NetSPI™

Public Report: Android Quick Share Application Penetration Test

Google

September 2nd, 2025

Prepared for:
David Kleidermacher
Rishika Hooda

Prepared by:
Sam Beaumont
Larry Trowell
Ruchit Patel
Paroksh Sharma
Will Strei

Chapter 1 ▪ Executive Summary

NetSPI performed an analysis of Google LLC's implementation of Quick Share to identify vulnerabilities, determine the level of risk they present to Google, and provide actionable recommendations to reduce this risk. NetSPI compiled this report to provide Google with detailed information on each vulnerability discovered within the application, including potential business impacts and specific remediation instructions.

1.1 Engagement Objectives

NetSPI's primary goal within this engagement was to provide Google with an understanding of the current level of security in the application. NetSPI completed the following objectives to accomplish this goal:

- Identifying application-based threats to and vulnerabilities in the application
- Comparing Google's current security measures with industry best practices
- Providing recommendations that Google can implement to mitigate threats and vulnerabilities and meet industry best practices

1.2 Timeframe, Scope, and Testing Information

1.2.1 Timeframe

Testing and verification were performed between August 09, 2025 and August 17, 2025. On August 28, 2025, remediation testing was carried out for the low-severity finding, and the issue was verified as remediated.

1.2.2 Scope

The scope of this engagement was limited to Google's implementation of Quick Share and risks relevant to the use of the application from supported devices when transmitting and receiving information. All other applications and servers were out of scope. All testing and verification were conducted from outside Google's offices.

1.2.3 Application Binaries

The following table(s) provides details of the application binaries that were in scope for testing:

Description	Value
Application Name	Quick Share
Operating System	Android
Application Package Name	com.google.android.mosey
Version Name	1.0.78857848
Version Code	460
SHA256	fbe6a9aebe571b0ed5a20ba54ef3fe6f3fdbc6ae1cae8530c5d09809b84ea648

Table 1: Android Application Details

1.3 Summary of Findings

NetSPI's assessment of the Quick Share implementation discovered and validated the low-severity issue noted below, which has since been remediated. No additional or incremental issues were identified beyond the known weaknesses inherent to the underlying protocol. Furthermore, NetSPI confirmed that Google's implementation of its version of Quick Share does not introduce vulnerabilities into the broader protocol ecosystem.

The findings are as follows:

- 1 Low severity vulnerability

Vulnerability Name	Severity	OWASP Mobile	OWASP Web	Remediation Status
Information Disclosure - Android - Device Logs	Low	M6-Inadequate Privacy Controls	A5-Security Misconfiguration	Remediated

Table 2: Findings Summary

1.4 Methodology Overview

This section outlines the manual and automated test cases executed during the penetration testing of the Quick Share application. While no high or critical risk findings were identified, the following test cases demonstrate comprehensive testing across various attack surfaces, including interaction with Google Play Services, Quick Share, and permissions.

