Test Report #: **j978x2x8mxan976c02ndp6m2w581bg58**

Mobile App Profile Developer Test Report

Leviathan Security Group

Platform	Android
Application Name	Mullvad VPN
Application Version	2026.2
Company	Mullvad VPN AB
Mobile App Specification Version	v1.0
Assurance Level	AL2
Summary	REMEDIATION REQUIRED
Approved by (name / position & signature)	Viktor Sytnik, Technical director
Date of Evaluation	February 18, 2026

Table of Contents

Application and Developer Info....8
Testing Verdicts....8
Assurance Levels....9
Evaluation Results....10
1.1 Storage....10
1.1.1 The app securely stores sensitive data in external storage....10
1.1.1.1 The app shall securely store sensitive data in external storage....10
Verification Evidence....10
Verification Result....10
1.1.2 The app prevents leakage of sensitive data....10
1.1.2.1 The Keyboard Cache Is Disabled for sensitive data inputs....10
Verification Evidence....10
Verification Result....11
1.1.2.2 No sensitive data is stored in system logs....12
Verification Evidence....12
Verification Result....13
1.2 Crypto....14
1.2.1 The app employs current strong cryptography and uses it according to industry best practices....14
1.2.1.1 No insecure random number generators shall be utilized for any security sensitive context....14
Verification Evidence....14
Verification Result....15
1.2.1.2 No insecure operations shall be used for symmetric cryptography....16
Verification Evidence....16
Verification Result....19
1.2.1.3 Strong cryptography shall be implemented according to industry best practices....19
Verification Evidence....19
Verification Result....21
1.2.2 The app performs key management according to industry best practices....22
1.2.2.1 Cryptographic keys shall only be used for their defined purpose....22
Verification Evidence....22
Verification Result....23

1.2.2.2 Cryptographic key management shall be implemented properly 23
Verification Evidence 23
Verification Result 25
1.3 Auth 25
1.3.1 The app uses secure authentication and authorization protocols and follows the relevant best practices 25
1.3.1.1 If using OAuth 2.0 for authorization, or if using OpenID Connect for authentication, Proof Key for Code Exchange (PKCE) shall be implemented to protect the code grant 25
Verification Evidence 25
Verification Result 26
1.4 Network 26
1.4.1 The app secures all network traffic according to the current best practices 26
1.4.1.1 Network connections shall be encrypted 26
Verification Evidence 26
Verification Result 27
1.4.1.2 TLS configuration of network connections shall adhere to industry best practices 28
Verification Evidence 28
Verification Result 28
1.4.1.3 Endpoint identity shall be verified on network connections 28
Verification Evidence 28
Verification Result 29
1.5 Platform 30
1.5.1 The app uses IPC mechanisms securely 30
1.5.1.1 The app shall limit content provider exposure and harden queries against injection attacks 30
Verification Evidence 30
Verification Result 30
1.5.1.2 The app shall use verified links and sanitize all link input data 30
Verification Evidence 30
Verification Result 30
1.5.1.3 Any sensitive functionality exposed via IPC shall be intentional and at the minimum required level 30
Verification Evidence 30
Verification Result 31
1.5.1.4 All Pending Intents shall be immutable or otherwise justified for mutability 31

Verification Evidence 31
Verification Result 32
1.5.2 The app uses WebViews securely 32
1.5.2.1 WebViews shall securely execute JavaScript 32
Verification Evidence 32
Verification Result 32
1.5.2.2 WebView shall be configured to allow the minimum set of protocol handlers required while disabling potentially dangerous handlers 33
Verification Evidence 33
Verification Result 34
1.5.3 The app uses the user interface securely 34
1.5.3.1 The app shall by default mask data in the User Interface when it is known to be sensitive 34
Verification Evidence 34
Verification Result 35
1.6 Code 35
1.6.1 The app requires an up-to-date platform version 35
1.6.1.1 The app shall set the targetSdkVersion to an up-to-date platform version 35
Verification Evidence 35
Verification Result 36
1.6.2 The app only uses software components without known vulnerabilities 36
1.6.2.1 The app only uses software components without known vulnerabilities 36
Verification Evidence 36
Verification Result 36
1.6.3 The app validates and sanitizes all untrusted inputs 37
1.6.3.1 Compiler security features shall be enabled 37
Verification Evidence 37
Verification Result 37
1.6.3.2 The App shall Mitigate Against Injection Flaws in Content Providers 38
Verification Evidence 38
Verification Result 38
1.6.3.3 Arbitrary URL redirects shall not be included in the app's webviews 38
Verification Evidence 38
Verification Result 38
1.6.3.4 Any use of implicit intents shall be appropriate for the app's functionality and any return data shall be handled securely 39

Verification Evidence 39
Verification Result 39
1.7 Resilience 39
1.7.1 The app implements anti-tampering mechanisms 39
1.7.1.1 The app shall be properly signed 39
Verification Evidence 39
Verification Result 40
1.7.2 The app implements anti-static analysis mechanisms 40
1.7.2.1 The app shall disable all debugging symbols in the production version 40
Verification Evidence 40
Verification Result 40
1.7.3 The app implements anti-dynamic analysis mechanisms 40
1.7.3.1 The app shall not be debuggable if installed from outside of commercial app stores 40
Verification Evidence 40
Verification Result 40
1.8 Privacy 41
1.8.1 The app minimizes access to sensitive data and resources 41
1.8.1.1 The app shall minimize access to sensitive data and resources provided by the platform 41
Verification Evidence 41
Verification Result 43
1.8.2 The app is transparent about data collection and usage 44
1.8.2.1 The app shall be transparent about data collection and usage 44
Verification Evidence 44
Verification Result 44
1.8.3 The app offers user control over their data 45
1.8.3.1 Users shall have the ability to request their data to be deleted via an in-app mechanism 45
Verification Evidence 45
Verification Result 45

Storage		Verdict			
		P	F	NA	INC
1.1.1	The app securely stores sensitive data in external storage	+			
1.1.2	The app prevents leakage of sensitive data	+			

Crypto		Verdict			
		P	F	NA	INC
1.2.1	The app employs current strong cryptography and uses it according to industry best practices	+			
1.2.2	The app performs key management according to industry best practices	+			

Auth		Verdict			
		P	F	NA	INC
1.3.1	The app uses secure authentication and authorization protocols and follows the relevant best practices			+	

Network		Verdict			
		P	F	NA	INC
1.4.1	The app secures all network traffic according to the current best practices	+			

Platform		Verdict			
		P	F	NA	INC
1.5.1	The app uses IPC mechanisms securely		+		
1.5.2	The app uses WebViews securely			+	

1.5.3	The app uses the user interface securely		+		

Code		Verdict			
		P	F	NA	INC
1.6.1	The app requires an up-to-date platform version	+			
1.6.2	The app only uses software components without known vulnerabilities		+		
1.6.3	The app validates and sanitizes all untrusted inputs		+		

Resilience		Verdict			
		P	F	NA	INC
1.7.1	The app implements anti-tampering mechanisms	+			
1.7.2	The app implements anti-static analysis mechanisms	+			
1.7.3	The app implements anti-dynamic analysis mechanisms	+			

Privacy		Verdict			
		P	F	NA	INC
1.8.1	The app minimizes access to sensitive data and resources	+			
1.8.2	The app is transparent about data collection and usage		+		
1.8.3	The app offers user control over their data		+		

Application and Developer Info

Application Name	Mullvad VPN
Application Version	2026.2
Link to Application	https://play.google.com/store/apps/details?id=net.mullvad.mullvadvpn&hl=en
Company Name	Mullvad VPN AB
Company Address	Box 53049, 400 14 Gothenburg, Sweden
Contact name and Email	David Göransson / david.goransson@mullvad.net
Onboarding Questionnaire Filename and Hash	Mullvad VPN AL2 Android Onboarding Questionnaire.xlsx SHA-256: 65f24d938d1bc023cd8a773cef6d04d55536aef745eb13376f814737387daf82

Testing Verdicts

PASS	P
FAIL	F
NA (PASS)	NA
INCONCLUSIVE	INC

Assurance Levels

Level	Description
AL0	The developer reviews applicable ADA requirements and confirms their product meets every requirement, then completes a questionnaire to represent their application's compliance with ADA requirements. The developer guarantees the completeness and accuracy of the questionnaire
AL1	The developer provides evidence and statements of compliance to each audit test case. The ADA approved lab reviews the evidence against the requirements. The ADA approved lab does not directly assess the application.
AL2	The ADA approved lab evaluates each audit test case directly against the application. In some cases, the developer may need to provide limited information or code snippets.

Evaluation Results

1.1 Storage

1.1.1 The app securely stores sensitive data in external storage

1.1.1.1 The app shall securely store sensitive data in external storage

Verification Evidence

After utilizing application functionalities, no data was observed to be stored in the external storage (application's package name is net.mullvad.mullvadvpn).

```
akita:/ # ls /sdcard/Android/data/
com.android.chrome/                 com.google.android.apps.maps/          com.google.android.apps.wear.companion/
com.android.vending/                com.google.android.apps.nbu.files/     com.google.android.gms/
com.google.android.GoogleCamera/    com.google.android.apps.photos/        com.google.android.googlequicksearchbox/
com.google.android.adservices.api/  com.google.android.apps.pixel.dcservice/  com.google.android.videos/
akita:/ # ls /sdcard/Android/data/com.
```

Verification Result

Result	Pass
Comment	

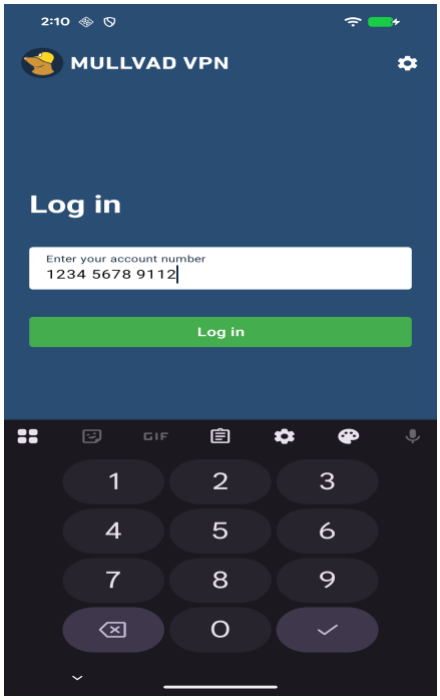
1.1.2 The app prevents leakage of sensitive data

1.1.2.1 The Keyboard Cache Is Disabled for sensitive data inputs

Verification Evidence

The application does not utilize keyboard cache for inputs. The login process only requires a 16-digit ID number, which is not cached.

```
(venv)                -MacBook-Pro sources % grep -R "inputType" -n .
./resources/res/layout/abc_search_view.xml:60:                android:inputType="textNoSuggestions"
./resources/res/layout/ime_secondary_split_test_activity.xml:13:        android:inputType="text"/>
./sources/net/mullvad/talpid/R.java:2662:        public static int SearchView_android_inputType = 0x00000002;
./sources/net/mullvad/talpid/R.java:2798:        public static int[] SearchView = {android.R.attr.focusable, android.R.attr.maxWidth, android.R.attr.inputType, android.R.attr.imeOpti
ons, R.attr.closeIcon, R.attr.commitIcon, R.attr.defaultQueryHint, R.attr.goIcon, R.attr.iconifiedByDefault, R.attr.layout, R.attr.queryBackground, R.attr.queryHint, R.attr.searchHin
tIcon, R.attr.searchIcon, R.attr.submitBackground, R.attr.suggestionRowLayout, R.attr.voiceIcon};
./sources/n/n.java:65:            int inputType = super.getInputType();
./sources/n/n.java:71:            super.setRawInputType(inputType);
./sources/u2/u.java:2123:                    editorInfo.inputType = 1;
./sources/u2/u.java:2125:                    editorInfo.inputType = 1;
./sources/u2/u.java:2128:                    editorInfo.inputType = 2;
./sources/u2/u.java:2130:                    editorInfo.inputType = 3;
./sources/u2/u.java:2132:                    editorInfo.inputType = 17;
./sources/u2/u.java:2134:                    editorInfo.inputType = 33;
./sources/u2/u.java:2136:                    editorInfo.inputType = 129;
./sources/u2/u.java:2138:                    editorInfo.inputType = 18;
./sources/u2/u.java:2143:                    editorInfo.inputType = 8194;
./sources/u2/u.java:2146:                    int i13 = editorInfo.inputType;
./sources/u2/u.java:2148:                        editorInfo.inputType = i13 | 131072;
./sources/u2/u.java:2154:                int i14 = editorInfo.inputType;
./sources/u2/u.java:2158:                        editorInfo.inputType = i14 | 4096;
./sources/u2/u.java:2160:                        editorInfo.inputType = i14 | 8192;
./sources/u2/u.java:2162:                        editorInfo.inputType = i14 | 16384;
./sources/u2/u.java:2165:                        editorInfo.inputType |= 32768;
./sources/r3/g.java:357:        int i14 = editorInfo.inputType & 4095;
./sources/t0/s.java:132:        editorInfo.inputType = i11;
./sources/t0/s.java:134:            editorInfo.inputType = 131072 | i11;
./sources/t0/s.java:139:        int i14 = editorInfo.inputType;
./sources/t0/s.java:143:                editorInfo.inputType = i14 | 4096;
./sources/t0/s.java:145:                editorInfo.inputType = i14 | 8192;
./sources/t0/s.java:147:                editorInfo.inputType = i14 | 16384;
./sources/t0/s.java:150:                editorInfo.inputType |= 32768;
```



Verification Result

Result	Pass
Comment	

1.1.2.2 No sensitive data is stored in system logs

Verification Evidence

Neither static nor dynamic analysis of the application revealed any sensitive data being logged. Furthermore, no hard-coded sensitive data was detected. Additionally, gitleaks revealed 133 occurrences of the same token value. As the application is a VPN, it appears to be non-sensitive, since it is used for configuration of VPN relays. It looks as in the following example (at the bottom):

"public_key": "0qSP0VxoIhEhRK+fAHVvmfRdjPs2DmmpOCNLFP/7cGw=",

"daita": false,

"shadowsocks_extra_addr_in": [

"omitted"

],

"features": {

"daita": null,

"quic": {

"addr_in": [

"omitted"

