VPN by Google One: Technical Security & Privacy Assessment

Google

April 4, 2021 – Version 1.0

Prepared by

Eric Schorn
John McGuiness
Tyler Colgan
Paul Bottinelli
Yujing Qiao
Tapan Singh

Table of Contents

Executive Summary

Synopsis

During the fourth calendar quarter of 2020 and the first calendar quarter of 2021, NCC Group conducted an in-depth review of the VPN by Google One virtual private network system. The focus of the engagement was to assess the product's technical security properties and review its associated privacy claims.

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."

The technical review initially identified fourteen findings. By the conclusion of the assessment and remediation period ten findings were fully fixed, one partially fixed, and the remainder were considered acceptable risks by Google. NCC Group found the resulting system to have a robust implementation and a strong technical security posture. The consulting team determined that diligent coding and development practices, integration with Android's VPN application programming interface (API), use of standardized VPN profiles and well-studied cryptographic protocols enable Google to provide a VPN which benefits its users with security enhancements for their network traffic.

To deliver on its privacy claims, the product introduces supplemental cryptographic measures to disassociate Google user identities from tunneled VPN traffic. While these measures do not categorically eliminate the opportunity for Google to circumvent its privacy claims, they do provide a structural framework within which the application can provide authentication and authorization for users without sending identifying information to the VPN exit nodes.

Project Logistics

The engagement occurred in multiple phases across three months and involved six specialist consultants. Open-source client-side code was made available via public GitHub repositories, while selected private server-side code components were transferred via the NCC Group Secure File Exchange server. For each component in scope, NCC Group provisioned a staff cryptography specialist to review Google's use of cryptography as a foundational component of the product's security and user privacy.

The scope of the assessment included the following primary components:

- Architecture review of the product's design from a holistic perspective
- Source code review of selected product components
  - Public client-side VPN library code from Google's repository on GitHub
  - User identity-aware server-side Google One authentication & authorization code, referred to as "Zinc"
  - User identity-blinded server-side VPN code, referred to as "Brass" and "Copper"
- Mobile application security and privacy assessment of the Google One Android application
- Network penetration test of the external VPN data-path infrastructure

Assessment Methodology

The engagement process involved assessing the components listed above to varying levels of assurance. The following methodologies were used:

- Manual and tool-assisted source code review
- Dynamic testing
- Design document review
- Penetration testing
- Engineering team interviews

The general strategy for analyzing the technical security of a particular component involved prioritizing the most sensitive functionality with the most productive method of assurance, partially constrained by the sensitivity of Google's proprietary internal infrastructure. As an example, NCC Group performed source-code analysis of the VPN library and

associated server-side code, rather than instrumenting and fuzz-testing the underlying communications.

For the purposes of the privacy claim assessment, the approach to each component's analysis was based on how significant its derived conclusions were to the overall goals as informed by a threat modeling exercise within the architectural review. As an example, NCC Group performed interviews regarding operational processes instead of server build audits on the VPN data-path hosts because the lack of continuous third-party build validation theoretically allows Google to modify its future deployment builds.

Project Limitations

This engagement represents a point-in-time evaluation of the VPN by Google One product. Security threats and attacker techniques evolve rapidly, and the results of this assessment are not intended to represent an endorsement of the adequacy of current security measures against future threats. Furthermore, statements made by NCC Group refer to the system as presented during the assessment window, and provide no assurance with regards to any future chosen or compelled technical changes or deviations in policy.

As noted above, a subset of components were assessed based on the attestation of Google staff and product design documents taken at face value. While Google supplied suitable support in that regard, NCC Group cannot attest to the accuracy of the information or associated conclusions where an in-depth technical resource was not available. Finally, application-level privacy, such as web browsers, was considered out of scope. See the Technical Component Analysis on page 11 for more detailed information on which methodology was used for a given component.

Key Findings

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

- One finding rated high-severity.
- Four findings rated medium-severity.
- Six findings rated low-severity.
- Three findings rated as informational observations.