Check	Description
File Handling to and from Android, iOS and OpenDrop	Validate that file transfers between Android and iOS function correctly, both as sender and receiver, using standard and custom sharing mechanisms such as OpenDrop. Tested successful transmission and reception of different file types (e.g., images, PDFs, text files, zip, xml, corrupted etc) across platforms. Verified integrity and accessibility of files post-transfer.
Data Leakage in Logs	Ensure that sensitive data (e.g., filenames, paths, file contents, session tokens) is not inadvertently written to device logs. Monitored logcat output on Android and system logs on iOS during file sharing operations. Reviewed logs for potential leaks of file metadata or user identifiers.
Secure File Transfer	Confirm that the file transfer mechanism uses encrypted channels and does not expose data in transit. Reviewed application's use of protocols (e.g., Bluetooth, Quick Share), sniffed traffic where applicable, and validated use of TLS or end-to-end encryption.
Path Traversal	Identify any ability to access or write files outside of the intended application directories via crafted file names or paths. Attempted to send and receive files with names like ../../, tested manipulation of file paths during transfer and storage.
Rate Limiting	Check whether the application enforces limits on the number or frequency of file transfer requests to prevent abuse (e.g., DoS). Repeatedly initiated file sharing requests in rapid succession and from multiple devices to see if any throttling, blocking, or timeouts were enforced.
Session Validation	Ensure sessions are valid and authenticated during file sharing, and that expired or invalid sessions are not accepted. Tested transfer attempts during and after session expiration, simulated session hijack scenarios, and verified proper session handling.
CVS	Examined file discovery and transfer capabilities using Quick Share when interacting with iOS devices. Operational in supported scenarios, no informational leak or abuse observed.
Source Code checks	Validate that the implementation aligns with secure coding practices and, in coordination with dynamic testing, ensures that no additional weaknesses are introduced Reviewed SSL configuration, logging operations, examined the use of unsafe functions and blocks in Rust, assessed cargo dependency settings, and analyzed the nature of data transmitted and received between parties.
Hardware checks	The security of the current application version was ensured by validating that previously identified vulnerabilities (CVEs/Issues) reported against AirDrop are not easily replicable. Using Frida, incoming and outgoing data were analyzed to examine potential attack vectors that could be leveraged for lateral or system exploitation, with memory also reviewed to identify any retained information that should be considered sensitive or potentially exploitable. Moreover, gzip parsing(webserver) and XML parsing(plist) were modified with Frida to detect any obvious flaws in these mechanisms, enabling the identification and addressing of potential security issues.

Chapter 2 • Technical Detail

2.1 Overview

The detailed findings section contains the analysis and documentation of the vulnerabilities identified within the application. This analysis included:

- Identifying potential vulnerabilities associated with the application
- Assigning appropriate severity rankings to valid vulnerabilities and risks
- Formulating useful action-based recommendations that can improve the security posture of the IT environment

Vulnerabilities are grouped according to severity. Information for each of the vulnerabilities includes the following:

Name: The name of the vulnerability.

Severity: Each of the vulnerabilities has been assigned a severity based on its CVSS score. The following table summarizes the five severity levels:

CVSS Score	Severity	Description
9.0 – 10.0	Critical	Vulnerability will result in complete compromise of the affected applications or systems. The vulnerability can be exploited remotely by an unauthenticated user.
7.0 – 8.9	High	Vulnerabilities that may result in significant unauthorized access to sensitive data or system or application functionality. Successful exploitation of the vulnerability is likely to require authentication or depends on conditions beyond the attacker's control.
4.0 – 6.9	Medium	Vulnerabilities that may result in partial compromise of the confidentiality, integrity, and availability.
0.1 – 3.9	Low	Security flaws that can contribute to additional attacks against the system or application but do not, by themselves, allow unauthorized access to targeted systems or applications.
0.0	Informational	Security best practices that do not have a direct or immediate impact to system or application security.

Table 3: Severity References

CVSS Score: This field contains the CVSS (Common Vulnerabilities Scoring System) Version 3.1 Base score as well as the scoring vector used to generate the score. Complete documentation of CVSS can be found at http://www.first.org/cvss.

OWASP Web Category: Reference to the OWASP Top 10 web application security risk categories (2021).

OWASP Mobile Category: Reference to the OWASP Mobile Top 10 application security risk categories (2024).

Affected Assets and Services: Specific assets and associated services on which the vulnerability was found.

Vulnerability Details: Comprehensive explanation of the vulnerability that was found, including a high-level summary of how the vulnerability works.

Impact: This describes the potential business impact of vulnerability, should it be exploited.

Recommendation: NetSPI's solution for repairing vulnerability or mitigating the problem if no fix is yet available.

Affected URLs and Parameters: URLs and parameters associated with the finding, if applicable.

Verification: Screenshot or sample data from one instance of the finding showing how NetSPI has verified the finding manually, when possible.

References: These are other resources that have more information on vulnerability.

2.2 Security Review of Protocol Implementation

NetSPI's assessment confirmed that Google's implementation of its version of Quick Share does not introduce vulnerabilities into the broader protocol's ecosystem.

NetSPI's assessment of the implementation of Quick Share revealed that while it shares specific characteristics with implementations made by other manufacturers, this implementation is reasonably more secure. In fact, the process of file exchange is notably stronger, as it doesn't leak any information, which is a common weakness in other manufacturers' implementations.

