediation of this issue can be considered a mitigation only.

Impact:

Source NAT rules may apply to outgoing traffic from VPN clients.

Recommendation:

- Specify the outgoing interface in iptables NAT rules.

Update :

The addition of a default FORWARD policy DROP now prevents client traffic from being forwarded to unwanted interfaces. Openly configured SNAT rules ensure that client IP addresses cannot leak, even if a forward rule is misconfigured. That is an appropriate design decision, although the firewall rules could be more explicit, for instance dropping packets instead of widely applying SNAT and relying on correct FORWARD rules.

4.16 MLL-027 — Reverse path filter in loose mode has no effect due to default routes

Vulnerability ID: MLL-027 **Status:** Unresolved

Vulnerability type: Misconfiguration

Threat level: Low

Description:

Loose reverse path forwarding was enabled on all interfaces, but without directionality has no effect due to default routes.

Technical description:

Reverse path filtering was found to be enabled in loose mode (2) on all interfaces, but has no effect because of the default routes, where a packet is only dropped if there is no route at all.

```
root@se-sto-wg-099:~# cat /proc/sys/net/ipv4/conf/*/rp_filter
2
2
2
2
2
2
2
2
```

```
2
2
2
2
```

```
root@se-sto-wg-099:~# ip r
default via 185.195.233.65 dev eth2 proto static
10.64.0.0/10 dev wg0 proto kernel scope link src 10.64.0.1
10.124.0.0/16 dev wg-multihop proto kernel scope link src 10.124.0.84
10.128.0.0/10 dev wg0 proto kernel scope link src 10.128.0.1
185.195.233.64/26 dev eth2 proto kernel scope link src 185.195.233.75
185.195.233.128/25 dev eth2 proto kernel scope link src 185.195.233.252
```

See also:

- https://datatracker.ietf.org/doc/html/rfc3704#section-2.4
- https://sysctl-explorer.net/net/ipv4/rp_filter/

Impact:

Reverse path filter rules have no effect due to the presence of default routes on the edge node.

Recommendation:

- Use RPF strict mode (`rp_filter=1`).

4.17 MLL-029 — Missing binary hardening on tcp2udp and blocklist-service

Vulnerability ID: MLL-029 **Status:** Unresolved

Vulnerability type: Missing Hardening

Threat level: Low

Description:

The tcp2udp and blocklist-service binaries, which are self-maintained by Mullvad, are missing binary hardening.

Technical description:

The tool checksec was used to validate the binary hardening of programs installed on the system. The binaries provided by Mullvad contain symbols, which allows attacker to reverse engineer their functionality more easily. Given that tcp2udp

is published on GitHub, and security by obscurity is seldom a good idea, this is not hugely problematic. However, the use of stack canaries and RELRO provide meaningful obstacles to exploitation. Fortify can help to detect and mitigate buffer overflows, and should also be enabled.

```
checksec  --file=/usr/local/bin/tcp2udp
RELRO           STACK CANARY      NX            PIE             RPATH      RUNPATH      Symbols
    FORTIFY Fortified       Fortifiable     FILE
Full RELRO      No canary found   NX enabled    PIE enabled     No RPATH   No RUNPATH   5063 Symbols
      No    0               0               /usr/local/bin/tcp2udp
```

```
checksec --file=/usr/bin/blocklist-service
RELRO           STACK CANARY      NX            PIE             RPATH      RUNPATH      Symbols
    FORTIFYFortified       Fortifiable     FILE
Partial RELRO   No canary found   NX enabled    PIE enabled     No RPATH   No RUNPATH   7122 Symbols
      No      0              0               /usr/bin/blocklist-service
```

Impact:

An attacker can exploit the unhardened binaries more easily. However, given that tcp2udp is written in Rust, the real world impact of this could be limited.

Recommendation:

- Ensure the build process includes the build flags for hardening, such as `-Z stack-protector`, and that in release builds which are deployed to servers, that `strip = true` is set.

