Executive Summary

Synopsis

During the summer of 2022, Google engaged **SLNT MISSION** to conduct a security assessment of VPN by Google One. VPN by Google One is a service that increases connection security and privacy to end users. Google provides several clients covering the most widely used operating systems; these VPN clients provide both encrypted transit and IP address dissociation for packets between user's devices and the VPN servers.

The product's security and privacy goals, as stated in the product's whitepaper, are:

- "We focus on three core principles: keeping our users' information secure, treating it responsibly, and putting our users in control."
- "With VPN by Google One, we will never use the VPN connection to track, log, or sell your online activity."
- "A Google-grade VPN that provides additional security and privacy to online connectivity without undue performance sacrifices."

Scope

SLNT MISSION's evaluation included:

- Security Design and Architecture Review
- VPN Library Code Review
- Windows Application Security Assessment
- MacOS Application Security Assessment
- Android Application Security Assessment
- iOS Application Security Assessment

Testing was performed in the production environment, having access to the relevant source code for the tested platforms.

Key Findings

The technical component analysis and source code review uncovered twenty-four initial findings in total, comprising:

- Three findings rated medium-severity.
- Nine findings rated as informational observations.
- Ten findings rated low-severity.

The most notable finding was related to the requirement of the Windows application to be executed with administrator privileges. While **SLNT MISSION** did not find any software vulnerabilities in this application, potential insecure coding practices could result in a privilege escalation attack. This issue was correctly addressed by Google during the retest, and now the application is executed with user privileges.

The other two medium risk findings found were in the login process of both Windows and MacOS applications, which would allow local malicious applications to deny the availability of the service, or obtain the OAuth token sent after a successful login, by manipulating local ports temporarily opened by the applications during the login process.

Recommendations

Although no significant risks were identified in this assessment, it is recommended that the issues outlined in this report are reviewed in line with a suitably robust defense in depth approach which continuously monitors the organization's security posture.

---

1. Reference link
2. Reference link
3. Reference link
4. Reference link
5. Reference link

Vulnerability Breakdown

Severity	Original Assessment	Remaining
Critical	0	0
High	0	0
Medium	0	0
Low	0	0
Informational	0	0

Table of Findings

For each finding, **SLNT MISSION** uses a composite risk score that takes into account the severity of the risk, application's exposure and user population, technical difficulty of exploitation, and other factors.

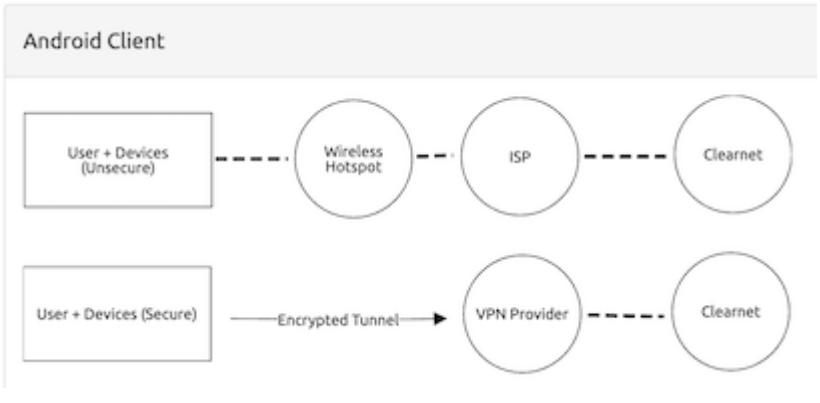
Title	Status	ID	Risk
Lack of Anti-Exploit Protections	Unresolved	SLNT-X-001	LOW
Binaries Contained Debug Information	Unresolved	SLNT-X-002	LOW
Mobile Application Data Storage Leaks GAIA ID in Log Files	Unresolved	SLNT-X-003	LOW

Threat Model

During the first two weeks of the engagement, **SLNT MISSION** performed an assessment of the Google PPN VPN service to ensure its design would be able to facilitate the product's security and privacy goals.

Architecture Description

The VPN by Google One is a product that endeavors to protect users in a way that reduces opportunities for manipulation, interception or analysis of network traffic by third parties in privileged positions.



Scope

SLNT MISSION performed an architecture assessment of both the current PPN environment as well as a future architecture that Google is expecting to begin to deploy in Q3 of 2022. During this assessment, **SLNT MISSION** created three architecture diagrams. The first two diagrams document the current environment, with the first including the client portions and the second focused on the server-side architecture. The third diagram documents the future release.