The development team subsequently delivered updated code and all fourteen findings were retested. Ten findings were found to be 'Fully Fixed'. Of the remaining four:

- One medium-severity finding, requiring a protocol change that would increase communication steps, was considered to be an acceptable risk by Google.
- One low-severity finding was considered to be an acceptable risk by Google.
- One informational observation was confirmed to be 'Partially Fixed' while a second was considered to be an acceptable risk by Google. *Note that informational observations do not indicate immediate security concerns*.

Additionally, NCC Group identified various ways that Google could theoretically circumvent its supplemental protections and privacy claims in the future to re-associate tunneled traffic with a user identity. These fall into four categories:

- Manipulation of the client application: In the future, Google could update the client application to facilitate attribution of traffic to users.
- Instrumentation of client device traffic: After a VPN tunnel is established, Google could reassociate a tunnel with a Google identity by generating specific, identifiable network traffic via other popular services it operates.
- Manipulation of cryptographic protocols: Google may be able to serve chosen cryptographic keys to users during the authorization flow which may subsequently allow VPN tunnel traffic to be re-associated with a Google user identity at later stages.
- Server-side analysis of network traffic: Google could correlate networking metadata or cleartext traffic to develop strong associative metrics between an identity and tunneled traffic.

While it wouldn't be possible to continually prove the absence of this type of activity, NCC Group did not observe any of these within the project's strategy or implementation artifacts. Furthermore, NCC Group did not observe logging inconsistent with whitepaper claims. For more detailed information on the architecture review and associated findings, see System Architecture Review on the next page.

Introduction

NCC Group performed an architectural design review of the VPN by Google One product during the first week of the engagement. The objective of this part of the review was to determine whether the system, as designed, would be able to fulfill the security goals and privacy claims outlined in the product's whitepaper. The consulting team considered the product from the perspective of multiple classes of threat actors, including external parties, such as nation state attackers and malicious ISPs, as well as internal parties such as rogue employees. Furthermore, NCC Group evaluated Google's capacity as an organization to circumvent its own supplemental technical protections that support the product's privacy initiatives.

Product Security Goals

The system's technical security goals were aligned with the traditional responsibilities of a VPN provider. It endeavors to protect users in a way that reduces opportunities for interception, manipulation, or analysis of network traffic by third parties in privileged positions.

The product whitepaper illustrates the contrast between a typical network connection and one protected by a VPN with the diagram shown in figure 1 below.

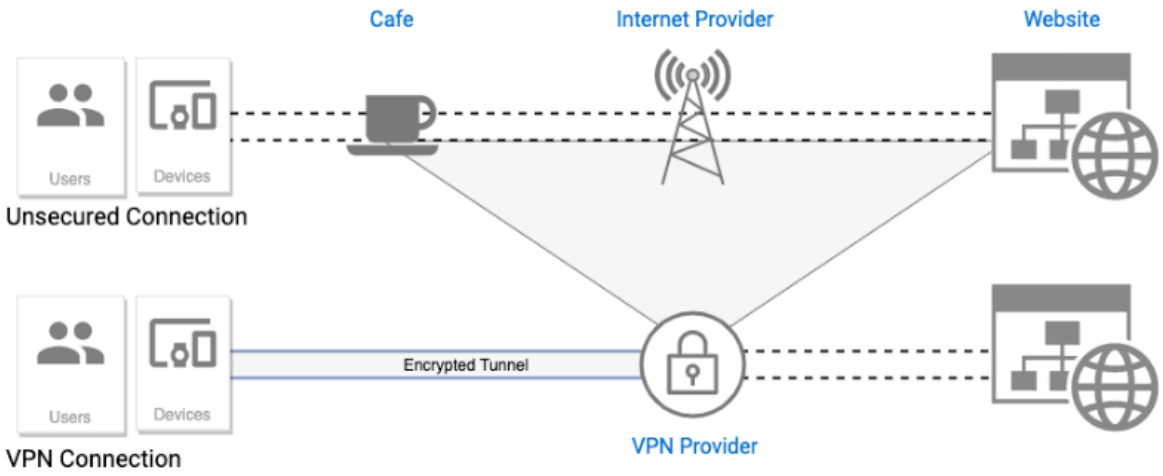

Figure 1: Contrast between a typical network connection and one involving a VPN from the Google whitepaper

The system itself must be designed and implemented securely to ensure the benefits cannot be circumvented, and such that the VPN provider does not introduce additional risk to itself or its users.