Update :

During re-testing stack canaries have been confirmed to be missing:

```
gronke@se-sto-wg-099:~/$ git clone https://github.com/slimm609/checksec.sh
Cloning into 'checksec.sh'...
remote: Enumerating objects: 1627, done.
remote: Counting objects: 100% (215/215), done.
remote: Compressing objects: 100% (65/65), done.
remote: Total 1627 (delta 169), reused 150 (delta 150), pack-reused 1412
Receiving objects: 100% (1627/1627), 883.07 KiB | 7.96 MiB/s, done.
Resolving deltas: 100% (863/863), done.
gronke@se-sto-wg-099:~/ cd checksec.sh/
gronke@se-sto-wg-099:~/checksec.sh$ git rev-parse HEAD
90e1d281bbf49424760e9a91798324747462b5b6
gronke@se-sto-wg-099:~/checksec.sh$ ./checksec --version
checksec v2.6.0, Brian Davis, github.com/slimm609/checksec.sh, Dec 2015
Based off checksec v1.5, Tobias Klein, www.trapkit.de, November 2011
```

```
gronke@se-sto-wg-099:~/checksec.sh$ sha256sum /usr/local/bin/tcp2udp
0b11a6f99aa5b6cd82d8f6fb71d9e76a4dc09769f7c456bd0cd52f789de93579  /usr/local/bin/tcp2udp
gronke@se-sto-wg-099:~/checksec.sh$ sha256sum /usr/bin/blocklist-service
8ecf23526a52f33e803dcd053697689dc88912e459dc7cd6eecbc43b7d7e52db  /usr/bin/blocklist-service
```

```
gronke@se-sto-wg-099:~/checksec.sh$ ./checksec  --file=/usr/local/bin/tcp2udp
RELRO           STACK CANARY      NX            PIE             RPATH      RUNPATH  Symbols
 FORTIFY Fortified   Fortifiable FILE
Full RELRO      No canary found   NX enabled    PIE enabled     No RPATH   No RUNPATH   5063 Symbols
      No    0       0       /usr/local/bin/tcp2udp
gronke@se-sto-wg-099:~/checksec.sh$ ./checksec  --file=/usr/bin/blocklist-service
RELRO           STACK CANARY      NX            PIE             RPATH      RUNPATH  Symbols
 FORTIFY Fortified   Fortifiable FILE
Partial RELRO   No canary found   NX enabled    PIE enabled     No RPATH   No RUNPATH   7122 Symbols
      No    0       0       /usr/bin/blocklist-service
```

4.18 MLL-030 — No pre-shared key on wg-internal interface

Vulnerability ID: MLL-030 **Status:** Unresolved

Vulnerability type: Missing Hardening

Threat level: Low

Description:

Internal WireGuard connections do not use a pre-shared key as an additional security layer to defend against cryptographic attacks.

Technical description:

A pre-shared key (PSK) in WireGuard is an optional feature designed to provide an additional layer of symmetric-key cryptography on top of the standard public-key cryptography for extra security. This provides an extra line of defense even in the unlikely event that the primary encryption mechanism is compromised and provides hardening against cryptographic attacks using quantum computers.

According to the protocol specification https://www.wireguard.com/protocol/ WireGuard defaults to an all-zero string of 32 bytes when no custom PSK is used.

```
[Interface]
Address=fc00:2000::185:[redacted]:75/64
ListenPort=52000
PrivateKey=[redacted]

# se-got-mon-121.infra.mullvad.net
[Peer]
PublicKey=AzIfHWBJEWOwoWXPWFQ8yAkHMueiN9915jNIwiFMrVc=
Endpoint=185.[redacted].142:52000
AllowedIPs=10.[redacted].101
AllowedIPs=fc00:2000::185:[redacted]:142

# se-got-librenms-132.mullvad.net
[Peer]
PublicKey=DRobtYBd1AAnLhofAFifSQYg4tzrn6mcpnYE8BeEMF0=
Endpoint=45.[redacted].221:52000
```

```
AllowedIPs=fc00:2000::45:[redacted]:221

# ns0-internal.mullvad.net
[Peer]
PublicKey=7EEbIt15GeRqqSo/vemWIi15wN0n8Rpzbor2TCVAOAc=
Endpoint=185.[redacted].72:52000
AllowedIPs=fc00:2000::185:[redacted]:72
```

Impact:

WireGuard connections without a pre-shared key do not benefit from an additional security layer to harden against potential flaws in the encryption mechanism.

Recommendation:

- Set WireGuard pre-shared keys for additional hardening.