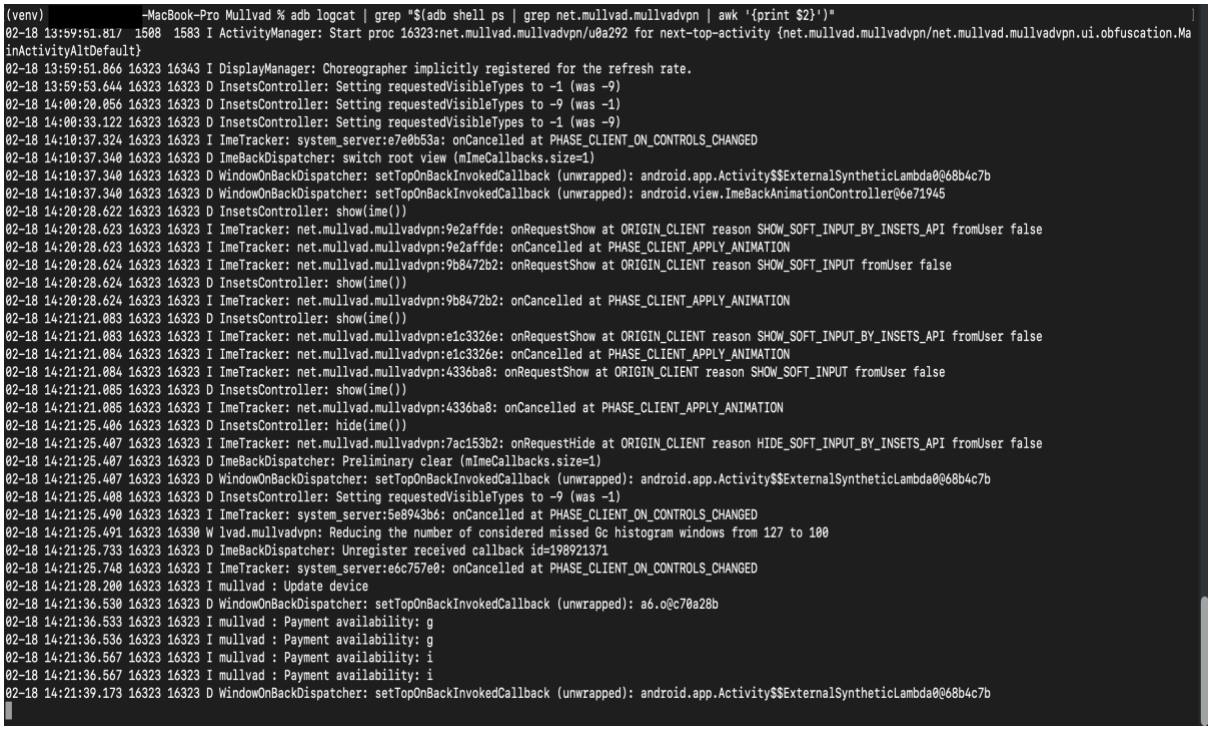
],

"token": "d18a23fc-7d5e-4fd2-8372-5e0e73de741a",

"domain": "de-ber-wg-001.blockerad.eu"

},

"lwo": {}



```
(venv)                -MacBook-Pro Mullvad % grep -iE "password" mullvad_logs.txt
(venv)                -MacBook-Pro Mullvad % grep -iE "^key" mullvad_logs.txt
(venv)                -MacBook-Pro Mullvad %
```

```
                -MacBook-Pro sources % gitleaks dir . -v

    ○
    │╲
    │ ○
    ○ ▒
    ▒    gitleaks

Finding:     "token": "d18a23fc-7d5e-4fd2-8372-5e0e73de741a"
Secret:      d18a23fc-7d5e-4fd2-8372-5e0e73de741a
RuleID:      generic-api-key
Entropy:     3.683542
File:        resources/assets/relays.json
Line:        598
Fingerprint: resources/assets/relays.json:generic-api-key:598
```

```
Finding:     "token": "d18a23fc-7d5e-4fd2-8372-5e0e73de741a"
Secret:      d18a23fc-7d5e-4fd2-8372-5e0e73de741a
RuleID:      generic-api-key
Entropy:     3.683542
File:        resources/assets/relays.json
Line:        16362
Fingerprint: resources/assets/relays.json:generic-api-key:16362

Finding:     } while (!"-----BEGIN PRIVATE KEY-----".equals(line));
        StringBuilder sb2 = new StringBuilder();
       ...                                          ".equals(line2)) {
Secret:      -----BEGIN PRIVATE KEY-----".equals(line));
        StringBuilder sb2 = new StringBuilder();
       ...
RuleID:      private-key
Entropy:     4.489470
File:        sources/ha/a.java
Line:        746
Fingerprint: sources/ha/a.java:private-key:746

10:28AM INF scanned ~31351978 bytes (31.35 MB) in 3.25s
10:28AM WRN leaks found: 134
                -MacBook-Pro sources %
```

Verification Result

Result	Pass
Comment	

1.2 Crypto

1.2.1 The app employs current strong cryptography and uses it according to industry best practices

1.2.1.1 No insecure random number generators shall be utilized for any security sensitive context

Verification Evidence

No unsafe random number generators (RNGs) were identified in any security-related context. We performed manual and dynamic reviews to confirm this.

The libmullvad_jni.so shared object, which the Android application utilizes for cryptographic operations, uses a cryptographically secure RNG.

```
[redacted]-macbook-pro-2 src % grep -Rni "new_from_random("
./device/service.rs:47:        let private_key = PrivateKey::new_from_random();
./device/service.rs:86:        let private_key = PrivateKey::new_from_random();
./device/service.rs:170:        let private_key = PrivateKey::new_from_random();
./device/service.rs:202:        let private_key = PrivateKey::new_from_random();
```

```
pub fn new_from_random() -> Self {
    PrivateKey(x25519_dalek::StaticSecret::random())
}
```

x25519_dalek

Struct StaticSecret

Source

Settings Help Summary

```
pub struct StaticSecret(/* private fields */);
```

Available on **crate feature** `static_secrets` only.

A Diffie-Hellman secret key that can be used to compute multiple `SharedSecret`s.

This type is identical to the `EphemeralSecret` type, except that the `StaticSecret::diffie_hellman` method does not consume the secret key, and the type provides serialization methods to save and load key material. This means that the secret may be used multiple times (but does not *have to be*).

Warning

```rust
/// Generate a new [`StaticSecret`] with the supplied RNG.
pub fn random_from_rng<T: RngCore + CryptoRng>(mut csprng: T) -> Self {
    // The secret key is random bytes. Clamping is done later.
    let mut bytes = [0u8; 32];
    csprng.fill_bytes(&mut bytes);
    StaticSecret(bytes)
}

/// Generate a new [`StaticSecret`].
#[cfg(feature = "getrandom")]
pub fn random() -> Self {
    Self::random_from_rng(rand_core::OsRng)
}
```

Within the Android application source code, the RNG is used only in the test suite at android/test/e2e/src/main/kotlin/net/mullvad/mullvadvpn/test/e2e/router/packetCapture/PacketCapture.kt:9:

```kotlin
@JvmInline
@Serializable(with = PacketCaptureSessionSerializer::class)
value class PacketCaptureSession(val value: UUID = UUID.randomUUID())
```

We performed an automated scan against the Android application codebase using a security rule targeting unsafe random number generation (randomstringutils.yml). No other instances of random were identified:

```
(venv) [redacted]-MacBook-Pro sources % semgrep --config "https://raw.githubusercontent.com/elttam/semgrep-rules/main/rules/java/lang/security/audit/crypto/randomstringutils.yml"

┌─────────────┐
│ Semgrep CLI │
└─────────────┘

Scanning 8345 files (only git-tracked) with 1 Code rule:

  CODE RULES
  Scanning 7911 files.

  SUPPLY CHAIN RULES

  💎 Run `semgrep ci` to find dependency
     vulnerabilities and advanced cross-file findings.

  PROGRESS

  ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 100% 0:00:00


┌──────────────┐
│ Scan Summary │
└──────────────┘
✅ Scan completed successfully.
 • Findings: 0 (0 blocking)
 • Rules run: 1
 • Targets scanned: 7911
 • Parsed lines: ~100.0%
 • Scan skipped:
   ◦ Files larger than  files 1.0 MB: 3
   ◦ Files matching .semgrepignore patterns: 1
 • For a detailed list of skipped files and lines, run semgrep with the --verbose flag
Ran 1 rule on 7911 files: 0 findings.

✨ If Semgrep missed a finding, please send us feedback to let us know!
   See https://semgrep.dev/docs/reporting-false-negatives/
(venv) [redacted]-MacBook-Pro sources %
```

Verification Result

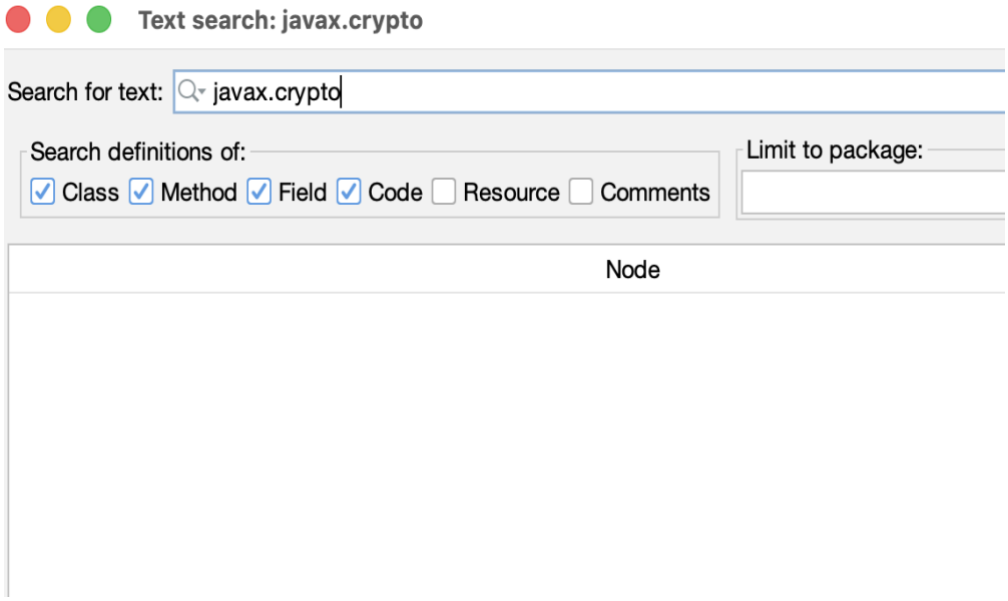
Result	Pass
Comment	

1.2.1.2 No insecure operations shall be used for symmetric cryptography

Verification Evidence

Neither static nor dynamic analysis identified any cryptographic functions implemented directly within the Android application. The application relies on the WireGuard protocol for tunnel encryption, with the underlying cryptographic logic residing in shared libraries (specifically, mullvad-daemon). The only cryptography-related functionality within the application itself is the user-facing option to configure a proxy for API access.

We performed a static scan of the Android application codebase, which returned findings related to exported activities only; no cryptographic weaknesses were identified. We additionally reviewed the mullvad-daemon shared library and confirmed that the cipher implementations in use are current and conform to accepted standards.




```
[redacted]-MacBook-Pro sources % grep -R "^crypto"
[redacted]-MacBook-Pro sources %
```

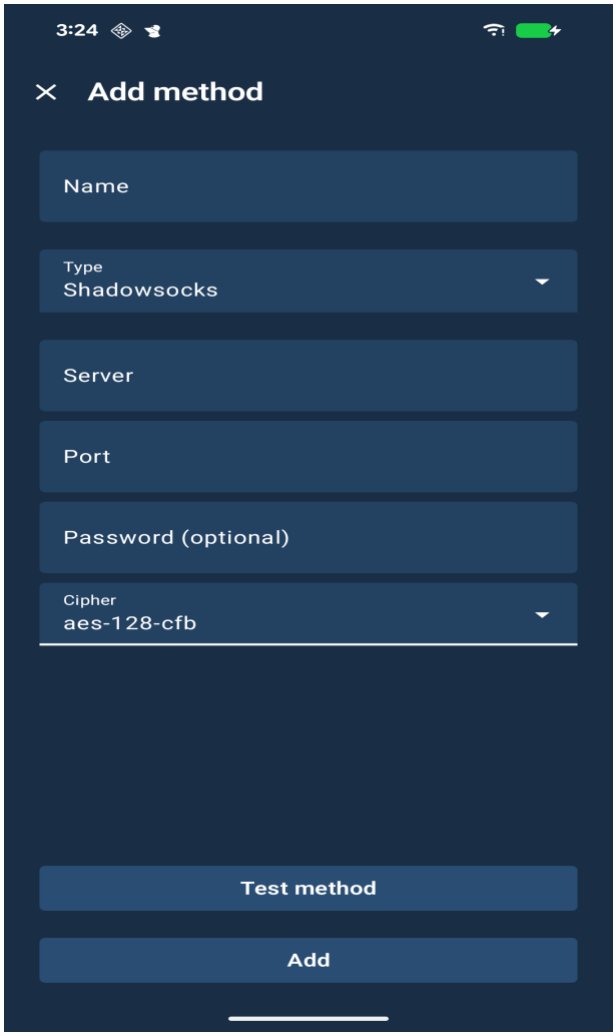
> 2) Please confirm whether shared libraries contain application's encryption logic and if they are, please explain it along with algorithms used, as we didn't observe them either.