Product Privacy Claims

Per the product whitepaper, Google attests that it will adhere to an explicit policy for the privacy of VPN user data as follows:

> The following data will not be logged by the VPN for a given user:
>
> - Network traffic, including DNS
> - IP addresses of the devices connecting to the VPN
> - Bandwidth utilized by an individual user
> - Connection timestamps by user
>
> The following data will be logged by the VPN in aggregate such that individual users' data cannot be identified:

- Aggregate throughput
- Aggregate VPN tunnel uptime
- Aggregate VPN tunnel setup latency
- Aggregate total bandwidth rate
- Aggregate packet loss rate
- Aggregate VPN tunnel failure rates
- Aggregate VPN tunnel retries
- Aggregate service/server CPU and memory load
- Aggregate VPN tunnel setup error rates

Additional logging may occur in an effort to "measure overall service experience, debug the service, and prevent fraud without compromising user privacy". This logging is further described by the product whitepaper to include:

- Use of the service in the last 28 days. This metric collects how often the service was used in the last 28 days but does not track the specific times they used the service nor the duration of the usage nor the amount of data used.
- Number of recent attempts by a user to set up a VPN session to ensure that the user does not exceed the maximum number of allowed concurrent sessions. User IDs are encrypted and therefore cannot be personally identified by the concurrent session check.
- Server error logs without request or response data

Lastly, the product whitepaper notes that additional data may be collected for users that choose to share feedback or errors with Google: "The client application provides users the option to send Google application and system logs from their device which contain personally identifiable information (such as email) and is used for debugging purposes. This is an optional capability and requires user permission every time they wish to submit such information".

In addition to traditional privacy enhancements provided by VPNs, Google has further incorporated technical and procedural protections with respect to itself and its employees as threat actors. The technical protections leverage cryptographic blind-signing to disassociate the user identity from subsequent tunneled traffic and are illustrated in figure 2 below.