Update :

We did not find a pre-shared key during a retest. At the time of writing, no vulnerabilities are known in the encryption mechanisms in use, so this finding has no direct security impact. Following a security-in-depth approach, we still recommend making use of this method to mitigate potential future flaws in the encryption mechanism.

4.19 MLL-037 — DNS hijacking on VPN clients

Vulnerability ID: MLL-037	**Status:** Resolved
Vulnerability type: DNS Hijacking	
Threat level: Info	

Description:

When using DNS blocklisting on the VPN server, firewall NAT rules apply to any connection to UDP port 53. Consequently, all outgoing DNS requests are hijacked and forced to use the VPN server's local DNS resolver.

Technical description:

Users of DNS blocking services are differentiated by source IP address, so that users can opt-in by altering the client VPN configuration. Firewall NAT rules however were found to hijack any DNS request, regardless of the client using the default DNS server:

```
iptables-save
[...]
-A PREROUTING -s 10.7.0.0/16 -p udp -m udp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.7.0.1
-A PREROUTING -s 10.7.0.0/16 -p tcp -m tcp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.7.0.1
-A PREROUTING -s 10.6.0.0/16 -p udp -m udp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.6.0.1
-A PREROUTING -s 10.6.0.0/16 -p tcp -m tcp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.6.0.1
-A PREROUTING -s 10.5.0.0/16 -p udp -m udp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.5.0.1
-A PREROUTING -s 10.5.0.0/16 -p tcp -m tcp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.5.0.1
-A PREROUTING -s 10.8.0.0/16 -p udp -m udp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.8.0.1
-A PREROUTING -s 10.8.0.0/16 -p tcp -m tcp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.8.0.1
-A PREROUTING -s 10.9.0.0/16 -p udp -m udp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.9.0.1
-A PREROUTING -s 10.9.0.0/16 -p tcp -m tcp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.9.0.1
-A PREROUTING -s 10.10.0.0/16 -p udp -m udp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.10.0.1
-A PREROUTING -s 10.10.0.0/16 -p tcp -m tcp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.10.0.1
-A PREROUTING -s 10.11.0.0/16 -p udp -m udp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.11.0.1
-A PREROUTING -s 10.11.0.0/16 -p tcp -m tcp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.11.0.1
-A PREROUTING -s 10.14.0.0/16 -p udp -m udp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.14.0.1
-A PREROUTING -s 10.14.0.0/16 -p tcp -m tcp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.14.0.1
-A PREROUTING -s 10.15.0.0/16 -p udp -m udp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.15.0.1
-A PREROUTING -s 10.15.0.0/16 -p tcp -m tcp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.15.0.1
-A PREROUTING -s 10.16.0.0/16 -p udp -m udp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.16.0.1
-A PREROUTING -s 10.16.0.0/16 -p tcp -m tcp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.16.0.1
-A PREROUTING -s 10.17.0.0/16 -p udp -m udp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.17.0.1
-A PREROUTING -s 10.17.0.0/16 -p tcp -m tcp --dport 53 -m comment --comment "ansible[dns_hijack]" -j
 DNAT --to-destination 10.17.0.1
```

We confirmed this finding with netcat by trying to query a specific DNS server directly. According to the above firewall rules, those requests are routed to the VPN server's local DNS daemon instead. Mullvad explicitly uses this as a security feature, to minimize the likelihood of DNS leaks. However, users can opt out of this feature, as documented in the FAQ.

Impact:

Clients using DNS blocklists can no longer issue DNS queries to DNS servers of their choice, because all DNS requests are redirected (by an insufficiently specific DNAT rule) to the local resolver on the VPN server.

Recommendation:

- Redirect DNS traffic (UDP port 53) only for DNS servers in private network ranges, so that clients can still connect to custom DNS servers.

Update :

DNS hijacking is used as a measure to prevent DNS leaks. Users who intentionally want to use a specific DNS server find OpenVPN ports without DNS hijacking documented on https://mullvad.net/en/help/faq/ MLL-039 (page 42). After consideration, we have altered the status of this finding to information only.

4.20 MLL-039 — Non-specific API passwords for blocklist-service

Vulnerability ID: MLL-039	**Status:** Resolved
Vulnerability type: Shared Credentials	
Threat level: Low	

Description:

The blocklist service shares the same credentials across all VPN servers.

Technical description:

The blocklist service is used by the VPN servers to add address lists to IPsec, which are then blocked by the firewall.