2.3 Low Severity Findings

2.3.1 Information Disclosure - Android - Device Logs [Remediated]

Vulnerability Unique ID: 3574602898

Severity: Low

CVSS V3 Base Score: 2.1 (CVSS:3.1/AV:P/AC:L/PR:L/UI:N/S:U/C:L/I:N/A:N)

OWASP Web Category: A5-Security Misconfiguration

OWASP Mobile Category: M6-Inadequate Privacy Controls

Affected Assets and Services

Asset
com.google.android.mosey

Vulnerability Details

The Android application logs sensitive data into the system log of the device including image thumbnails, and SHA256 hashes of phone numbers and emails under certain circumstances.

Impact

An attacker with physical access to a user's device may be able to access sensitive information related to the application in the device's system logs. The severity of the attack can vary depending on the context and the type of sensitive information stored in the logs.

Recommendation

Do not log sensitive data. Only log compile-time constants whenever possible. Avoid logs that may contain unexpected or sensitive information depending on the error triggered. Logged data should be limited to necessary and predictable information.

If sensitive data must be included in logs, it's crucial to sanitize them to protect confidentiality. Recommended techniques include tokenization, where a token references sensitive data stored securely; data masking, which alters sensitive information to appear similar but conceals critical details (e.g., changing a credit card number to XXXX-XXXX-XXXX-1313); redaction, which completely hides the information in a field (e.g., replacing a credit card number with XXXX-XXXX-XXXX-XXXX); and filtering, which uses format strings in logging libraries to modify non-constant values.

Verification

The application logged sensitive information to Android device logs, making it accessible to other apps or users with log access.