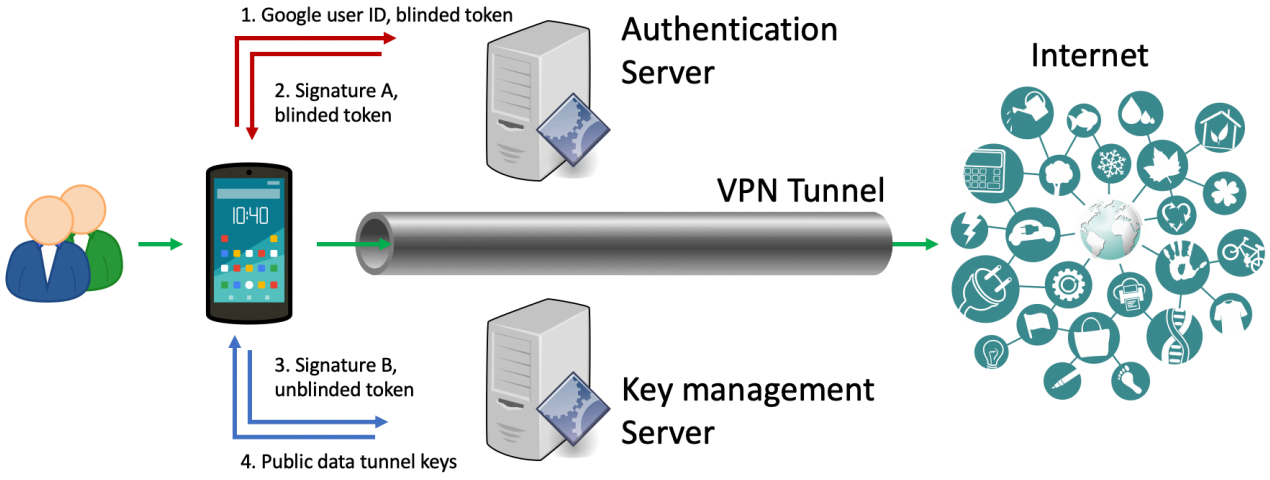

Figure 2: Blind signing separation between authentication servers and data-path servers

Google has also demonstrated, via publications and engineer interviews, that it has procedures in place to limit its capability to circumvent these policies. Discussion of these procedures is included in the Findings subsection below.

System Design Diagram

The following diagram demonstrates a high-level representation of how the product's components interact:

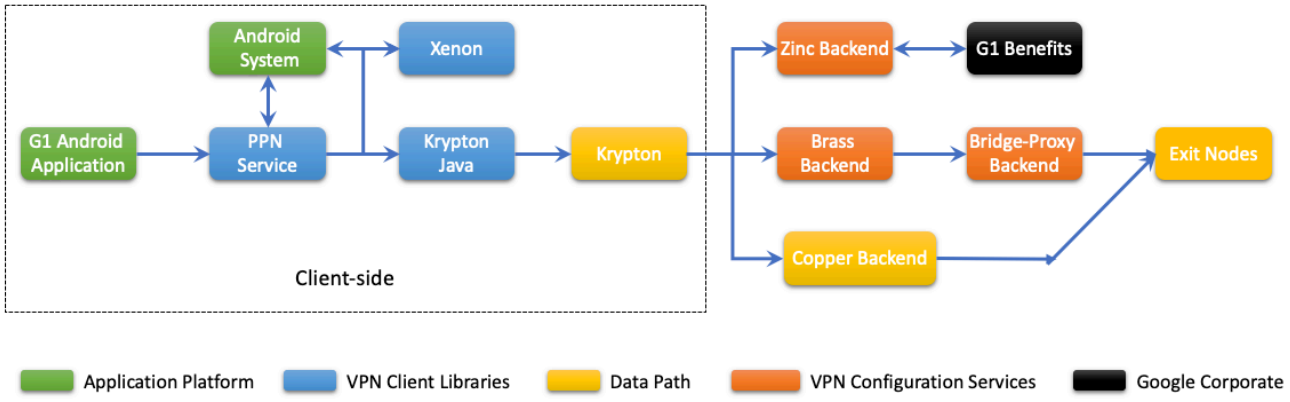

Figure 3: VPN Architecture Overview Diagram

Each component is described briefly below, and in greater detail in Technical Component Analysis on page 11.

Client Components

- G1 Android application: The Google One application for Android, as delivered by the Play Store
- Android system: The set of Android system libraries which support VPN application functionality
- PPN service: The main service for managing VPN functionality via the Google One application
- Krypton Java: The proxy between the PPN service and Krypton C++ libraries
- Xenon: The library for managing networking calls to Android system
- Krypton C++: Primary handler for communications between Android client and Google servers

Server Components

- Zinc backend: The user identity-aware service for authorizing VPN feature and for performing blind-signing
- Google corporate: The backend Google infrastructure which manages information about user identity
- Brass backend: The user identity-unaware service for setting up VPN connection parameters
- Bridge-proxy backend: The Google infrastructure for distributing VPN connection parameters to data-path servers
- Copper backend: The data-path server which receives VPN connection from client
- Exit nodes: The Google infrastructure which distributes tunneled traffic out to its destination

Cryptographic Principles

The Google One VPN improves privacy by separating **usage**-specific information known by the VPN data-path subsystem from **user**-specific information known by the VPN authentication subsystem via advanced cryptographic functionality. This separation inhibits the opportunity for collusion between authentication and data-path subsystems over the session information to associate VPN users with their VPN traffic. The cryptographic functionality extends the traditional RSA signature algorithm with blinding properties as described below.