The environment files for each server class (OpenVPN, WireGuard) share the same `BLOCK_SERVICE_PASSWORD`.

`se-sto-wg-099.relays.mullvad.net`

```
# cat /etc/default/blocklist-service
BLOCK_SERVICE_HOSTNAME=se-sto-wg-099
BLOCK_SERVICE_INTERFACE=bond0
BLOCK_SERVICE_URL=https://api.mullvad.net
BLOCK_SERVICE_USERNAME=wireguard
BLOCK_SERVICE_PASSWORD=[REDACTED]
BLOCK_SERVICE_SET=BLOCKED_NETWORKS_IPV4
BLOCK_SERVICE_PORTS=25,137,138,139,445,1900,2869
```

```
BLOCK_SERVICE_PRIVATE_IPV4=10.0.0.0/8,172.16.0.0/12,192.168.0.0/16
BLOCK_SERVICE_PRIVATE_IPV6=fc00::/7
```

`ch-zrh-ovpn-299.relays.mullvad.net`

```
# cat /etc/default/blocklist-service
BLOCK_SERVICE_HOSTNAME=ch-zrh-ovpn-299
BLOCK_SERVICE_INTERFACE=eth2
BLOCK_SERVICE_URL=https://api.mullvad.net
BLOCK_SERVICE_USERNAME=relays
BLOCK_SERVICE_PASSWORD=[REDACTED]
BLOCK_SERVICE_SET=BLOCKED_NETWORKS_IPV4
BLOCK_SERVICE_PORTS=25,137,138,139,445,1900,2869
BLOCK_SERVICE_PRIVATE_IPV4=10.0.0.0/8,172.16.0.0/12,192.168.0.0/16
BLOCK_SERVICE_PRIVATE_IPV6=fc00::/7
```

Impact:

An attacker with access to a VPN server can use the credentials to interact with the private API paths. Revocation of the credentials is non-trivial due to their shared nature.

Recommendation:

- Ensure that unique credentials are generated per server.
- Consider using client certificates to authenticate towards internal services.

4.21 MLL-043 — Blocklist-service is listening globally

Vulnerability ID: MLL-043

Vulnerability type: Missing Hardening

Threat level: Info

Description:

Blocklist-service is listening on 0.0.0.0, but it is not allowlisted in the firewall configuration.

Technical description:

Blocklist-service listens to all network interfaces on UDP port `35964`:

```
# ss -tulpen | grep blocklist
```

```
udp   UNCONN 0    0    *:35964    *:*    users:(("blocklist-servi",pid=14754,fd=3))
```

However it's not allowlisted in the firewall configuration, so the service is not exposed to the external interface:

```
# iptables -L INPUT | head -n1
Chain INPUT (policy DROP)
# iptables -L -v -n | grep 35964 | wc -l
0
```

Impact:

In case of a firewall misconfiguration, blocklist-service could be exposed on all interfaces, including the public network uplink. However, on the tested VPN relay servers, the firewall configuration was found to block access from unintended source.

Recommendation:

- Bind blocklist-service to a specific interface to prevent public exposure in case of a firewall misconfiguration.

4.22 MLL-047 — Sudo without password

Vulnerability ID: MLL-047 **Status:** Unresolved

Vulnerability type: Missing Hardening

Threat level: Low

Description:

Sudoers on VPN servers do not require password authentication, so a compromised SSH key could allow immediate access to privileged accounts.

Technical description:

The sudoers configuration allows members of the `sudo` group to become `root` without further authentication:

```
$ sudo cat /etc/sudoers | grep NOPASS
%sudo  ALL=(ALL) NOPASSWD:ALL
```

Admin IPs are allowlisted, so specific jumpboxes have to be used in order to connect to a VPN server. These jumpboxes were not in scope for this project.

Impact:

A leaked SSH key or misconfiguration of SSH agent forwarding as in MLL-007 (page 33) could result in root access to other servers by an attacker.

Recommendation:

- Consider requiring further authentication (password, OTP) by administrator accounts before allowing them to become root on VPN servers.

Update :

Sudo without password is implemented by design. Mullvad is considering disabling remote access to production machines entirely, so this finding would only affect testing systems in debug mode.