Components

- G1 APP: The application responsible for hosting the VPN service and initializing the PPN library. It also provides notifications and account management.
- PPN Service: Java subclass of Android VPN service that runs in the same memory space as the host app. Responsible for the actual implementation of the Android VPN APIs.
- Xenon: The PPN network-switching layer. This library talks directly to Android and is responsible for switching Krypton's data plane between Wi-Fi and cell networks.
- Krypton Service and library: PPN C++ library that implements data plane of the VPN, talking directly to Datapath session manager Copper.
- Datapath Client: Open-Source software application and communication protocol that implements VPN techniques to create secure point-to-point connections in routed or bridged configurations.

Server Components

- Brass Backend: Dataplane manager and key management service. It provides the public key of the data node. According to the request made from client device, it determines the specific dataplane node to use, programs the node with the client device's public key and receives a new public key the dataplane generates for return to the client device.
- Copper(Session Manager): The exit node responsible for sending and receiving the packets from the device and the internet.
- Zinc Backend: The current authentication server that is responsible for authentication and authorization of the PPN service where it proxies GAIA authentications to Bronze.
- Keyring Bucket: Responsible for serving the public key, signing and verification requests.

- Bridge-Proxy Backend: talks to Bridge server to get valid Bridge token and sends to the bridge controller.
- Bridge-Server: Provides a valid token to the bridge-proxy.
- Bridge Controller: Handles requests to program the packet Processor.
- Phosphor: New closed source authentication server that will replace Zinc and Bronze in

a future release. It provides access to the public key, authentication, signing and eligibility remote procedure calls (RPCs).

- Bronze: Closed source authentication server that is responsible for GAIA and G1 service

authorizations which Zinc talks to and validates from.

- Attestation Service: A service running within the Phosphor instance that handles device

attestation from Android and iOS devices.

- DB Spanner: Used to store the nonces that will be signed by the Attestation Service.
- G1 Benefits: Tells if the user has a PPN subscription.
- Play: Responsible for device attestation (Android).
- GAIA: Responsible for service authorization.
- Packet Processor: Provides a platform for low-level network packet processing

applications.

General Conclusions

At the end of the review, **SLNT MISSION** concluded that the PPN design allows Google to implement user authentication and authorization for the service in a way that isolates the user's Google identity (referred to internally as Gaia ID) from the VPN session network flows. The use of cryptographic blind signing during authorization is the traffic anonymization strategy, protecting user's identity from direct association with the VPN session token. However, as the privacy threat model considers Google itself as an adversary in a privileged position, this review also identified several techniques that could be employed to compromise user anonymity should Google choose or be compelled to actively violate its privacy claims.

It should be noted that none of these techniques were observed to be part of the product's strategy or implementation. Furthermore, the migration planned with the induction of Phosphor server include a specific component known as Attestation Service with the specific purpose of refusing authentication if the client application has been manipulated

Vulnerability Details - Mobile Application

Lack of Certificate Pinning - (Low)

Risk	Impact	Exploitability	Finding ID	Component	Category	Status
Low	Medium	Low	SLNT-X-001	Android Application	Cryptography	Reported

Impact

TLS traffic between the application and the server can be intercepted if a trusted certificate authority is compromised; or if an attacker is able to install a malicious certificate on the user's device and has a privileged network position.

Description

The authentication communications with the PPN service did not implement certificate pinning. This is a security feature which involves hard-coding the expected TLS certificate of the server (or a particular certificate authority) into the application, rather than relying on the certificate chain validation function offered by the underlying platform and the PKI infrastructure. This mitigates the risk from various active attacks which could be performed against the application's TLS connection, and lead to attackers being able to intercept the application's communications.

In particular, the use of certificate pinning mitigates the risk associated with one of the device's trusted certificate authorities becoming compromised. Although this has happened on several occasions in recent years2,3 , certification authorities are required to follow strict security standards, so these kind of attacks are usually performed by state sponsored or highly profile threat actors.

Recommendation

In order to further secure communications and information handled by the application, it is recommended to implement certificate pinning to mitigate the risk of interception when a certification authority is compromised. Since Android 7.0 (SDK 24), applications can use the Network Security Config4 to define a list of trusted certificate hashes without manual checking being necessary. Information about this mechanism is available on Android's developer documentation5.

For applications which need to support older devices, consider using a library with built-in support for pinning. For example, Square's OkHttp library enables pinning with a few lines of code6. Consider also pinning intermediate or root certification authorities instead of individual host certificates, since it reduces the risk associated with certificate handling but still provides a strong protection, as the attack surface is reduced to the specific CA pinned, and not the whole PKI infrastructure.