Yes, shared libraries contain the application encryption logic. As mentioned we use the WireGuard protocol for our tunnel, it makes use of Curve25519 (asymmetric encryption), ChaCha20 (symmetric encryption) along with other crypto primitives. The WireGuard white paper describes it in more detail.

```
[redacted]-MacBook-Pro Mullvad % semgrep --lang java -e "AES" sources/

semgrep --lang java -e "Cipher" sources/

semgrep --lang java -e "KeyGenerator" sources/

semgrep --lang java -e "SecretKeySpec" sources/
```

The results of running semgrep rules were the same:

```
Scan Summary
✅ Scan completed successfully.
 • Findings: 0 (0 blocking)
 • Rules run: 1
 • Targets scanned: 7911
 • Parsed lines: ~100.0%
 • Scan skipped:
   ○ Files matching .semgrepignore patterns: 1
 • For a detailed list of skipped files and lines, run semgrep with the --verbose flag
Ran 1 rule on 7911 files: 0 findings.

🚹 A new version of Semgrep is available. See https://semgrep.dev/docs/upgrading

✨ If Semgrep missed a finding, please send us feedback to let us know!
   See https://semgrep.dev/docs/reporting-false-negatives/
```




```
Scan Summary
✅ Scan completed successfully.
 • Findings: 6 (6 blocking)
 • Rules run: 216
 • Targets scanned: 1073
 • Parsed lines: ~99.9%
 • Scan skipped:
   ○ Files matching .semgrepignore patterns: 208
 • Scan was limited to files tracked by git
 • For a detailed list of skipped files and lines, run semgrep with the --verbose flag
Ran 216 rules on 1073 files: 6 findings.
```

```
[redacted]-macbook-pro-2 mullvad-daemon % grep -Rni "cipher"
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:12:
    "cipher": "aes-128-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:26:
    "cipher": "aes-256-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:40:
    "cipher": "aes-256-gcm",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:54:
    "cipher": "aes-128-gcm",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:68:
    "cipher": "aes-128-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:82:
    "cipher": "aes-256-gcm",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:96:
    "cipher": "aes-128-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:110:
     "cipher": "aes-256-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:124:
     "cipher": "aes-128-gcm",
./src/migrations/v11.rs:204:                        "cipher": "aes-128-cfb"
./src/migrations/v11.rs:218:                        "cipher": "aes-256-cfb"
./src/migrations/v11.rs:232:                        "cipher": "aes-256-gcm"
./src/migrations/v11.rs:246:                        "cipher": "aes-128-gcm"
./src/migrations/v11.rs:260:                        "cipher": "aes-128-cfb"
./src/migrations/v11.rs:274:                        "cipher": "aes-256-gcm"
./src/migrations/v11.rs:288:                        "cipher": "aes-128-cfb"
./src/migrations/v11.rs:302:                        "cipher": "aes-256-cfb"
./src/migrations/v11.rs:316:                        "cipher": "aes-128-gcm"
./src/migrations/v7.rs:246:                cipher: extract_str(custom_bridge_shadowsocks.get("cipher"))?.to_string(),
./src/migrations/v7.rs:418:              "cipher": "aes-128-cfb"
./src/migrations/v7.rs:584:              "cipher": "aes-128-cfb"
./src/migrations/v7.rs:742:        "cipher": "aes-128-cfb",
./src/migrations/v7.rs:765:        "cipher": "aes-128-cfb"
./src/migrations/v7.rs:1003:              "cipher": "aes-128-cfb"
./src/migrations/v7.rs:1028:              "cipher": "aes-128-cfb"
```

```
[redacted]-macbook-pro-2 mullvad-daemon % semgrep scan .

┌─────────────┐
│ Semgrep CLI │
└─────────────┘

Scanning 81 files (only git-tracked) with:

✔ Semgrep OSS
  ✔ Basic security coverage for first-party code vulnerabilities.

✔ Semgrep Code (SAST)
  ✔ Find and fix vulnerabilities in the code you write with advanced scanning and expert security rules.

✘ Semgrep Supply Chain (SCA)
  ✘ Find and fix the reachable vulnerabilities in your OSS dependencies.

──────────────────────────────────────── 100% 0:00:00


┌──────────────────┐
│ 2 Code Findings  │
└──────────────────┘

    src/migrations/mod.rs
   ❯❯❱ rust.actix.path-traversal.tainted-path.tainted-path
          The application builds a file path from potentially untrusted data, which can lead to a path
          traversal vulnerability. An attacker can manipulate the path which the application uses to access
          files. If the application does not validate user input and sanitize file paths, sensitive files such
          as configuration or user data can be accessed, potentially creating or overwriting files. To prevent
          this vulnerability, validate and sanitize any input that is used to create references to file paths.
          Also, enforce strict file access controls. For example, choose privileges allowing public-facing
          applications to access only the required files.
          Details: https://sg.run/YWX5

          146┆ let settings_bytes = fs::read(&path).await.map_err(Error::Read)?;
```

```
[redacted]-macbook-pro-2 mullvad-daemon % grep -Rni "cipher"
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:12:
    "cipher": "aes-128-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:26:
    "cipher": "aes-256-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:40:
    "cipher": "aes-256-gcm",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:54:
    "cipher": "aes-128-gcm",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:68:
    "cipher": "aes-128-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:82:
    "cipher": "aes-256-gcm",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:96:
    "cipher": "aes-128-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:110:
     "cipher": "aes-256-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:124:
     "cipher": "aes-128-gcm",
./src/migrations/v11.rs:204:                        "cipher": "aes-128-cfb"
./src/migrations/v11.rs:218:                        "cipher": "aes-256-cfb"
./src/migrations/v11.rs:232:                        "cipher": "aes-256-gcm"
./src/migrations/v11.rs:246:                        "cipher": "aes-128-gcm"
./src/migrations/v11.rs:260:                        "cipher": "aes-128-cfb"
./src/migrations/v11.rs:274:                        "cipher": "aes-256-gcm"
./src/migrations/v11.rs:288:                        "cipher": "aes-128-cfb"
./src/migrations/v11.rs:302:                        "cipher": "aes-256-cfb"
./src/migrations/v11.rs:316:                        "cipher": "aes-128-gcm"
./src/migrations/v7.rs:246:                cipher: extract_str(custom_bridge_shadowsocks.get("cipher"))?.to_string(),
./src/migrations/v7.rs:418:              "cipher": "aes-128-cfb"
./src/migrations/v7.rs:584:              "cipher": "aes-128-cfb"
./src/migrations/v7.rs:742:        "cipher": "aes-128-cfb",
./src/migrations/v7.rs:765:        "cipher": "aes-128-cfb"
./src/migrations/v7.rs:1003:               "cipher": "aes-128-cfb"
./src/migrations/v7.rs:1028:               "cipher": "aes-128-cfb"
```

Verification Result

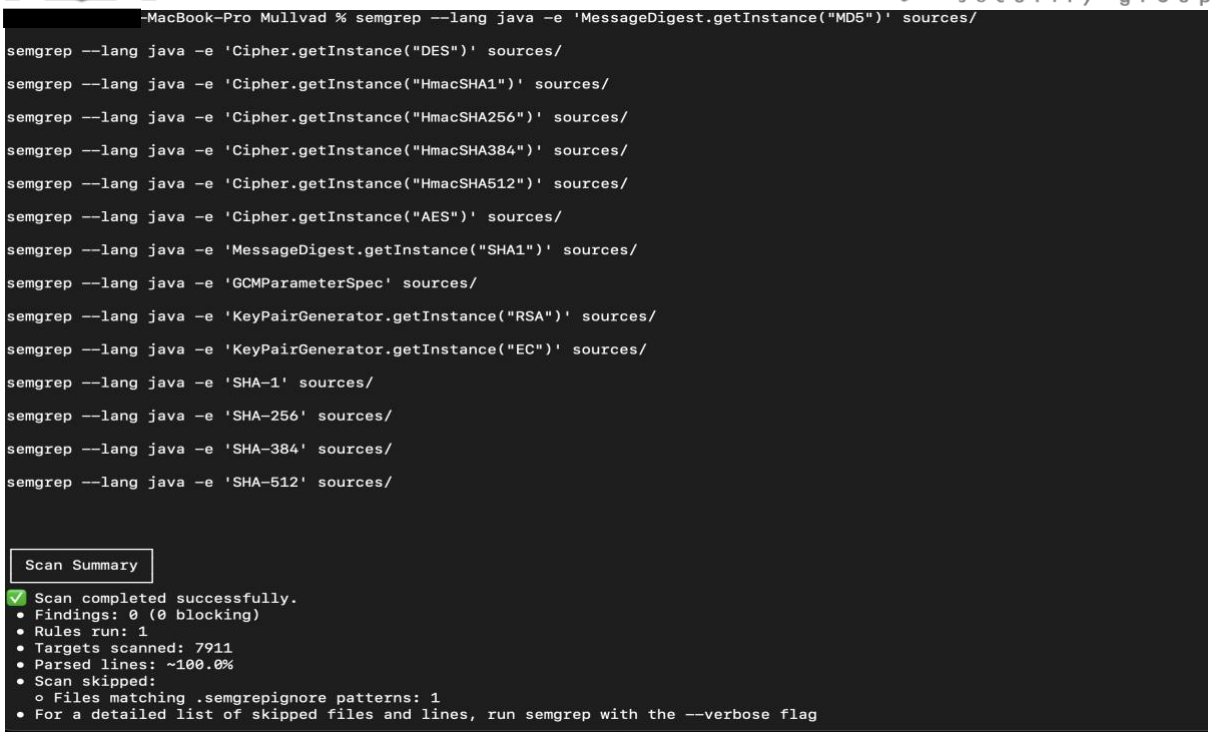
Result	Pass
Comment	

1.2.1.3 Strong cryptography shall be implemented according to industry best practices

Verification Evidence

Neither static nor dynamic analysis identified any cryptographic functions implemented directly within the Android application. The application relies on the WireGuard protocol for tunnel encryption, with the underlying cryptographic logic residing in shared libraries (specifically, mullvad-daemon).

We performed a static scan of the Android application codebase, which returned findings related to exported activities only; no cryptographic weaknesses were identified. We additionally reviewed the mullvad-daemon shared library and confirmed that the cipher implementations in use are current and conform to accepted standards.



```
[redacted]-MacBook-Pro Mullvad % semgrep --lang java -e 'MessageDigest.getInstance("MD5")' sources/

semgrep --lang java -e 'Cipher.getInstance("DES")' sources/

semgrep --lang java -e 'Cipher.getInstance("HmacSHA1")' sources/

semgrep --lang java -e 'Cipher.getInstance("HmacSHA256")' sources/

semgrep --lang java -e 'Cipher.getInstance("HmacSHA384")' sources/

semgrep --lang java -e 'Cipher.getInstance("HmacSHA512")' sources/

semgrep --lang java -e 'Cipher.getInstance("AES")' sources/

semgrep --lang java -e 'MessageDigest.getInstance("SHA1")' sources/

semgrep --lang java -e 'GCMParameterSpec' sources/

semgrep --lang java -e 'KeyPairGenerator.getInstance("RSA")' sources/

semgrep --lang java -e 'KeyPairGenerator.getInstance("EC")' sources/

semgrep --lang java -e 'SHA-1' sources/

semgrep --lang java -e 'SHA-256' sources/

semgrep --lang java -e 'SHA-384' sources/

semgrep --lang java -e 'SHA-512' sources/


 Scan Summary

✅ Scan completed successfully.
 • Findings: 0 (0 blocking)
 • Rules run: 1
 • Targets scanned: 7911
 • Parsed lines: ~100.0%
 • Scan skipped:
   ○ Files matching .semgrepignore patterns: 1
 • For a detailed list of skipped files and lines, run semgrep with the --verbose flag
```

```
[redacted]-macbook-pro-2 mullvad-daemon % grep -Rni "MD5"
[redacted]-macbook-pro-2 mullvad-daemon % grep -Rni "SHA-"
[redacted]-macbook-pro-2 mullvad-daemon % grep -Rni "SHA1"
[redacted]-macbook-pro-2 mullvad-daemon % grep -Rni "Cipher"
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:12:
    "cipher": "aes-128-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:26:
    "cipher": "aes-256-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:40:
    "cipher": "aes-256-gcm",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:54:
    "cipher": "aes-128-gcm",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:68:
    "cipher": "aes-128-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:82:
    "cipher": "aes-256-gcm",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:96:
    "cipher": "aes-128-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:110:
     "cipher": "aes-256-cfb",
./src/migrations/snapshots/mullvad_daemon__migrations__v11__test__v11_to_v12_migration_access_method_name_duplicates.snap:124:
     "cipher": "aes-128-gcm",
./src/migrations/v11.rs:204:                        "cipher": "aes-128-cfb"
./src/migrations/v11.rs:218:                        "cipher": "aes-256-cfb"
./src/migrations/v11.rs:232:                        "cipher": "aes-256-gcm"
./src/migrations/v11.rs:246:                        "cipher": "aes-128-gcm"
./src/migrations/v11.rs:260:                        "cipher": "aes-128-cfb"
./src/migrations/v11.rs:274:                        "cipher": "aes-256-gcm"
./src/migrations/v11.rs:288:                        "cipher": "aes-128-cfb"
./src/migrations/v11.rs:302:                        "cipher": "aes-256-cfb"
./src/migrations/v11.rs:316:                        "cipher": "aes-128-gcm"
./src/migrations/v7.rs:246:                cipher: extract_str(custom_bridge_shadowsocks.get("cipher"))?.to_string(),
./src/migrations/v7.rs:418:              "cipher": "aes-128-cfb"
./src/migrations/v7.rs:584:              "cipher": "aes-128-cfb"
./src/migrations/v7.rs:742:        "cipher": "aes-128-cfb",
./src/migrations/v7.rs:765:        "cipher": "aes-128-cfb"
./src/migrations/v7.rs:1003:             "cipher": "aes-128-cfb"
./src/migrations/v7.rs:1028:             "cipher": "aes-128-cfb"
```

```
[redacted]-MacBook-Pro android % grep -Rni "MD5"
./app/src/main/baseline-prof.txt:8972:SPLandroidx/compose/runtime/tooling/ComposeStackTraceMode$Companion;->getNone-MD5MrJc()I
./app/src/main/baseline-prof.txt:8973:SPLandroidx/compose/runtime/tooling/ComposeStackTraceMode$Companion;->getSourceInformation-MD5MrJc()I
./test/mockapi/src/main/res/raw/relay_list.json:4031:        "public_key": "s1c/NsfnqnwQSxao70DY4Co69AFT9e0h88IFuMD5mjs=",
./gradle/verification-keyring.keys:228:941ChvyZqbbiXHoe2SetpmD5K3oABvOaboHno8AsPA+IX+WcIC9GE4DrRhpQ4Ffj
./gradle/verification-keyring.keys:1950:Nts5E6d1OX8fBkUBtYanyyjNL1T3yDjO6VaV0SCsgAzNjUZqc4lxS9atN6md5m6l
./lib/model/src/main/kotlin/net/mullvad/mullvadvpn/lib/model/Cipher.kt:14:    RC4_MD5("rc4-md5"),
```

```
[redacted]-MacBook-Pro android % grep -Rni "Cipher"
./app/proguard-rules.pro:26:-dontwarn com.squareup.okhttp.CipherSuite
./app/src/main/baseline-prof.txt:25302:Lio/grpc/okhttp/internal/CipherSuite;
./app/src/main/baseline-prof.txt:25303:SPLio/grpc/okhttp/internal/CipherSuite;-><clinit>()V
./app/src/main/baseline-prof.txt:25304:SPLio/grpc/okhttp/internal/CipherSuite;-><init>(Ljava/lang/String;ILjava/lang/String;I)V
./app/src/main/baseline-prof.txt:25305:SPLio/grpc/okhttp/internal/CipherSuite;-><init>(Ljava/lang/String;ILjava/lang/String;IIII)V
./app/src/main/baseline-prof.txt:25320:SPLio/grpc/okhttp/internal/ConnectionSpec$Builder;->cipherSuites([Lio/grpc/okhttp/internal/Ciph
erSuite;)Lio/grpc/okhttp/internal/ConnectionSpec$Builder;
./test/mockapi/src/main/res/raw/relay_list.json:11495:        "cipher": "aes-256-gcm",
./test/mockapi/src/main/res/raw/relay_list.json:11501:        "cipher": "aes-256-cfb",
./test/mockapi/src/main/res/raw/relay_list.json:11507:        "cipher": "aes-256-gcm",
./lib/ui/resource/src/main/res/values-ru/strings.xml:82:    <string name="cipher">Шифр</string>
./lib/ui/resource/src/main/res/values-zh-rTW/strings.xml:82:    <string name="cipher">加密方式</string>
./lib/ui/resource/src/main/res/values-it/strings.xml:82:    <string name="cipher">Codice</string>
./lib/ui/resource/src/main/res/values-zh-rCN/strings.xml:82:    <string name="cipher">加密方式</string>
./lib/ui/resource/src/main/res/values-ja/strings.xml:82:    <string name="cipher">暗号化</string>
./lib/ui/resource/src/main/res/values-da/strings.xml:82:    <string name="cipher">Chiffer</string>
./lib/ui/resource/src/main/res/values/strings.xml:315:    <string name="cipher">Cipher</string>
./lib/ui/resource/src/main/res/values-pl/strings.xml:82:    <string name="cipher">Szyfrowanie</string>
./lib/ui/resource/src/main/res/values-sv/strings.xml:82:    <string name="cipher">Chiffrering</string>
./lib/ui/resource/src/main/res/values-tr/strings.xml:82:    <string name="cipher">Şifre</string>
./lib/ui/resource/src/main/res/values-th/strings.xml:82:    <string name="cipher">เข้ารหัส</string>
```

```
    CipherSelection(cipher = formData.cipher, onCipherChange = onCipherChange)
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodScreen.kt:499:
private fun CipherSelection(cipher: Cipher, onCipherChange: (Cipher) -> Unit) {
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodScreen.kt:502:
        label = stringResource(id = R.string.cipher),
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodScreen.kt:503:
        title = cipher.label,
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodScreen.kt:506:
        Cipher.listAll().forEachIndexed { index, item ->
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodScreen.kt:514:
                    onCipherChange(item)
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodScreen.kt:522:
                            .alpha(if (item == cipher) AlphaVisible else AlphaInvisible),
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodUiState.kt:7:i
mport net.mullvad.mullvadvpn.lib.model.Cipher
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodUiState.kt:40:
    val cipher: Cipher = Cipher.first(),
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodUiState.kt:63:
                    cipher = customProxy.cipher,
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodViewModel.kt:3
2:import net.mullvad.mullvadvpn.lib.model.Cipher
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodViewModel.kt:9
2:    fun onCipherChanged(cipher: Cipher) {
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodViewModel.kt:9
3:        formData.update { it.copy(cipher = cipher) }
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodViewModel.kt:1
94:                cipher = formData.cipher,
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/edit/EditApiAccessMethodUiStatePreview
ParameterProvider.kt:34:                            cipher = data.cipher,
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/detail/ApiAccessMethodDetailsUiStatePr
eviewParameterProvider.kt:8:import net.mullvad.mullvadvpn.lib.model.Cipher
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/screen/detail/ApiAccessMethodDetailsUiStatePr
eviewParameterProvider.kt:44:                                Cipher.CHACHA20_IETF_POLY1305,
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/ApiAccessPreviewData.kt:7:import net.mullvad.
mullvadvpn.lib.model.Cipher
./lib/feature/apiaccess/impl/src/main/java/net/mullvad/mullvadvpn/feature/apiaccess/impl/ApiAccessPreviewData.kt:54:            cipher
 = Cipher.fromString("aes-128-cfb"),
```