4.23 MLL-021 — Shared snmp credentials across servers

Vulnerability ID: MLL-021 **Status:** Resolved

Vulnerability type: Shared Credentials

Threat level: Low

Description:

SNMP daemon credentials are encrypted but are re-used across servers.

Technical description:

```
root@ch-zrh-ovpn-299# cat /etc/snmp/snmpd.conf
agentAddress udp6:[fc00:2000::46:19:140:194]:161
createUser librenms SHA-256 REDACTED AES128 REDACTED
rouser librenms priv 1.3.6.1.2.1
```

```
root@se-sto-wg-099# cat /etc/snmp/snmpd.conf
agentAddress udp6:[fc00:2000::185:195:233:75]:161
createUser librenms SHA-256 REDACTED AES128 REDACTED
rouser librenms priv 1.3.6.1.2.1
```

Impact:

A compromise of one server's SNMP credentials will affect all other servers as well.

Recommendation:

- Use unique SNMP credentials per server.

Update :

During a retest, we found that SNMP credentials on the two test-servers are now different.

4.24 MLL-018 — Slack API key shared across servers

Vulnerability ID: MLL-018 **Status:** Unresolved

Vulnerability type: Shared Credentials

Threat level: Low

Description:

Slack notification credentials can be used to push misleading messages to the alerting Slack channel. Credentials are re-used across VPN servers, so that identifying the origin for revocation may be hindered.

Technical description:

```
root@se-sto-wg-099:~# systemctl cat mullvad-alert-slack-high@mullvad-check-read-only-
disks.service.service
# /etc/systemd/system/mullvad-alert-slack-high@.service
[Unit]
Description=Sends slack alerts of high priority
After=network.target

[Service]
ExecStart=/usr/bin/env curl --silent -X POST -H 'Content-type: application/json' \
--data '{"text":"%i @ %H has failed"}' https://hooks.slack.com/services/[REDACTED]/[REDACTED]/
nvAH[REDACTED]
```

POST requests to the Slack hook endpoint allows adversaries to post arbitrary messages (that are indistinguishable from legitimate ones) to appear in the alerting channel:

```
% curl --silent -X POST -H 'Content-type: application/json' \
```

```
--data '{"text":"ROS was here"}' https://hooks.slack.com/services/[REDACTED]/[REDACTED]/
nvAH[REDACTED]
```

Impact:

Adversaries who obtain access to the shared Slack notification credentials may pollute the alerting Slack channel with misleading messages. Use of shared credentials affects the ability to identify the origin of such messages and can hinder revocation of compromised credentials.

Recommendation:

- Use unique Slack notification credentials per VPN server.
- Consider using a pull approach instead of allowing VPN servers to push unfiltered alert messages.

4.25 MLL-016 — Telegraf password shared across servers

Vulnerability ID: MLL-016 **Status:** Unresolved

Vulnerability type: Shared Credentials

Threat level: Low

Description:

Telegraf credentials are re-used across VPN servers.

Technical description:

From `/etc/telegraf/telegraf.conf`:

```
"influx": {
  "host": "{{ hostvars[groups.monitoring[0]].internal_overlay_hostname }}",
  "port": 8086,
  "database": "mullvad",
  "username": "telegraf",
  "password": "NBGjfxq[REDACTED]"
},
```

The credentials can also be found in `/home/mad/telegraf/telegraf.conf` and the Ansible inventory as well. Neither of the files is accessible by unprivileged system accounts.

Impact:

Shared Influx database credentials used by Telegraf across VPN servers allows manipulation of global server metrics, such as CPU and disk usage or network metrics.

Recommendation:

- Use unique credentials per VPN server appliance.

5 Non-Findings

In this section we list some of the things that were tried but turned out to be dead ends.