Reproduce

1. Install a custom system CA in the mobile device (for devices with SDK >= 24), or user CA (for devices with SDK < 24)
2. Intercept the application's SSL traffic to pass through an interception proxy

3. Verify that TLS traffic can be decrypted

---

1. DigiNotar - Issuance of fraudulent certificates
2. Comodo - Certificate hacking
3. Android Developers - Network Security Configuration
4. Android Developers - CertificatePinning
5. OkHttp library

Vulnerability Details - Mobile Application

Missing Permissions on Android Receivers - (Info)

Risk	Impact	Exploitability	Finding ID	Component	Category	Status
Informational	Low	None	SLNT-X-001	Mobile Application	Access Controls	Reported

Impact

Other android applications installed on the device can interact with the exported services. However, it is not exploitable due to checks inherited from the tiktok libraries.

Description

The PPN package of the G1 Android Application exported two broadcast receivers that were not protected by Android permissions. In normal conditions, this would allow other applications installed on the Android device to interact with these receivers, causing unwanted behavior. However, the receivers expected protected intents that can only be sent by system services, and inherited methods from the tiktok libraries that ensured that the action of the intent matched the ones in the filter, avoiding the exploitation of this issue.

The following two (2) broadcast receivers were exported in the Android manifest file:

```
<receiver
    android:exported="true"
    android:name=".PackageReplacedPpnStateCheckReceiver_Receiver">
    <intent-filter>
        <action android:name="android.intent.action.MY_PACKAGE_REPLACED"/>
    </intent-filter>
    </receiver>
    <receiver
        android:exported="true"
        android:name=".BootCompletedPpnStateCheckReceiver_Receiver">
    <intent-filter>
        <action android:name="android.intent.action.BOOT_COMPLETED"/>
    </intent-filter>
</receiver>
```

The intent filters declared define that the expected actions are `MY_PACKAGE_REPLACED` or `BOOT_COMPLETED`. Both intents are protected and can only be broadcasted by Android system components, and therefore other applications can't send these.

The source code of the components called startPpnIfUserEnabled , which received the action sent as an argument. However, this function only checked that the action was one of the expected ones to log an event and would continue the execution even with other actions.

```
if (!Strings.isNullOrEmpty(intentAction)) {
    switch (intentAction) {
        case "android.intent.action.MY_PACKAGE_REPLACED":
            clearcutLogger.logEvent(GoogleOneClientEventType.PPN_START_ON_PACKAGE_REPLACED);
        break;
        case "android.intent.action.BOOT_COMPLETED":
            clearcutLogger.logEvent(GoogleOneClientEventType.PPN_START_ON_BOOT_COMPLETED);
        break;
        default:
            break;
    }
}

return PropagatedFutures.transformAsync(ppnStateController.getEligiblePpnAccountId(), this::startPpn,
directExecutor());
```

This code would have allowed other applications installed in the device to directly interact with these receivers by sending explicit intents and other arbitrary actions. However, it was found that the receivers used the `IntentFilterAcledReceiver` class of the tiktok library, that checked if the intent action was expected by the intent filters, raising an exception when the action did not match with the intent filter.

Recommendation

As the tiktok IntentFilterAcledReceiver class is protecting the receivers from receiving unwanted intent actions, no action is needed. However, consideration should be given to implement android permissions for these components.

Reproduce

N/A.

1. Reference link
2. Reference link
3. Reference link
4. Reference link
5. Reference link

Vulnerability Details - Mobile Application

User Email Address Stored Without Application-Level Encryption - (Info)

Risk	Impact	Exploitability	Finding ID	Component	Category	Status
Informational	Low	Low	SLNT-X-001	Android Application	Data Exposure	Reported

Impact

Unencrypted data is at risk of being exposed in rooted devices

Description

The application did not use any application-level encryption mechanism to store the user email address on shared preferences files in the application data folder.

In non-rooted devices, as the android backup was not enabled, other applications could not access the application data folder due to file system permissions. However, the application could be run on rooted devices and other applications could ask for permission to run code as root. With this permission, the contents of any file on the device could be read.

Recommendation

Consider encrypting files and SharedPreferences using the Jetpack Security library7 where possible. The Jetpack Security library is provided by the Android team to enable secure, standardized encryption mechanisms based on the Android Keystore system8, and to ease the transition for developers.