Verification Result

Result	Pass
Comment	VK: sorry for the confuse, but the sources were checked initially. Only crypto-related thing in the app is their custom API access method through proxy, that could be configured by user (like choose cipher for the proxy). As for the lib, all ok, nothing to report as well.

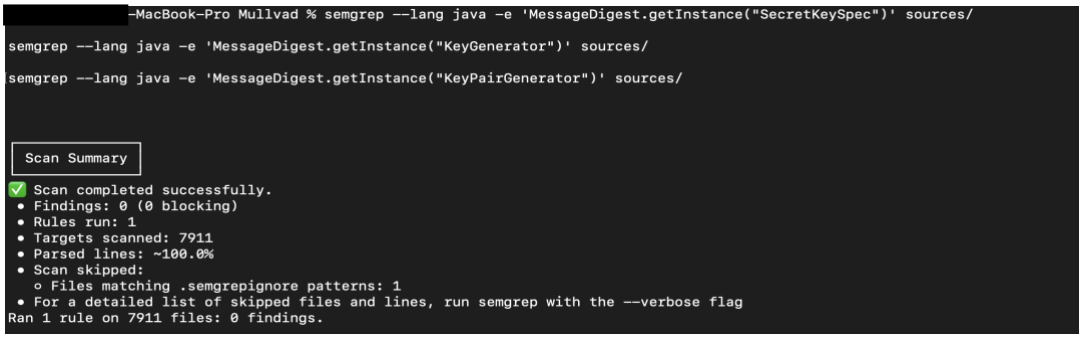
1.2.2 The app performs key management according to industry best practices

1.2.2.1 Cryptographic keys shall only be used for their defined purpose

Verification Evidence

As the application does not implement cryptographic functions directly, no cryptographic key generation functions were identified within the application code.

We observed private key generation with RSA and EC algorithms (see the image below); however, this logic resides exclusively within a third-party package (io.grpc.okhttp) and is not invoked by the application's own code. We reviewed the remaining sources and identified no cases of improper key usage. Hard-coded values present in the codebase are test values only.




```
public static PrivateKey y(InputStream inputStream) throws InvalidKeySpecException, IOException {
    String line;
    BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(inputStream, "UTF-8"));
    do {
        line = bufferedReader.readLine();
        if (line == null) {
            break;
        }
    } while (!"-----BEGIN PRIVATE KEY-----".equals(line));
    StringBuilder sb2 = new StringBuilder();
    while (true) {
        String line2 = bufferedReader.readLine();
        if (line2 == null || "-----END PRIVATE KEY-----".equals(line2)) {
            break;
        }
        sb2.append(line2);
    }
    PKCS8EncodedKeySpec pKCS8EncodedKeySpec = new PKCS8EncodedKeySpec(f8.e.f7873c.a(sb2.toString()));
    try {
        try {
            return KeyFactory.getInstance("RSA").generatePrivate(pKCS8EncodedKeySpec);
        } catch (InvalidKeySpecException unused) {
            return KeyFactory.getInstance("EC").generatePrivate(pKCS8EncodedKeySpec);
        }
    } catch (InvalidKeySpecException e10) {
        throw new InvalidKeySpecException("Neither RSA nor EC worked", e10);
    }
}
```

Usage for: ha.a.y(InputStream) PrivateKey

Node	
io.grpc.okhttp.OkHttpChannelBuilder.createKeyManager(InputStream, InputStream) KeyManager[]	PrivateKey privateKeyY = ha.a.y(inputStream2);

```
[redacted]-macbook-pro-2 mullvad-daemon % grep -Rni "private_key"
./src/migrations/device.rs:80:        if device.pubkey == wg_data.private_key.public_key() {
./src/migrations/account_history.rs:151:      "private_key": "mAdSb4AfQOsAD5O/5+zG1oIhk3cUl0jUsyOeaOMFu3o=",
./src/migrations/account_history.rs:162:      "private_key": "mAdSb4AfQOsAD5O/5+zG1oIhk3cUl0jUsyOeaOMFu3o=",
./src/migrations/account_history.rs:252:    "private_key": "mAdSb4AfQOsAD5O/5+zG1oIhk3cUl0jUsyOeaOMFu3o=",
./src/tunnel.rs:166:            private_key: data.device.wg_data.private_key,
./src/device/service.rs:47:        let private_key = PrivateKey::new_from_random();
./src/device/service.rs:48:        let pubkey = private_key.public_key();
./src/device/service.rs:73:                        private_key,
./src/device/service.rs:86:        let private_key = PrivateKey::new_from_random();
./src/device/service.rs:87:        let pubkey = private_key.public_key();
./src/device/service.rs:109:                    private_key,
./src/device/service.rs:170:       let private_key = PrivateKey::new_from_random();
./src/device/service.rs:174:       let pubkey = private_key.public_key();
./src/device/service.rs:191:           private_key,
./src/device/service.rs:202:       let private_key = PrivateKey::new_from_random();
./src/device/service.rs:206:       let pubkey = private_key.public_key();
./src/device/service.rs:222:           private_key,
./src/device/mod.rs:198:        if device.pubkey != wg_data.private_key.public_key() {
./src/device/mod.rs:213:        if device.pubkey != self.wg_data.private_key.public_key() {
./src/device/mod.rs:228:            pubkey: device.wg_data.private_key.public_key(),
./src/device/mod.rs:806:                let current_pubkey = current_config.device.wg_data.private_key.public_key();
```

```
[redacted]-MacBook-Pro android % grep -Rni "PrivateKey"
[redacted]-MacBook-Pro android % grep -Rni "PublicKey"
./test/mockapi/src/main/kotlin/net/mullvad/mullvadvpn/test/mockapi/util/JsonUtils.kt:17:fun deviceJson(id: String, name: String, publi
cKey: String, creationDate: ZonedDateTime) =
./test/mockapi/src/main/kotlin/net/mullvad/mullvadvpn/test/mockapi/util/JsonUtils.kt:21:        put("pubkey", publicKey)
./test/mockapi/src/main/kotlin/net/mullvad/mullvadvpn/test/mockapi/server/MockApiRouter.kt:98:                    publicKey = cachedKe
y,
./test/mockapi/src/main/kotlin/net/mullvad/mullvadvpn/test/mockapi/server/MockApiRouter.kt:116:                            publicKey =
 newKey,
./test/mockapi/src/main/kotlin/net/mullvad/mullvadvpn/test/mockapi/server/MockApiRouter.kt:134:                          publicKey = cac
hedKey,
[redacted]-MacBook-Pro android %
```

Verification Result

Result	Pass
Comment	

1.2.2.2 Cryptographic key management shall be implemented properly

Verification Evidence

No sensitive cryptographic data was found to be logged or exposed within the application.

We scanned the application and its shared libraries for hard-coded keys. The scan identified 133 occurrences of the same token used for VPN relay configuration and 1 instance of private key-related code within the third-party package io.grpc.okhttp. Neither finding represents cryptographic data exposure; the VPN relay tokens are non-sensitive operational values, and the private key logic is confined to a third-party dependency.

We manually reviewed System.out.print function calls and confirmed that they output only non-sensitive informational and error messages. Hard-coded values identified in the source code are limited to non-sensitive test values.

```
[redacted]-MacBook-Pro arm64-v8a % strings libmullvad_jni.so > libmullvad.txt
[redacted]-MacBook-Pro arm64-v8a % gitleaks dir libmullvad.txt -vvv

    ○
    │╲
    │ ○
    ○ ▒
    ▒    gitleaks

10:53AM INF scanned ~526864 bytes (526.86 KB) in 718ms
10:53AM INF no leaks found
```

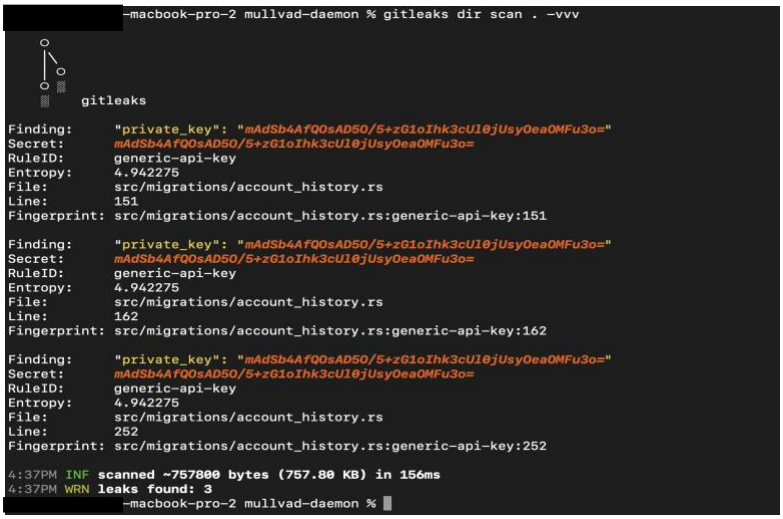
```
[redacted]-MacBook-Pro arm64-v8a % strings libdatastore_shared_counter.so > libdatastore.txt
[redacted]-MacBook-Pro arm64-v8a % gitleaks dir libdatastore.txt -vvv

    ○
    │╲
    │ ○
    ○ ▒
    ▒    gitleaks

10:53AM INF scanned ~1423 bytes (1.42 KB) in 1.81ms
10:53AM INF no leaks found
vkachan@Vadyms-MacBook-Pro arm64-v8a % strings libandroidx.graphics.path.so > libandroid.txt
vkachan@Vadyms-MacBook-Pro arm64-v8a % gitleaks dir libandroid.txt -vvv

    ○
    │╲
    │ ○
    ○ ▒
    ▒    gitleaks

10:54AM INF scanned ~1074 bytes (1.07 KB) in 2.9ms
10:54AM INF no leaks found
[redacted]-MacBook-Pro arm64-v8a %
```