5.1 NF-006 — Bind DNS server is up to date

Bind 9.18.12 (DNS) has no known security vulnerabilities or outstanding security updates:

```
root@ch-zrh-ovpn-299# named -v
BIND 9.18.12-0ubuntu0.22.04.1-Ubuntu (Extended Support Version) <id:>
```

5.2 NF-023 — Influxdb user is write-only

The `telegraf` Influx DB account that we noted uses a shared password in MLL-016 (page 47) is write-only, so it is not possible to access database content using that account:

```
root@se-sto-wg-099# influx -host 'fc00:2000::185:213:154:142' -port 8086 -username 'telegraf' -
password [CENSORED] -database 'telegraf' -ssl -unsafeSsl
password:
Connected to https://[fc00:2000::185:213:154:142]:8086 version 1.8.10
InfluxDB shell version: 1.6.7~rc0
> SHOW MEASUREMENTS
ERR: error authorizing query: telegraf not authorized to execute statement 'SHOW MEASUREMENTS',
 requires READ on telegraf
> SHOW DATABASES
name: databases
name
----
mullvad
> SHOW QUERIES
ERR: error authorizing query: telegraf not authorized to execute statement 'SHOW QUERIES', requires
 READ on telegraf
>
```

5.3 NF-025 — danted does not appear to write log files

Manual inspection of the Dante proxy server configuration indicates the absence of logging of VPN traffic, in line with Mullvad's privacy objectives. We verified this by observing the processes involved using strace:

```
root@se-sto-wg-099# strace -Tfe trace=open,close,read,write,truncate -p 15723 -p 15729 -p 2267615 -p
 2267617 -p 2925249 -p 2925250 -p 2925251 -p 2925252 -p 2925253 -p 2925254 -p 2925255 -p 2925256 -p
 2925257 -p 292
5258 -p 2925259 -p 2925260 -p 2925261 -p 2925262 -p 2925263 -p 2925264 -p 2925265 -p 2925266 -p
 2925267 -p 2925268 -p 2925430 -p 2925432 -p 2925434 -p 2925435 -p 2925436 -p 2925437 -p 2925438 -p
 2925446 -p 2925450 -p 292546
0 -p 2925461 -p 2925462 -p 2925465 -p 2925466 -p 2925467 -p 2925468 -p 2925470 -p 2925474
[...]
```

```
[pid 15723] close(7)                    = 0 <0.000006>

             [659/1928]
[pid 2925637] close(0 <unfinished ...>
[pid 15723] close(12 <unfinished ...>
[pid 2925637] <... close resumed>)      = 0 <0.000009>
[pid 15723] <... close resumed>)        = 0 <0.000028>
[pid 15723] close(13)                   = 0 <0.000021>
[pid 15723] close(14)                   = 0 <0.000069>
[pid 2925637] close(0 <unfinished ...>
[pid 15723] close(15 <unfinished ...>
[pid 2925637] <... close resumed>)      = 0 <0.000007>
[pid 15723] <... close resumed>)        = 0 <0.000010>
[pid 2925637] close(7 <unfinished ...>
[pid 15723] close(16 <unfinished ...>
[pid 2925637] <... close resumed>)      = 0 <0.000009>
[pid 15723] <... close resumed>)        = 0 <0.000010>
[pid 2925637] close(10)                 = 0 <0.000005>
[pid 2925637] close(12)                 = 0 <0.000006>
[pid 2925637] close(13)                 = 0 <0.000008>
[pid 2925637] close(14)                 = 0 <0.000027>
[pid 2925637] close(15)                 = 0 <0.000012>
[pid 2925637] close(16)                 = 0 <0.000068>
[pid 2925637] close(17)                 = 0 <0.000071>
[pid 2925637] close(18)                 = 0 <0.000005>
[pid 2925637] close(19)                 = 0 <0.000021>
[pid 2925637] close(20)                 = 0 <0.000005>
[pid 2925637] close(21)                 = 0 <0.000006>
[pid 2925637] close(22)                 = 0 <0.000006>
[pid 2925637] close(23)                 = 0 <0.000006>
[pid 2925637] close(24)                 = 0 <0.000005>
[pid 2925637] close(25)                 = 0 <0.000005>
[pid 2925637] close(26)                 = 0 <0.000005>
[pid 2925637] close(27)                 = 0 <0.000005>
[pid 2925637] close(28)                 = 0 <0.000005>
[pid 2925637] close(29)                 = 0 <0.000063>
[pid 2925637] close(30)                 = 0 <0.000005>
[pid 2925637] close(31)                 = 0 <0.000005>
[pid 2925637] close(32)                 = 0 <0.000005>
[pid 2925637] close(33)                 = 0 <0.000005>
[pid 2925637] close(34)                 = 0 <0.000006>
[pid 2925637] close(35)                 = 0 <0.000005>
[pid 2925637] close(36)                 = 0 <0.000005>
[pid 2925637] close(37)                 = 0 <0.000005>
[pid 2925637] close(38)                 = 0 <0.000005>
[pid 2925637] close(39)                 = 0 <0.000005>
[pid 2925637] close(40)                 = 0 <0.000005>
[pid 2925637] close(41)                 = 0 <0.000005>
[pid 2925637] close(42)                 = 0 <0.000005>
[pid 2925637] close(43)                 = 0 <0.000005>
[pid 2925637] close(44)                 = 0 <0.000005>
[pid 2925637] close(45)                 = 0 <0.000005>
[pid 2925637] close(46)                 = 0 <0.000005>
[pid 2925637] close(47)                 = 0 <0.000005>
[pid 2925637] close(48)                 = 0 <0.000005>
[...]
```

```
root@se-sto-wg-099# head -n 8 /etc/dante.conf  /etc/dante_multihop.conf
==> /etc/dante.conf <==
# This configuration will only work with dante 1.4.x and (possibly) newer versions
```

```
# For an explanation of these options please see the man page and
# https://www.inet.no/dante/doc/1.4.x/config/server.html

# Do not log anything
errorlog: /dev/null
logoutput: /dev/null
debug: 0

==> /etc/dante_multihop.conf <==
# This configuration will only work with dante 1.4.x and (possibly) newer versions
# For an explanation of these options please see the man page and
# https://www.inet.no/dante/doc/1.4.x/config/server.html

# Do not log anything
errorlog: /dev/null
logoutput: /dev/null
debug: 0
```