1. Connect the rooted Android device with a USB cable to a computer that can run the Android Debug Bridge command. Run the below commands to capture the logs while running the application.

```
Command: adb logcat
[TRUNCATED]
08-15 14:58:36.808  5113  6489 V Mosey.ReceiveProvider: Discover request from /[fe80::4c-
cd:6aff:fe34:4a33%mosey0]:50219: {
08-15 14:58:36.808  5113  6489 V Mosey.ReceiveProvider:    "SenderRecordData" =
08-15 14:58:36.808  5113  6489 V Mosey.ReceiveProvider:          <>;
08-15 14:58:36.808  5113  6489 V Mosey.ReceiveProvider: }
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider: Sending discover response: {
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider:    "ReceiverComputerName" = "Pixel 10 Pro";
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider:    "ReceiverMediaCapabilities" =
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider:          <7b0a 2020 2256 6572 7369 6f6e
223a 2033 2c0a 2020 2243 6f64 6563 7322 3a20 7b
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider: 0a 2020 2020 2268 7663 3122 3a20 7b0a
2020 2020 2020 2250 726f 6669 6c65 7322 3a20
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider: 7b0a 2020 2020 2020 2020 2256 5449 7348
4452 416c 6c6f 7765 644f 6e44 6576 6963 65
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider: 22 3a20 6661 6c73 652c 0a20 2020 2020
2020 2022 5654 5375 7070 6f72 7465 6450 726f
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider: 6669 6c65 7322 3a20 5b0a 2020 2020 2020
2020 2020 312c 0a20 2020 2020 2020 2020 20
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider: 32 2c0a 2020 2020 2020 2020 2020 340a
2020 2020 2020 2020 5d2c 0a20 2020 2020 2020
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider: 2022 5654 5065 7250 726f 6669 6c65 5375
7070 6f72 7422 3a20 7b0a 2020 2020 2020 20
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider: 20 2020 2231 223a 207b 0a20 2020 2020
2020 2020 2020 2022 5654 4d61 7844 6563 6f64
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider: 654c 6576 656c 223a 2031 3830 2c0a 2020
2020 2020 2020 2020 2020 2256 5449 7348 61
08-15 14:58:36.811  5113  6489 V Mosey.ReceiveProvider: 72 6477 6172 6541 6363 656c 6572 6174
6564 223a 2074 7275 650a 2020 2020 2020 2020
[TRUNCATED]
08-15 14:58:40.069  5113  5113 V Mosey.ReceiveProvider: Scan result: ScanResult{device=XX:XX:XX-
:XX:C2:16, scanRecord=ScanRecord [mAdvertiseFlags=26, mServiceUuids=null, mServiceSolicitationU-
uids=[], mManufacturerSpecificData={76=[5, 18, 0, 0, 0, 0, 0, 0, 0, 0, 2, 26, 34, 113, -72, -108,
10, -25, 67, 0]}, mServiceData={}, mTxPowerLevel=-2147483648, mDeviceName=null, mTransportDiscov-
eryData=null], rssi=-40, timestampNanos=13385662856852, eventType=16, primaryPhy=1, secondary-
Phy=0, advertisingSid=255, txPower=127, periodicAdvertisingInterval=0}
[TRUNCATED]
08-15 12:39:36.746  5113  6489 D NSDictionaryCallback: Request: POST /Ask Host:
[fe80::5c8f:a4ff:fe8e:3e84%awdl0]:46127
08-15 12:39:36.746  5113  6489 D NSDictionaryCallback: Content-Length: 283
08-15 12:39:36.746  5113  6489 D NSDictionaryCallback: Content-Type: application/octet-stream
08-15 12:39:36.746  5113  6489 D NSDictionaryCallback: Connection: keep-alive
08-15 12:39:36.746  5113  6489 D NSDictionaryCallback: Accept: */*
08-15 12:39:36.746  5113  6489 D NSDictionaryCallback: User-Agent: AirDrop/1.0
08-15 12:39:36.746  5113  6489 D NSDictionaryCallback: Accept-Language: en-us
08-15 12:39:36.746  5113  6489 D NSDictionaryCallback: Accept-Encoding: br, gzip, deflate
08-15 12:39:36.746  5113  6489 D NSDictionaryCallback:
08-15 12:39:36.750  5113  6489 V Mosey.ReceiveProvider: Ask request from /[fe80::40f1:c9ff:fec-
b:9208%mosey0]:58001: {
08-15 12:39:36.750  5113  6489 V Mosey.ReceiveProvider:    "BundleID" = "com.apple.finder";
08-15 12:39:36.750  5113  6489 V Mosey.ReceiveProvider:    "ConvertMediaFormats" = NO;
08-15 12:39:36.750  5113  6489 V Mosey.ReceiveProvider:    "Items" =
08-15 12:39:36.750  5113  6489 V Mosey.ReceiveProvider:          ("tel:+1234567890");
```

Verification continued

```
08-15 12:39:36.750  5113  6489 V Mosey.ReceiveProvider:    "SenderComputerName" = "\Ud83d\Udc68\
U200d\Ud83d\Udc69\U200d\Ud83d\Udc67\U200d\Ud83d\Udc66\Ud83d\Udc68\U200d\Ud83d\Udc69\U200d\Ud83d\
Udc67\U200d\Ud83d\Udc66\Ud83d\Udc68\U200d\Ud83d\Udc69\U200d\Ud83d\Udc67\U200d\Ud83d\Udc66";
08-15 12:39:36.750  5113  6489 V Mosey.ReceiveProvider:    "SenderID" = "b5a1be54ec11";
08-15 12:39:36.750  5113  6489 V Mosey.ReceiveProvider:    "SenderModelName" = "OpenDrop";
08-15 12:39:36.750  5113  6489 V Mosey.ReceiveProvider: }
08-15 12:39:36.751  5113  6489 I Mosey.ReceiveProvider: Incoming transfer from 🔠🔠🔠: id=null
08-15 12:39:36.751  5113  6489 I Mosey.ReceiveProvider: Shared files in Ask request: []
08-15 12:39:36.757  5113  6489 I Mosey.ReceiveProvider: Shared URLs in Ask request: [TextAttach-
ment<id: 0, textBody: tel:+1234567890, type: URL, size: 15, title: tel:+1234567890, isSensitive-
Text: false>]
[TRUNCATED]
```

References

- https://developer.android.com/privacy-and-security/risks/log-info-disclosure
- https://mas.owasp.org/MASTG/best-practices/MASTG-BEST-0002/