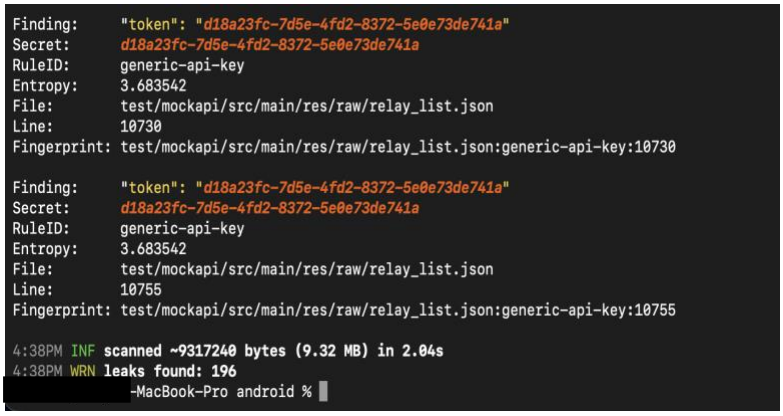
```
Finding:     "token": "d18a23fc-7d5e-4fd2-8372-5e0e73de741a"
Secret:      d18a23fc-7d5e-4fd2-8372-5e0e73de741a
RuleID:      generic-api-key
Entropy:     3.683542
File:        resources/assets/relays.json
Line:        16335
Fingerprint: resources/assets/relays.json:generic-api-key:16335

Finding:     "token": "d18a23fc-7d5e-4fd2-8372-5e0e73de741a"
Secret:      d18a23fc-7d5e-4fd2-8372-5e0e73de741a
RuleID:      generic-api-key
Entropy:     3.683542
File:        resources/assets/relays.json
Line:        16362
Fingerprint: resources/assets/relays.json:generic-api-key:16362

Finding:     } while (!"-----BEGIN PRIVATE KEY-----".equals(line));
        StringBuilder sb2 = new StringBuilder();
      ...                                       ".equals(line2)) {
Secret:      -----BEGIN PRIVATE KEY-----".equals(line));
        StringBuilder sb2 = new StringBuilder();
      ...
RuleID:      private-key
Entropy:     4.489470
File:        sources/ha/a.java
Line:        746
Fingerprint: sources/ha/a.java:private-key:746

11:31AM INF scanned ~31351978 bytes (31.35 MB) in 3.07s
```

```
System.out.println(" 0 " + mVar.v(0.0d));
System.out.println(" 1 " + mVar.v(1.0d));
System.out.println("EXCEPTION : " + e);
System.out.println((Object) "FocusRelatedWarning: Dispatching rotary event while the focus system is invalidated.");
System.out.println((Object) "FocusRelatedWarning: Dispatching indirect pointer event while the focus system is invalidated.");
System.out.println((Object) "FocusRelatedWarning: Dispatching intercepted soft keyboard event while the focus system is invalidated.");
System.out.println((Object) string);
System.out.println("couldn't find reference for " + next2);
System.out.println((Object) "FocusRelatedWarning: Dispatching key event while focus system is invalidated.");
System.out.println((Object) "FocusRelatedWarning: \n   FocusRequester is not initialized. Here are some possible fixes:\n\n   1. Rememb
System.out.println((Object) "FocusRelatedWarning: \n   FocusRequester is not initialized. Here are some possible fixes:\n\n   1. Rememb
System.out.println("WTF");
```

```
[redacted]-macbook-pro-2 mullvad-daemon % gitleaks dir scan . -vvv

    gitleaks

Finding:     "private_key": "mAdSb4AfQOsAD5O/5+zG1oIhk3cUl0jUsyOeaOMFu3o="
Secret:      mAdSb4AfQOsAD5O/5+zG1oIhk3cUl0jUsyOeaOMFu3o=
RuleID:      generic-api-key
Entropy:     4.942275
File:        src/migrations/account_history.rs
Line:        151
Fingerprint: src/migrations/account_history.rs:generic-api-key:151

Finding:     "private_key": "mAdSb4AfQOsAD5O/5+zG1oIhk3cUl0jUsyOeaOMFu3o="
Secret:      mAdSb4AfQOsAD5O/5+zG1oIhk3cUl0jUsyOeaOMFu3o=
RuleID:      generic-api-key
Entropy:     4.942275
File:        src/migrations/account_history.rs
Line:        162
Fingerprint: src/migrations/account_history.rs:generic-api-key:162

Finding:     "private_key": "mAdSb4AfQOsAD5O/5+zG1oIhk3cUl0jUsyOeaOMFu3o="
Secret:      mAdSb4AfQOsAD5O/5+zG1oIhk3cUl0jUsyOeaOMFu3o=
RuleID:      generic-api-key
Entropy:     4.942275
File:        src/migrations/account_history.rs
Line:        252
Fingerprint: src/migrations/account_history.rs:generic-api-key:252

4:37PM INF scanned ~757800 bytes (757.80 KB) in 156ms
4:37PM WRN leaks found: 3
[redacted]-macbook-pro-2 mullvad-daemon %
```

```
Finding:     "token": "d18a23fc-7d5e-4fd2-8372-5e0e73de741a"
Secret:      d18a23fc-7d5e-4fd2-8372-5e0e73de741a
RuleID:      generic-api-key
Entropy:     3.683542
File:        test/mockapi/src/main/res/raw/relay_list.json
Line:        10730
Fingerprint: test/mockapi/src/main/res/raw/relay_list.json:generic-api-key:10730

Finding:     "token": "d18a23fc-7d5e-4fd2-8372-5e0e73de741a"
Secret:      d18a23fc-7d5e-4fd2-8372-5e0e73de741a
RuleID:      generic-api-key
Entropy:     3.683542
File:        test/mockapi/src/main/res/raw/relay_list.json
Line:        10755
Fingerprint: test/mockapi/src/main/res/raw/relay_list.json:generic-api-key:10755

4:38PM INF scanned ~9317240 bytes (9.32 MB) in 2.04s
4:38PM WRN leaks found: 196
[redacted]-MacBook-Pro android %
```





Verification Result

Result	Pass
Comment	

1.3 Auth

1.3.1 The app uses secure authentication and authorization protocols and follows the relevant best practices

1.3.1.1 If using OAuth 2.0 for authorization, or if using OpenID Connect for authentication, Proof Key for Code Exchange (PKCE) shall be implemented to protect the code grant

Verification Evidence

The application's requests were unavailable for observation. The developer team mentioned that the application employs certificate pinning, but due to time constraints, attempting to bypass it was not feasible. However, we observed that the application only requires a 16-digit number for login, and the developer team confirmed the flow:

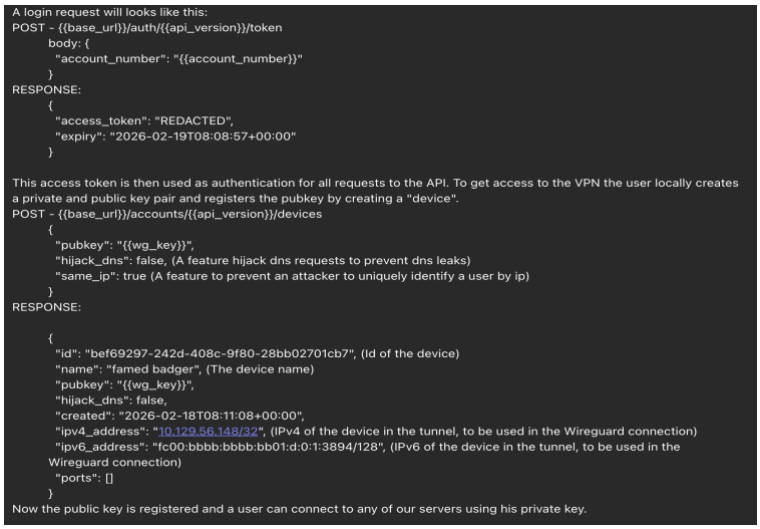
1) The user sends a request with their account number.

2) The server responds by providing the access token.

3) The access token is then used to make requests to the API.

Additionally, the flow for VPN access was provided, where the user simply sends their public key to the server to register it.

Therefore, no OAuth 2.0 flow is utilized, and the Mobile App ADA 1.3.1.1 requirement is not applicable.



Verification Result

Result	NA (Pass)
Comment	

1.4 Network

1.4.1 The app secures all network traffic according to the current best practices

1.4.1.1 Network connections shall be encrypted

Verification Evidence

The only identified http:// connection-related occurrences were observed to be used solely for testing purposes in the application's source code, but not in its APK.

No unencrypted HTTP connections were identified. In addition, the developer team confirmed that all API connections use TLSv1.3.

android/test/e2e/src/main/kotlin/net/mullvad/mullvadvpn/test/e2e/router/packetCapture/PacketCapture.kt:

```
private fun defaultHttpClient(): HttpClient =
    HttpClient(CIO) {
        defaultRequest { url("http://${InstrumentationRegistry.getArguments().getRaasHost()}") }
        engine { requestTimeout = REQUEST_TIMEOUT_MS }

        install(ContentNegotiation) {
            json(
                Json {
                    ignoreUnknownKeys = true
                    isLenient = true
                    prettyPrint = true
                }
            )
        }
```

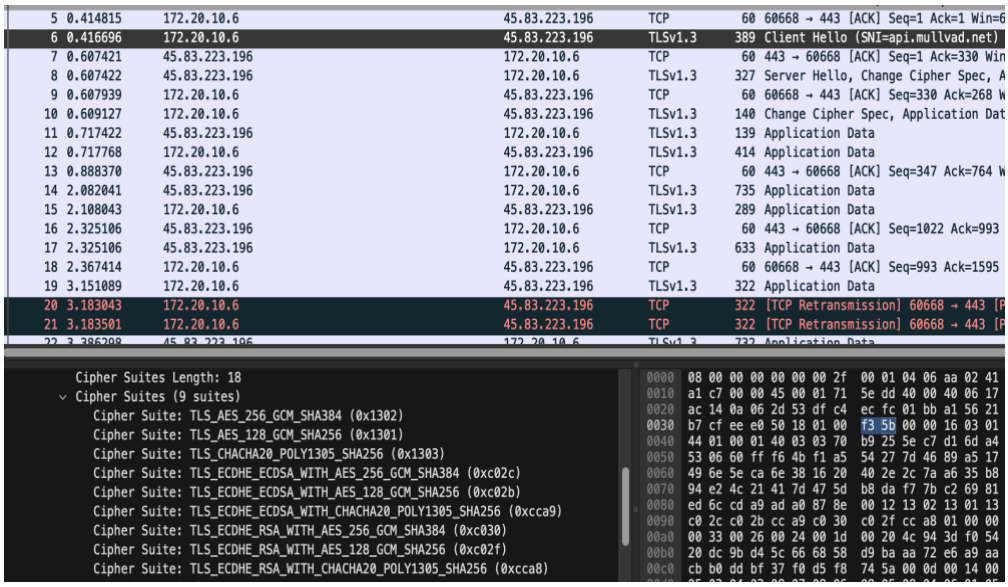
android/test/e2e/src/main/kotlin/net/mullvad/mullvadvpn/test/e2e/router/firewall/FirewallClient.kt:

```
private fun defaultHttpClient(): HttpClient =
    HttpClient(CIO) {
        defaultRequest {
            url {
                protocol = URLProtocol.HTTP
                host = InstrumentationRegistry.getArguments().getRaasHost()
            }
        }
```

android/test/e2e/src/main/kotlin/net/mullvad/mullvadvpn/test/e2e/constant/Constants.kt:

```
fun Bundle.getRaasHost() =
    InstrumentationRegistry.getArguments().getRequiredArgument("mullvad.test.e2e.config.raas.host")
```

Regarding API traffic: All API connections use TLS 1.3 with certificate pinning.



Verification Result

Result	Pass
Comment	

1.4.1.2 TLS configuration of network connections shall adhere to industry best practices

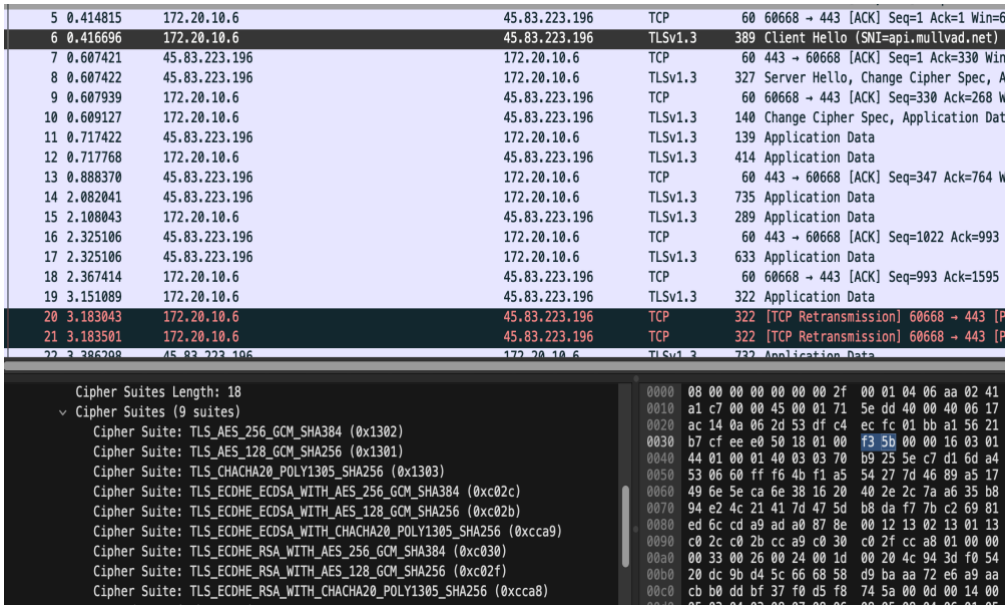
Verification Evidence

The application uses the platform TLS stack via SSLContext.getInstance("TLS"). This call appears in the application bundle; the application source code does not reference TLS directly, as HTTP connections are handled through okhttp. Dynamic analysis confirmed support for TLSv1.3.

The application's targetMinSdk is set to 36, which enables TLSv1.3 by default. We confirmed that TLSv1.3 is used for connections to the API.

The following code snippet from k9.j demonstrates TLS initialization:

Provider provider4 = SSLContext.getDefault().getProvider(); try { try { SSLContext sSLContext = SSLContext.getInstance("TLS", provider4);



Regarding API traffic: All API connections use TLS 1.3 with certificate pinning. The app comes bundled with the Let's encrypt root certificate and only accepts connections with servers having a valid certificate issued to api.mullvad.net and signed with this bundled certificate. You can find more details here, the full document also covers multiple aspects of our security.

Verification Result

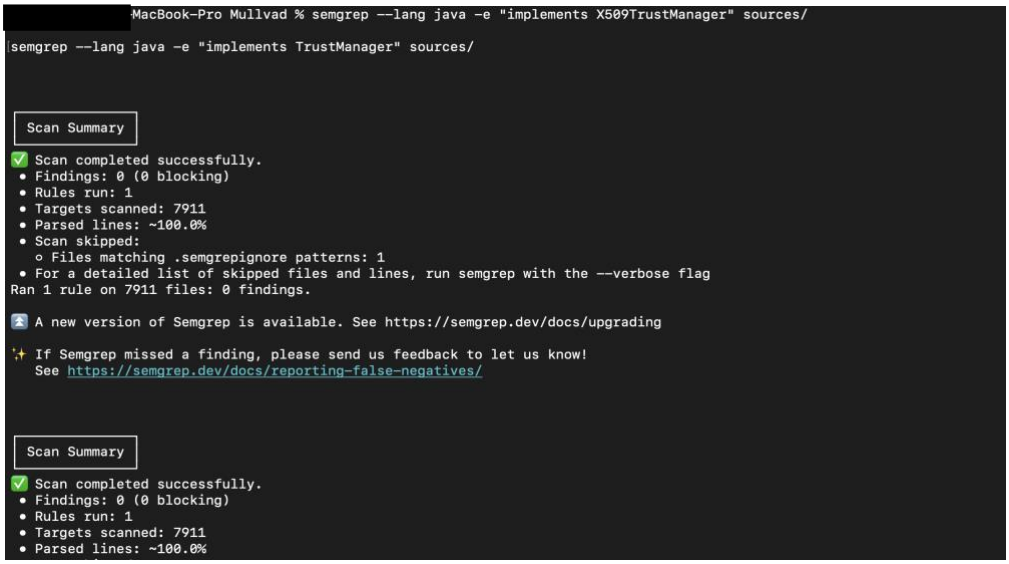
Result	Pass
Comment	

1.4.1.3 Endpoint identity shall be verified on network connections

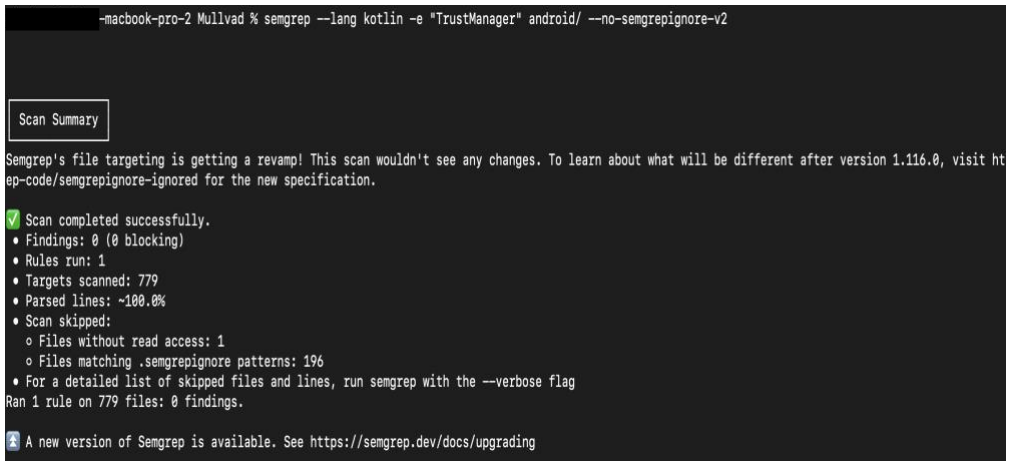
Verification Evidence

No permissive TrustManagers, trust-all implementations, or HostnameVerifier bypasses were identified. We confirmed this through both static analysis of the application bundle and a review of the accessible source code.