5.4 NF-026 — Bind DNS server does not log queries

DNS resolvers on the VPN servers are configured to not log queries, in line with Mullvad's privacy objectives:

From `/etc/bind/named.conf.logging`:

```
logging {
     # This is the bind default in ubuntu
     # See https://help.ubuntu.com/community/BIND9ServerHowto#Logging
     # category default { default_syslog; default_debug; };
     # No logging
     category default { null; };
     category unmatched { null; };
};
```

```
root@se-sto-wg-099# named-checkconf  -p /etc/bind/named.conf |grep -A 7 logging
logging {
        category "default" {
                "null";
        };
        category "unmatched" {
                "null";
        };
};
```

This assumption was verified by inspecting running processes for file operations using strace:

```
root@se-sto-wg-099:# strace -Tfe trace=open,close,read,write,truncate -p 12683
strace: Process 12683 attached with 14 threads
[pid 12690] write(15, "\1\0\0\0\0\0\0\0", 8) = 8 <0.000007>
[pid 12685] read(15, "\1\0\0\0\0\0\0\0", 1024) = 8 <0.000077>
[pid 12685] write(15, "\1\0\0\0\0\0\0\0", 8) = 8 <0.000021>
[pid 12685] read(15, "\1\0\0\0\0\0\0\0", 1024) = 8 <0.000008>
[pid 12685] write(15, "\1\0\0\0\0\0\0\0", 8) = 8 <0.000111>
[pid 12685] close(6)                    = 0 <0.000072>
[pid 12685] read(15, "\1\0\0\0\0\0\0\0", 1024) = 8 <0.000021>
[pid 12685] write(15, "\1\0\0\0\0\0\0\0", 8) = 8 <0.000006>
[pid 12685] read(15, "\1\0\0\0\0\0\0\0", 1024) = 8 <0.000006>
```

```
[pid 12690] write(31, "\1\0\0\0\0\0\0\0", 8) = 8 <0.000007>
[pid 12689] read(31, "\1\0\0\0\0\0\0\0", 1024) = 8 <0.000091>
[pid 12689] write(31, "\1\0\0\0\0\0\0\0", 8) = 8 <0.000021>
[pid 12689] read(31, "\1\0\0\0\0\0\0\0", 1024) = 8 <0.000010>
[pid 12690] write(11, "\1\0\0\0\0\0\0\0", 8) = 8 <0.000007>
[pid 12684] read(11, "\1\0\0\0\0\0\0\0", 1024) = 8 <0.000063>
[pid 12684] close(6)                    = 0 <0.000027>
[pid 12684] write(11, "\1\0\0\0\0\0\0\0", 8) = 8 <0.000023>
[pid 12684] read(11, "\1\0\0\0\0\0\0\0", 1024) = 8 <0.000005>
[pid 12690] write(15, "\1\0\0\0\0\0\0\0", 8) = 8 <0.000052>
[pid 12685] read(15, "\1\0\0\0\0\0\0\0", 1024) = 8 <0.000114>
```