Reproduce

After enabling the VPN on the device, execute the following command on a device's root shell:

```
cat
/data/data/com.google.android.apps.subscriptions.red/shared_prefs/com.google.android.libraries.privacy.ppn.Settings

<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="AccountName">userEmail@gmail.com</string>
</map>
```

1. Android Developers - Work with data more securely
2. Android Developers - Android Keystore system

Vulnerability Details - Mobile Application

Vulnerability Title - (Severity)

Risk	Impact	Exploitability	Finding ID	Component	Category	Status
Informational	Low	Low	SLNT-X-001	Android Application	Data Exposure	Reported

Impact

Description

Recommendation

Reproduce

```
Insert code snippet
```

1. Reference link
2. Reference link

3. Reference link
4. Reference link
5. Reference link

Vulnerability Details - Mobile Application

Vulnerability Title - (Severity)

Risk	Impact	Exploitability	Finding ID	Component	Category	Status
Informational	Low	Low	SLNT-X-001	Android Application	Data Exposure	Reported

Impact

Description

Recommendation

Reproduce

```
Insert code snippet
```

---

1. Reference link
2. Reference link
3. Reference link
4. Reference link
5. Reference link

Vulnerability Details - Mobile Application

Vulnerability Title - (Severity)

Risk	Impact	Exploitability	Finding ID	Component	Category	Status
Informational	Low	Low	SLNT-X-001	Android Application	Data Exposure	Reported

Impact

Description

Recommendation

Reproduce

```
Insert code snippet
```

---

1. Reference link
2. Reference link
3. Reference link
4. Reference link
5. Reference link

Field Finding Definitions

The following sections describe the risk rating and category assigned to issues **SLNT MISSION** identified.

Risk Scale

SLNT MISSION uses a composite risk score that takes into account the severity of the risk, application's exposure and user population, technical difficulty of exploitation, and other factors. The risk rating is **SLNT MISSION's** recommended prioritization for addressing findings. Every organization has a different risk sensitivity, so to some extent these recommendations are more relative than absolute guidelines.

Rating	Description
Critical	Implies an immediate, easily accessible threat of total compromise.
High	Implies an immediate threat of system compromise, or an easily accessible threat of large-scale breach.
Medium	A difficult to exploit threat of large-scale breach, or easy compromise of a small portion of the application.

Rating	Description
Low	Implies a relatively minor threat to the application.
Informational	No immediate threat to the application. May provide suggestions for application improvement, functional issues with the application, or conditions that could later lead to an exploitable finding.

Impact

Impact reflects the effects that successful exploitation has upon the target system or systems. It takes into account potential losses of confidentiality, integrity and availability, as well as potential reputational losses

Rating	Description
High	Attackers can read or modify all data in a system, execute arbitrary code on

the system, or escalate their privileges to superuser level.| |Medium|Attackers can read or modify some unauthorized data on a system, deny access to that system, or gain significant internal technical information.| |Low|Attackers can gain small amounts of unauthorized information or slightly degrade system performance. May have a negative public perception of security.|

Exploitability

Exploitability reflects the ease with which attackers may exploit a finding. It takes into account the level of access required, availability of exploitation information, requirements relating to social engineering, race conditions, brute forcing, etc, and other impediments to exploitation.

Rating	Description
High	Attackers can unilaterally exploit the finding without special permissions or significant roadblocks.
Medium	Attackers would need to leverage a third party, gain non-public information, exploit a race condition, already have privileged access, or otherwise overcome moderate hurdles in order to exploit the finding.
Low	Exploitation requires implausible social engineering, a difficult race condition, guessing difficult-to-guess data, or is otherwise unlikely.

Category

SLNT MISSION categorizes findings based on the security area to which those findings belong. This can help organizations identify gaps in secure development, deployment, patching, etc.

Category Name	Description
Access Controls	Related to authorization of users, and assessment of rights.
Auditing and Logging	Related to auditing of actions, or logging of problems.
Authentication	Related to the identification of users.
Configuration	Related to security configurations of servers, devices, or software.
Cryptography	Related to mathematical protections for data.
Data Exposure	Related to unintended exposure of sensitive information.
Data Validation	Related to improper reliance on the structure or values of data.
Denial of Service	Related to causing system failure.
Error Reporting	Related to the reporting of error conditions in a secure fashion.
Patching	Related to keeping software up to date.
Session Management	Related to the identification of authenticated users.
Timing	Related to race conditions, locking, or order of operations.

Tip: Use blue boxes (alert-info) for tips and notes.

Success: This alert box indicates a successful or positive action.

Example: Use yellow boxes for examples that are not inside code cells, or use for mathematical formulas if needed. Typically also used to display warning messages.

Danger: This alert box indicates a dangerous or potentially negative action.