The application implements certificate pinning for sensitive network connections. We verified this through dynamic analysis of the application's network behavior.



Regarding API traffic: All API connections use TLS 1.3 with certificate pinning. The app comes bundled with the Let's encrypt root certificate and only accepts connections with servers having a valid certificate issued to api.mullvad.net and signed with this bundled certificate. You can find more details here, the full document also covers multiple aspects of our security.



Verification Result

Result	Pass
Comment	

1.5 Platform

1.5.1 The app uses IPC mechanisms securely

1.5.1.1 The app shall limit content provider exposure and harden queries against injection attacks

Verification Evidence

No provider is accessible to third-party applications, and all IPC exposure is minimized by design. Where file sharing is required, access is mediated via FileProvider with explicit URI permission grants.

```
(venv)                -MacBook-Pro base % grep -R "<provider" -n AndroidManifest.xml
AndroidManifest.xml:132:        <provider android:authorities="net.mullvad.mullvadvpn.FileProvider" android:exported="false" android:grantUriPermissions="true" android:name="net.mullv
ad.mullvadvpn.provider.MullvadFileProvider">
AndroidManifest.xml:164:        <provider android:authorities="net.mullvad.mullvadvpn.androidx-startup" android:exported="false" android:name="androidx.startup.InitializationProvider"
>
```

Verification Result

Result	Pass
Comment	

1.5.1.2 The app shall use verified links and sanitize all link input data

Verification Evidence

The application does not use deep links. Therefore, Mobile App ADA 1.5.1.2 requirement is satisfied.

```
                -MacBook-Pro base % grep -R "^autoVerify" -n AndroidManifest.xml
                -MacBook-Pro base %
```

```
                -MacBook-Pro base % grep -R "<data" -n AndroidManifest.xml
                -MacBook-Pro base %
```

Verification Result

Result	Pass
Comment	

1.5.1.3 Any sensitive functionality exposed via IPC shall be intentional and at the minimum required level

Verification Evidence

During automated analysis, several intent usages were flagged. However, we were not able to identify insecure or unintended exposure of sensitive functionality. All IPC-exposed

functionality appears to be intentional, scoped to the minimum required access level, and aligned with secure Android IPC design practices.

```
[redacted]-MacBook-Pro base % grep -R "android:exported=\"true\"" AndroidManifest.xml
AndroidManifest.xml:        <activity-alias android:banner="@mipmap/ic_banner_browser" android:enabled="false" android:exported="true" android:icon="@mipmap/ic_launcher_browser" andr
oid:label="@string/app_name_browser" android:name="net.mullvad.mullvadvpn.ui.obfuscation.MainActivityAltBrowser" android:targetActivity="net.mullvad.mullvadvpn.ui.MainActivity">
AndroidManifest.xml:        <activity-alias android:banner="@mipmap/ic_banner_ninja" android:enabled="false" android:exported="true" android:icon="@mipmap/ic_launcher_ninja" android:
label="@string/app_name_ninja" android:name="net.mullvad.mullvadvpn.ui.obfuscation.MainActivityAltNinja" android:targetActivity="net.mullvad.mullvadvpn.ui.MainActivity">
AndroidManifest.xml:        <activity-alias android:banner="@mipmap/ic_banner_notes" android:enabled="false" android:exported="true" android:icon="@mipmap/ic_launcher_notes" android:
label="@string/app_name_notes" android:name="net.mullvad.mullvadvpn.ui.obfuscation.MainActivityAltNotes" android:targetActivity="net.mullvad.mullvadvpn.ui.MainActivity">
AndroidManifest.xml:        <activity-alias android:banner="@mipmap/ic_banner_weather" android:enabled="false" android:exported="true" android:icon="@mipmap/ic_launcher_weather" andr
oid:label="@string/app_name_weather" android:name="net.mullvad.mullvadvpn.ui.obfuscation.MainActivityAltWeather" android:targetActivity="net.mullvad.mullvadvpn.ui.MainActivity">
AndroidManifest.xml:        <activity-alias android:banner="@mipmap/ic_banner_game" android:enabled="false" android:exported="true" android:icon="@mipmap/ic_launcher_game" android:la
bel="@string/app_name_game" android:name="net.mullvad.mullvadvpn.ui.obfuscation.MainActivityAltGame" android:targetActivity="net.mullvad.mullvadvpn.ui.MainActivity">
AndroidManifest.xml:        <activity-alias android:banner="@mipmap/ic_banner" android:enabled="true" android:exported="true" android:icon="@mipmap/ic_launcher" android:label="@strin
g/app_name" android:name="net.mullvad.mullvadvpn.ui.obfuscation.MainActivityAltDefault" android:targetActivity="net.mullvad.mullvadvpn.ui.MainActivity">
AndroidManifest.xml:        <service android:exported="true" android:icon="@null" android:label="@string/toggle_vpn" android:name="net.mullvad.mullvadvpn.tile.MullvadTileService" and
roid:permission="android.permission.BIND_QUICK_SETTINGS_TILE">
AndroidManifest.xml:        <activity android:exported="true" android:name="androidx.compose.ui.tooling.PreviewActivity"/>
AndroidManifest.xml:        <receiver android:directBootAware="false" android:enabled="true" android:exported="true" android:name="androidx.profileinstaller.ProfileInstallReceiver" a
ndroid:permission="android.permission.DUMP">
```

Verification Result

Result	Pass
Comment	

1.5.1.4 All Pending Intents shall be immutable or otherwise justified for mutability

Verification Evidence

We identified the following PendingIntent that does not have the flag FLAG_IMMUTABLE set.

zf.b decompilation:

Intent intent2 = new Intent(); intent2.setClassName(mullvadTileService2.getApplicationContext().getPackageName(), "net.mullvad.mullvadvpn.ui.MainActivity"); intent2.setFlags(872415232); intent2.setAction("android.intent.action.MAIN"); int i14 = Build.VERSION.SDK_INT; if (i14 >= 34) { mullvadTileService2.startActivityAndCollapse(PendingIntent.getActivity(mullvadTileService2.getApplicationContext(), 0, intent2, i14 >= 34 ? 184549376 : i14 >= 31 ? 167772160 : 134217728));

As the application is open-source, the previous decompiled code corresponds to the following at /android/app/src/main/kotlin/net/mullvad/mullvadvpn/app/tile/MullvadTileService.kt

```
@SuppressLint("StartActivityAndCollapseDeprecated")
private fun MullvadTileService.startActivityAndCollapseCompat(intent: Intent) {
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.UPSIDE_DOWN_CAKE) {
        val pendingIntent =
            PendingIntent.getActivity(
                applicationContext,
                0,
                intent,
                SdkUtils.getSupportedPendingIntentFlags(),
            )
        startActivityAndCollapse(pendingIntent)
    } else {
        @Suppress("DEPRECATION") startActivityAndCollapse(intent)
    }
}
```

This does not comply with the Mobile App ADA 1.5.1.4 requirement.

Recommendation: Set FLAG_IMMUTABLE for the PendingIntents.

References: https://github.com/appdefensealliance/ASA-WG/blob/main/Mobile%20App%20Profile/Mobile%20App%20Test%20Guide.md#1514-all-pending-intents-shall-be-immutable-or-otherwise-justified-for-mutability

Verification Result

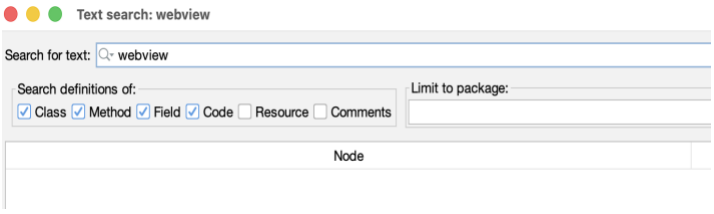
Result	Fail
Comment	

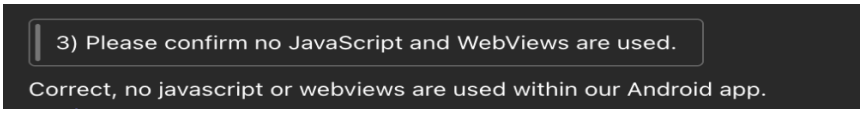
1.5.2 The app uses WebViews securely

1.5.2.1 WebViews shall securely execute JavaScript

Verification Evidence

The application does not use WebViews, which was additionally confirmed by the developer team. Therefore, the Mobile App ADA 1.5.2.1 requirement is not applicable.





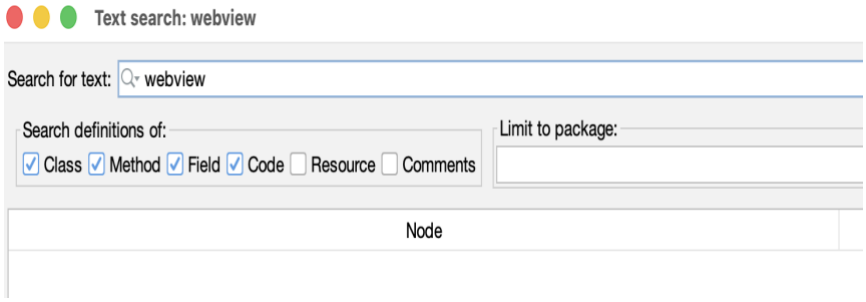
Verification Result

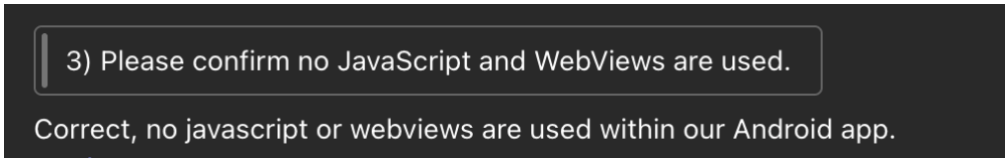
Result	NA (Pass)
Comment	

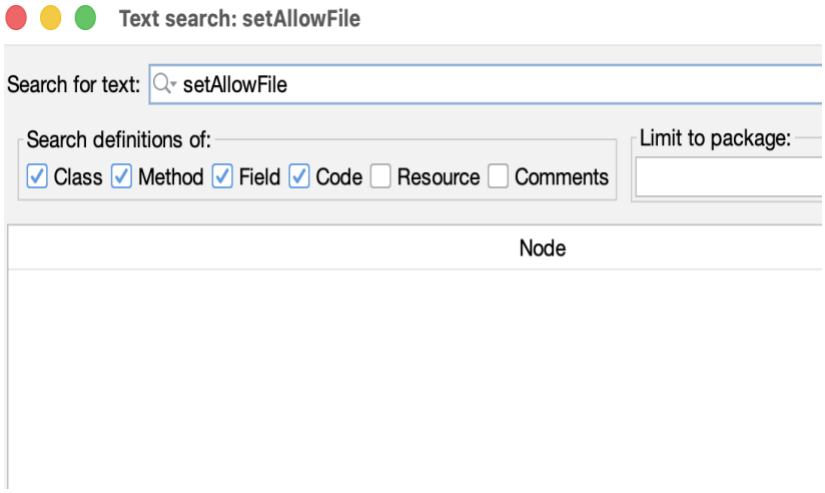
1.5.2.2 WebView shall be configured to allow the minimum set of protocol handlers required while disabling potentially dangerous handlers

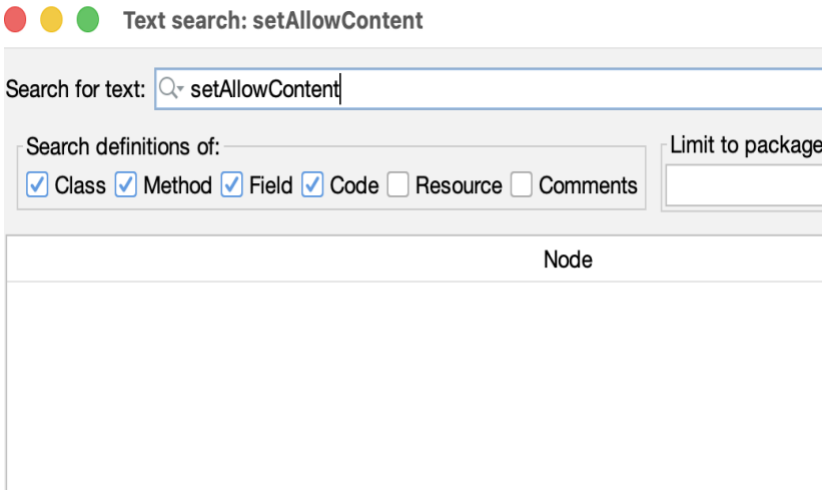
Verification Evidence

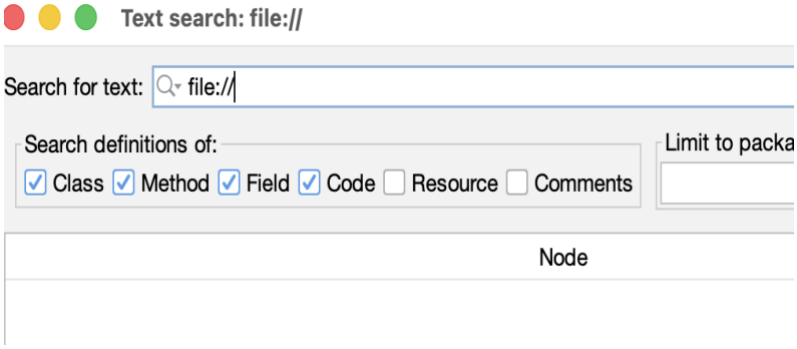
The application does not use WebViews (as additionally confirmed by the developer team) or any potentially dangerous handlers. Therefore, the Mobile App ADA 1.5.2.2 requirement is not applicable.