6 Future Work

- **Retest of findings**
  When mitigations for the vulnerabilities described in this report have been deployed, a repeat test should be performed to ensure that they are effective and have not introduced other security problems.
- **Regular security assessments**
  Security is an ongoing process and not a product, so we advise undertaking regular security assessments and penetration tests, ideally prior to every major release or every quarter.
- **Code audit of blocklist-service**
  The blocklist-service was running on the Mullvad VPN servers as an ELF 64-bit binary. This pentest covered the runtime environment, but time constraints did not permit binary analysis of the daemon. Because of the service's network exposure we recommend conducting a code audit of this critical component.
- **Detailed review of the boot process**
  This pentest was conducted on two booted and provisioned VPN servers without any physical disk mounts. A detailed review of the boot process was not part of this engagement, but is essential to verify the secure operation of diskless hardware servers, and is therefore strongly recommended for future investigation.
- **Code audit of wg-manager**
  Conduct a code audit of the WireGuard manager source code in order to identify issues which could not be identified during this gray-box test.

7 Conclusion

We discovered 1 High, 6 Elevated, 4 Moderate, 10 Low and 4 Info-severity issues during this penetration test.

The Mullvad VPN relays which were the subject of this test showed a mature architecture, with the administrators demonstrating a high degree of experience and motivation to tackle the issues we found. The memory-only design mitigates multiple attack vectors, and also at least minimizes the chances of persistent attacks. However, due to the use of IPMI modules, some persistence could be obtained.

During the test we found no logging of user activity data, however we have no way of validating whether the real production machines are set up in the same way, due to lack of secure boot, remote attestation, and the ability of administrators to log into live production systems and tamper with them. The absence of remote audit logging is especially troublesome, as one compromised administrator can impact the entire organisation, without any audit trail. We would recommend a model were booted systems are immutable, are running a minimized deployment using network namespaces and isolated services, without allowing any administrative access except when they are rebooted into a debug state without production traffic. However, we are aware of the operational hurdles this creates.

During the test period we noticed production traffic on our testing machines, which exposed actual user traffic patterns through the wg-multihop feature. This oversight concerned us.

Issues relating to missing system hardening and the shared credentials we observed were addressed or revoked during the testing period, but we did not find it possible to use these shared credentials for lateral movement anyway. We did not identify any remote code execution vulnerabilities in the services either. Further clarity could be obtained with a crystal-box test of the exposed components.

We'd like to wrap up this report up by thanking the Mullvad team for their excellent co-operation and communication, and insightful conference calls and chats.

We recommend fixing all of the issues found and then performing a retest in order to ensure that mitigations are effective and that no new vulnerabilities have been introduced.

Finally, we want to emphasize that security is a process – this penetration test is just a one-time snapshot. Security posture must be continuously evaluated and improved. Regular audits and ongoing improvements are essential in order to maintain control of your corporate information security. We hope that this pentest report (and the detailed explanations of our findings) will contribute meaningfully towards that end.

Please don't hesitate to let us know if you have any further questions, or need further clarification on anything in this report.

Appendix 1 Testing team

Stefan Grönke	Stefan is a highly adaptable senior security consultant, pentester and code auditor. He has over a decade of experience in (reverse) engineering, architecture and quality assurance, with a large focus on security and simplicity. He commits most of his free time to development projects that enable him and others to run secure infrastructure. As a full-stack developer he has always enjoyed learning from and with open source code; Stefan has contributed to a variety of projects, often on GitHub. Stefan can be a terrible chaos monkey in the ROS infra, but always cleans up behind him. In fact he likes constructing more than disruption. Therefore he went over from setting things on fire to participating in the ROS development and infra team. Apart from that he enjoys speaking at conferences like the Chaos Communication Congress or hosting workshops at local hackerspaces. He was one of the winning participants of team proTRon at the Shell Eco Contest in 2013/14 for building a CAN-Bus based telemetry system for a lightweight fuel-cell driven car.
Melanie Rieback	Melanie Rieback is a former Asst. Prof. of Computer Science from the VU, who is also the co-founder/CEO of Radically Open Security.