Verification Result

Result	NA (Pass)
Comment	

1.5.3 The app uses the user interface securely

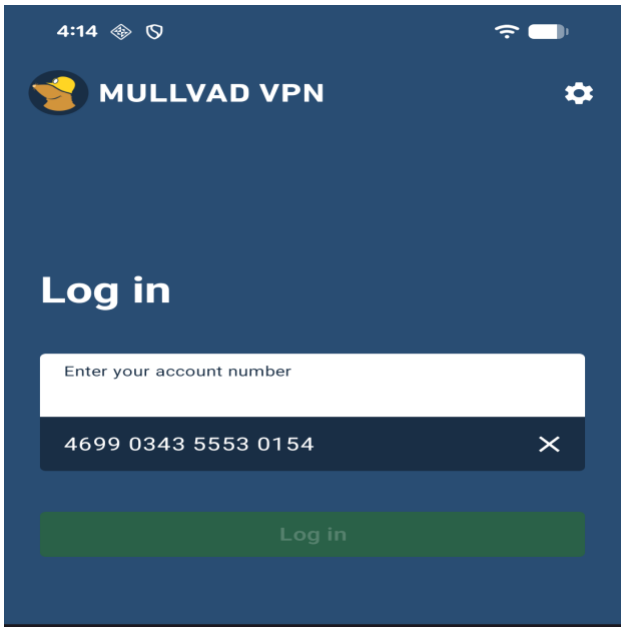
1.5.3.1 The app shall by default mask data in the User Interface when it is known to be sensitive

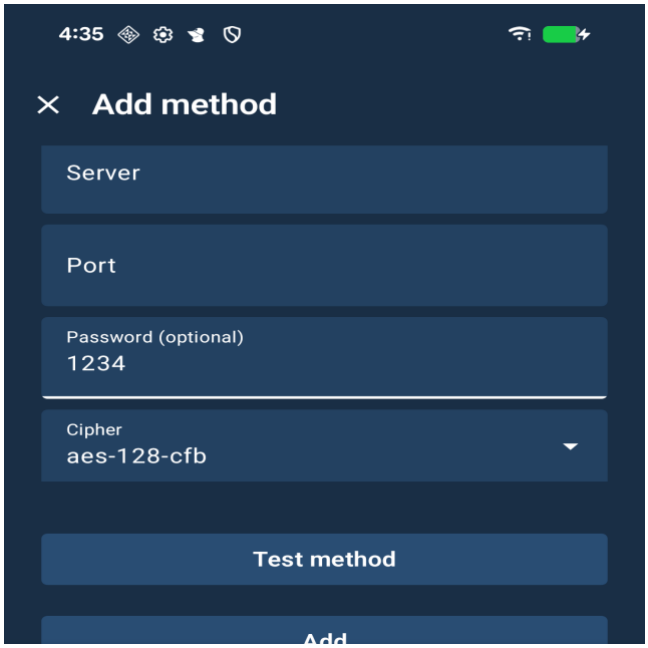
Verification Evidence

We observed that the application's login flow requires only the 16-digit account number. As this is the only value used for authentication, it is considered sensitive. However, the application does not mask it. Moreover, the application suggests this value, stored in its SharedPreferences, after the first successful login.

Additionally, the application has an optional password field that can be accessed while adding a custom API access method, and it is not masked either.

This does not comply with the Mobile App ADA 1.5.3.1 requirement.





Recommendation: Mask sensitive values by default, so that they are not visible in cleartext.

Location: Android application

References: https://github.com/appdefensealliance/ASA-WG/blob/main/Mobile%20App%20Profile/Mobile%20App%20Test%20Guide.md#1531-the-app-shall-by-default-mask-data-in-the-user-interface-when-it-is-known-to-be-sensitive

Verification Result

Result	Fail
Comment	

1.6 Code

1.6.1 The app requires an up-to-date platform version

1.6.1.1 The app shall set the targetSdkVersion to an up-to-date platform version

Verification Evidence

The targetSdkVersion of the application was identified as set to 36, which is the latest available version at the time of testing. According to the requirement, the application must have targetSdkVersion set to N-2 or above, where N is the latest SDK at the time of the test. Thus, the requirement is considered satisfied.

```
<uses-sdk
    android:minSdkVersion="26"
    android:targetSdkVersion="36"/>
```

Result	Pass
Comment	

1.6.2 The app only uses software components without known vulnerabilities

1.6.2.1 The app only uses software components without known vulnerabilities

Verification Evidence

After gathering a list of package names and versions with a static scan, we searched for associated publicly known CVEs. We found several package dependencies in use with outdated versions, each vulnerable to at least 1 CVE with a CVSS severity of 7.0 or higher, as detailed below:

play-services-location:19.0.0 - CVE-2022-2390

play-services-places-placereport:17.0.0 - CVE-2022-2390

This does not comply with the Mobile App ADA 1.6.2.1 requirement.

```
[✅] Vulnerability check completed. Results saved in vulnerability_report.txt
```

```
(venv) [redacted]-macbook-pro-2 Tools % cat vulnerability_report.txt
play-services-location:19.0.0 - CVE-2022-2390, CVE-2022-2390
play-services-places-placereport:17.0.0 - CVE-2022-2390, CVE-2022-2390
```

Recommendation: Update the aforementioned dependencies to the latest available non-vulnerable versions. If a non-vulnerable version is not currently available, it is recommended to continuously monitor for updates and apply patches as they become available, in accordance with the established patch management procedures.

References: https://github.com/appdefensealliance/ASA-WG/blob/main/Mobile%20App%20Profile/Mobile%20App%20Test%20Guide.md#1621-the-app-only-uses-software-components-without-known-vulnerabilities

Verification Result

Result	Fail
Comment	

1.6.3 The app validates and sanitizes all untrusted inputs

1.6.3.1 Compiler security features shall be enabled

Verification Evidence

The following shared libraries lack stack canary protection, which makes the application susceptible to stack-based buffer overflow attacks.

arm64-v8a/libmullvad_jni.so

arm64-v8a/libdatastore_shared_counter.so

This does not comply with the Mobile App ADA 1.6.3.1 requirement.

NO	SHARED OBJECT	NX	PIE	STACK CANARY	RELRO	RPATH	RUNPATH	FORTIFY	SYMBOLS STRIPPED
2	arm64-v8a/libmullvad_jni.so Analyze	True info The binary has NX bit set. This marks a memory page non-executable making attacker injected shellcode non-executable.	Dynamic Shared Object (DSO) info The shared object is build with -fPIC flag which enables Position independent code. This makes Return Oriented Programming (ROP) attacks much more difficult to execute reliably.	False high This binary does not have a stack canary value added to the stack. Stack canaries are used to detect and prevent exploits from overwriting return address. Use the option -fstack-protector-all to enable stack canaries. Not applicable for Dart/Flutter libraries unless Dart FFI is used.	Full RELRO info This shared object has full RELRO enabled. RELRO ensures that the GOT cannot be overwritten in vulnerable ELF binaries. In Full RELRO, the entire GOT (.got and .got.plt both) is marked as read-only.	None info The binary does not have run-time search path or RPATH set.	None info The binary does not have RUNPATH set.	False warning The binary does not have any fortified functions. Fortified functions provides buffer overflow checks against glibc's commons insecure functions like strcpy, gets etc. Use the compiler option -D_FORTIFY_SOURCE=2 to fortify functions. This check is not applicable for Dart/Flutter libraries.	True info Symbols are stripped.
3	arm64-v8a/libdatastore_shared_counter.so Analyze	True info The binary has NX bit set. This marks a	Dynamic Shared Object (DSO) info The shared object is build with -fPIC	False high This binary does not have a stack canary value added	Full RELRO info This shared object has full RELRO enabled.	None info The binary does not	None info The binary does not have	False warning The binary does not have any fortified functions. Fortified functions provides buffer	True info Symbols are stripped.

Recommendation: Recompile shared libraries with stack canary protection enabled.

References:

https://www.sans.org/blog/stack-canaries-gingerly-sidestepping-the-cage/

https://github.com/appdefensealliance/ASA-WG/blob/main/Mobile%20App%20Profile/Mobile%20App%20Test%20Guide.md#1631-compiler-security-features-shall-be-enabled

Verification Result

Result	Fail
Comment	

1.6.3.2 The App shall Mitigate Against Injection Flaws in Content Providers

Verification Evidence

Neither manual review nor automated scanning identified any Content Provider–related injection vectors.

```
dz> run scanner.provider.injection -a net.mullvad.mullvadvpn
Attempting to run shell module
Scanning net.mullvad.mullvadvpn...
Not Vulnerable:
  content://net.mullvad.mullvadvpn.FileProvider
  content://net.mullvad.mullvadvpn.FileProvider/
  content://net.mullvad.mullvadvpn.androidx-startup
  content://net.mullvad.mullvadvpn.androidx-startup/

Injection in Projection:
  No vulnerabilities found.

Injection in Selection:
  No vulnerabilities found.
```

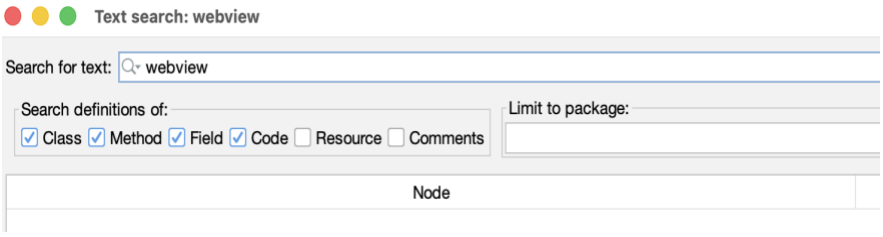
Verification Result

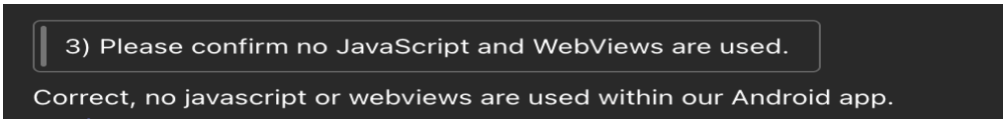
Result	Pass
Comment	

1.6.3.3 Arbitrary URL redirects shall not be included in the app's webviews

Verification Evidence

The application does not use WebViews, which was additionally confirmed by the developer team. Therefore, the Mobile App ADA 1.6.3.3 requirement is not applicable.





Verification Result

Result	NA (Pass)
Comment	

1.6.3.4 Any use of implicit intents shall be appropriate for the app's functionality and any return data shall be handled securely

Verification Evidence

Manual analysis confirmed that no sensitive data is passed via implicit intents and no untrusted return data is processed.

```
[redacted]-macbook-pro-2 sources % grep -Rni "startActivity"
./sources/androidx/lifecycle/s.java:244:                context.startActivity(intentCreateChooser);
./sources/q7/k.java:34:                 ((GoogleApiActivity) this.f15917c).startActivityForResult(intent, 2);
./sources/q7/k.java:41:                 ((p7.s) this.f15917c).startActivityForResult(intent2, 2);
./sources/a/a.java:1208:            context.startActivity(new Intent("android.settings.VPN_SETTINGS"));
./sources/zf/b.java:121:                        mullvadTileService2.startActivityAndCollapse(PendingIntent.getActivity(mullvadTil
eService2.getApplicationContext(), 0, intent2, i14 >= 34 ? 184549376 : i14 >= 31 ? 167772160 : 134217728));
./sources/zf/b.java:123:                        mullvadTileService2.startActivityAndCollapse(intent2);
./sources/u2/t0.java:21:            this.f18692a.startActivity(new Intent("android.intent.action.VIEW", Uri.parse(str)));
./sources/l0/a.java:39:                ((Context) obj).startActivity(className);
./sources/ce/w0.java:56:                    context2.startActivity(new Intent("android.settings.APPLICATION_DETAILS_SETTINGS", Uri.pa
rse("package:" + context2.getPackageName())));
./sources/m/i.java:498:                 iVar.f13450a.startActivity(intent);
./sources/c/m.java:636:     public final void startActivityForResult(Intent intent, int i10) {
./sources/c/m.java:638:         super.startActivityForResult(intent, i10);
./sources/c/m.java:648:     public final void startActivityForResult(Intent intent, int i10, Bundle bundle) {
./sources/c/m.java:650:         super.startActivityForResult(intent, i10, bundle);
[redacted]-macbook-pro-2 sources %
```

Verification Result

Result	Pass
Comment	

1.7 Resilience

1.7.1 The app implements anti-tampering mechanisms

1.7.1.1 The app shall be properly signed

Verification Evidence

The signing certificate indicates that the application is signed and both v2 and v3 signature schemes are enabled, which is acceptable under the given requirement.

```
SIGNER CERTIFICATE

Binary is signed
v1 signature: False
v2 signature: True
v3 signature: True
v4 signature: False
X.509 Subject: C=US, ST=California, L=Mountain View, O=Google Inc., OU=Android, CN=Android
Signature Algorithm: rsassa_pkcs1v15
Valid From: 2020-03-17 13:50:30+00:00
Valid To: 2050-03-17 13:50:30+00:00
Issuer: C=US, ST=California, L=Mountain View, O=Google Inc., OU=Android, CN=Android
Serial Number: 0xa423c545d84d6655de92d8a6f3267e3a2a4ce463
Hash Algorithm: sha256
md5: 7f5cc48563a3832b8f7740baa50a61e9
sha1: 62609928e582b54443c2189e85ead0fe6d7bcb26
sha256: d74ce0e0b29f4d1d57abf5ef7f9a3757e787cca7a6259b9c32bb5bb18e3463bd
sha512: 66d452a0fb9602c1e9662f8c534ca8c75c39ede6880da62d538e2ccfa5a409429d146ac8202baaea4409fe35489fdf63966285bd39ca57c49fad80a3385bfd28
PublicKey Algorithm: rsa
Bit Size: 4096
Fingerprint: a826642ffe03157e4df6706fd36f79431fd07db329268ff195f8ce8e5d696e5c
Found 1 unique certificates
```

Result	Pass
Comment	

1.7.2 The app implements anti-static analysis mechanisms

1.7.2.1 The app shall disable all debugging symbols in the production version

Verification Evidence

No debugging symbols were detected in the application's shared libraries.

```
[redacted]s-macbook-pro-2 arm64-v8a % diff <(nm libmullvad_jni.so) <(nm -a libmullvad_
jni.so)
libmullvad_jni.so: no symbols
libmullvad_jni.so: no symbols
vkachan@vadyms-macbook-pro-2 arm64-v8a % diff <(nm libdatastore_shared_counter.so) <(nm -
a libdatastore_shared_counter.so)
libdatastore_shared_counter.so: no symbols
libdatastore_shared_counter.so: no symbols
vkachan@vadyms-macbook-pro-2 arm64-v8a % diff <(nm libandroidx.graphics.path.so) <(nm -a
libandroidx.graphics.path.so)
libandroidx.graphics.path.so: no symbols
libandroidx.graphics.path.so: no symbols
[redacted]-macbook-pro-2 arm64-v8a %
```

Verification Result

Result	Pass
Comment	

1.7.3 The app implements anti-dynamic analysis mechanisms

1.7.3.1 The app shall not be debuggable if installed from outside of commercial app stores

Verification Evidence

The application's manifest does not have the flag android:debuggable present, which satisfies the given requirement.

```
akita:/ # run-as net.mullvad.mullvadvpn
run-as: package not debuggable: net.mullvad.mullvadvpn
1|akita:/ #
```

Verification Result

Result	Pass
Comment	

1.8 Privacy

1.8.1 The app minimizes access to sensitive data and resources

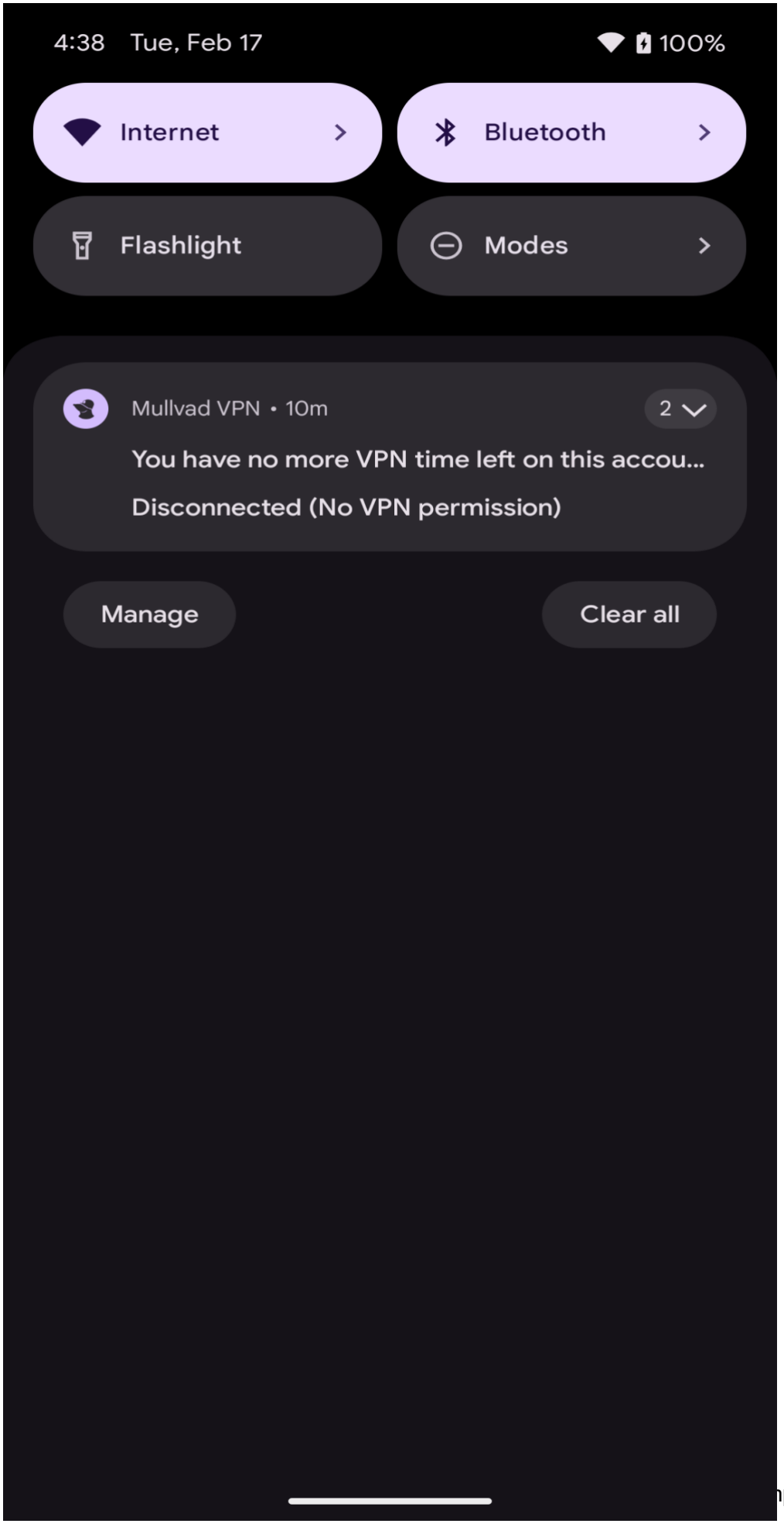
1.8.1.1 The app shall minimize access to sensitive data and resources provided by the platform

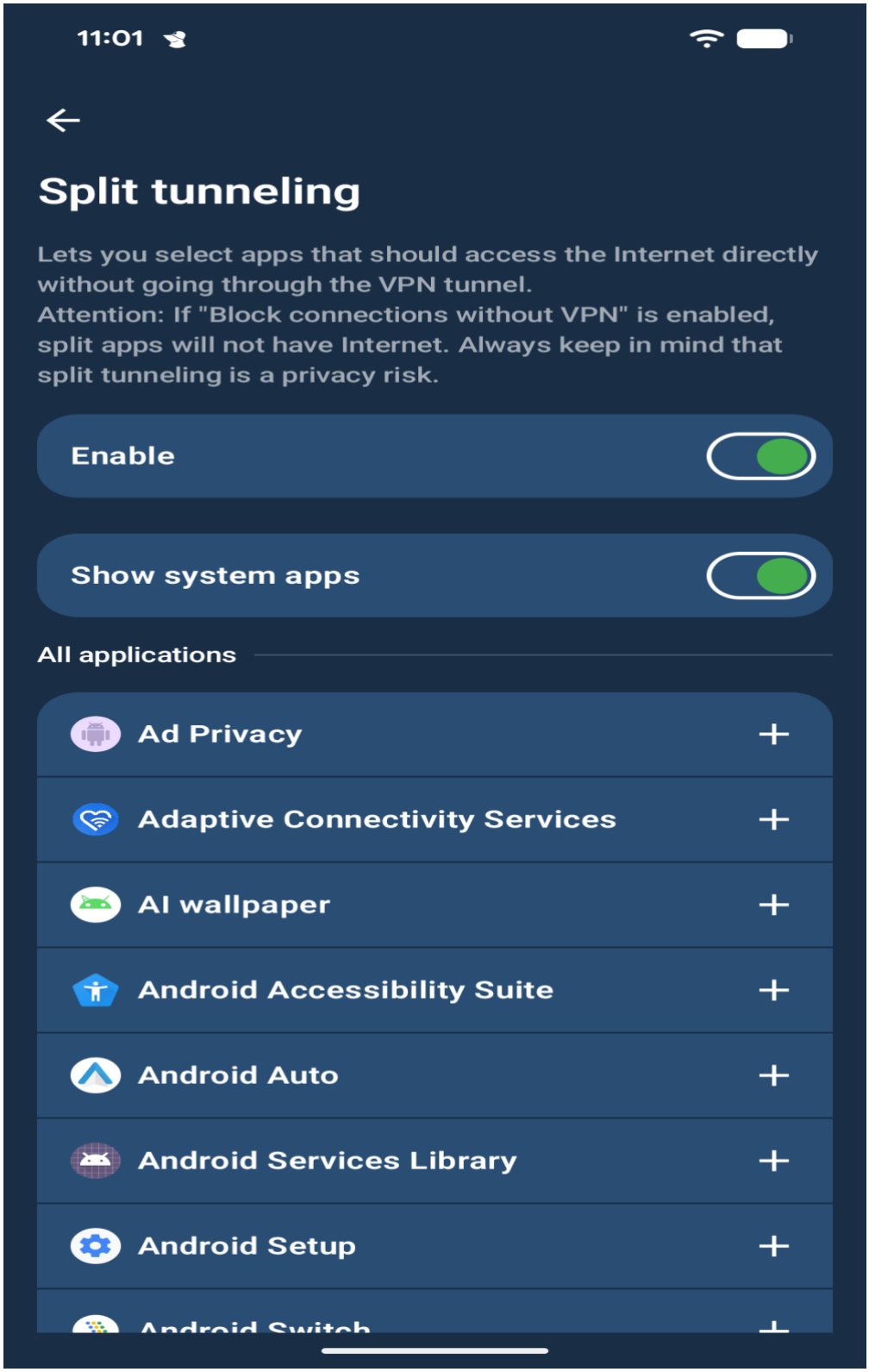
Verification Evidence

Dangerous permissions requested by the application are android.permission.POST_NOTIFICATIONS and android.permission.QUERY_ALL_PACKAGES. They are confirmed to be utilized by the application, as the application posts notifications and has a Split Tunneling feature, where all packages are requested to let the user select which applications should access the internet without the VPN tunnel.

APPLICATION PERMISSIONS

PERMISSION	STATUS	INFO
android.permission.POST_NOTIFICATIONS	dangerous	allows an app to post notifications.
android.permission.ACCESS_NETWORK_STATE	normal	view network status
android.permission.FOREGROUND_SERVICE	normal	enables regular apps to use Service.startForeground.
android.permission.FOREGROUND_SERVICE_SYSTEM_EXEMPTED	normal	allows system-exempted types of foreground services.
android.permission.INTERNET	normal	full Internet access


4:38 Tue, Feb 17
100%
Internet
Bluetooth
Flashlight
Modes
Mullvad VPN • 10m
2
You have no more VPN time left on this accou...
Disconnected (No VPN permission)
Manage
Clear all



Verification Result

Result	Pass
Comment	

1.8.2 The app is transparent about data collection and usage

1.8.2.1 The app shall be transparent about data collection and usage

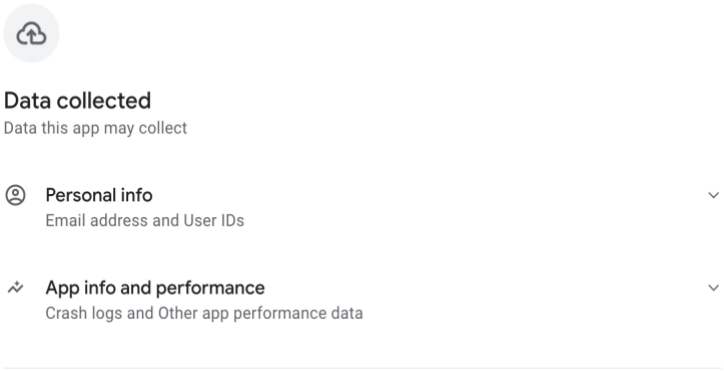
Verification Evidence

We observed that the application's Play Store Data Safety section only specifies the collection of Personal and App Info. However, the privacy policy states the following:

[...]

Certain payment data must be kept for the statutory retention period described in applicable local laws such as the Swedish Accounting Act (some information must be stored for seven years from the end of the fiscal year). If not required by law or stated above, the data will be stored for no longer than necessary for the purpose. After the statutory retention periods, the data will be permanently deleted.

[...]



Recommendation: Revise the Play Store Data Safety section to clearly reflect the data collection practices mentioned in the privacy policy; update it by adding the relevant label specifying the payment information collection practices.

Location: Android application's Play Store Data Safety section

References:

https://support.google.com/googleplay/android-developer/answer/10787469?hl=en#zippy=%2Cdata-types%2Cdata-sharing%2Cdata-collection

https://github.com/appdefensealliance/ASA-WG/blob/main/Mobile%20App%20Profile/Mobile%20App%20Test%20Guide.md#1821-the-app-shall-be-transparent-about-data-collection-and-usage

Verification Result

Result	Fail
Comment	

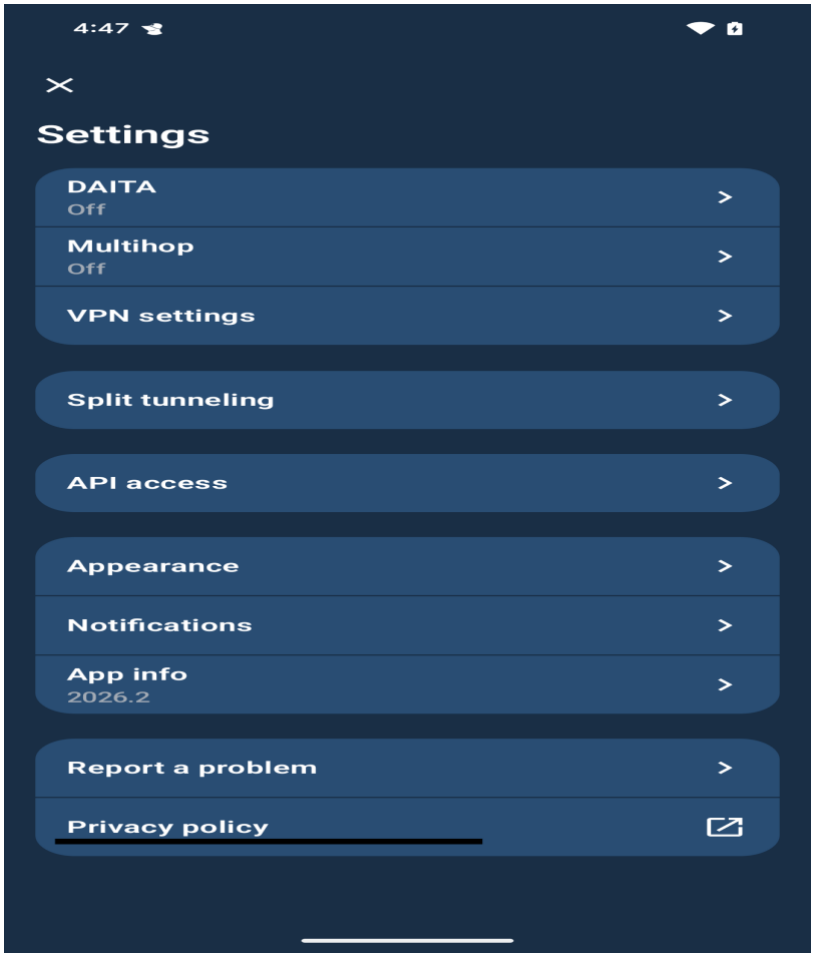
1.8.3 The app offers user control over their data

1.8.3.1 Users shall have the ability to request their data to be deleted via an in-app mechanism

Verification Evidence

We observed that the application does not provide a built-in in-app functionality to delete personal data.

This does not comply with the Mobile App ADA 1.8.3.1 requirement.



Recommendation: Implement an in-app user data deletion mechanism.

References:

https://github.com/appdefensealliance/ASA-WG/blob/main/Mobile%20App%20Profile/Mobile%20App%20Test%20Guide.md#1831-users-shall-have-the-ability-to-request-their-data-to-be-deleted-via-an-in-app-mechanism

Verification Result

Result	Fail
Comment	