Traditional RSA Signatures

The RSA public-key cryptosystem provides a very mature, well-studied and widely used foundation for implementing digital signatures.[1] Simplistically, with a correctly generated public key consisting of a public modulus N and a public exponent e, the signer's private key d and a message m interpreted as an integer, the cryptosystem revolves around the central fact that $m = m^{d \cdot e} \ (mod\ N)$. In other words, if m is exponentiated by d (then reduced $mod\ N$) into

[1] Section 5 of FIPS PUB 186-4 Digital Signature Standard (DSS) https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf

ciphertext and that ciphertext is then exponentiated again by e (then reduced $mod\ N$), the original message m takes a round trip back to itself. To create a signature over a message m, a signer uses their private key d to calculate the signature as $sig = m^d\ (mod\ N)$. To subsequently validate this signature sig over message m, the validator uses the public exponent e to confirm that $sig^e\ (mod\ N)$, which corresponds to $m^{d \cdot e}\ (mod\ N)$, does in fact match m. As an aside, note that m is typically a padded hash of an original/larger message M, but this step is extraneous to the above math; In this context the message m will be derived from session IDs.

Consider a hypothetical VPN system where the client first submits correct user credentials along with a random session ID SID to an authentication server. If the credentials are acceptable, the authentication server returns the session ID SID along with a signature sig signifying authorization. The client subsequently presents the **same** session ID SID and signature sig to a (separate and different) data-path server which provides access upon signature validation. In this VPN system, both the authentication server and the data-path server see the exact same session ID (and signature), so they have an opportunity to collude and associate VPN users to their VPN usage based on this information.

Blinded RSA Signatures

The concept of blind signatures was introduced in 1983 by David Chaum[2] to better separate a message signature from the message contents. The concept is typically illustrated with a sealed envelope made of carbon paper, where a signature on the outside is transferred to an unknown message inside. The blinded RSA signature cryptosystem allows the client to separate and decouple the session-related information known by the authentication server from that known by the data-path server. The client first generates an additional private random value r (co-prime to N) and exponentiates it by the authentication server's public exponent e to calculate a blinding factor $r^e\ (mod\ N)$. The client multiplies the original session ID with this blinding factor prior to sending the result to the authentication server as $SID_{auth} = SID_{orig} \cdot r^e\ (mod\ N)$. The authentication server signs SID_{auth} by exponentiating it by its private key d as usual, so that the returned signature is $sig_{auth} = SID_{auth}^d = SID_{orig}^d \cdot r^{e \cdot d}\ (mod\ n)$. This corresponds to signing the carbon-paper envelope; note that the latter r takes a round trip back to itself. So, multiplying the authentication sig_{auth} by the inverse of r gives us a sig_{orig} that corresponds to SID_{orig} which could be considered as having been inside the envelope. The authentication server never sees the latter two values, but they are fully valid for use with the data-path server.

For the Google One VPN system, the above strategy allows the client to receive a signature sig_{auth} over a blinded session ID SID_{auth} from the authentication server. The client is able to unblind that signature into sig_{orig} such that it then becomes valid for the original yet-unseen session ID SID_{orig}. This latter **different** signature and session ID are then submitted to the data-path server to begin tunnel configuration. This effectively breaks the information linkage between the authentication server and the data-path server, which prevents their colluding to associate VPN users with their VPN traffic over session IDs and signatures.

Observations

There are a variety of means by which, in the absence of Google's business processes and controls, a group of rogue employees or the organization could circumvent its supplemental privacy protections, should they choose or be compelled to break the commitments described in the product white paper. Each subsection below represents a broad category of capability, and includes a specific representative example of a technique that could be leveraged in practice. It should be noted that there are likely other distinct techniques within these categories.

NCC Group did not observe any of these techniques to be part of the product's design or implementation artifacts. Furthermore, Google demonstrated business processes which would impede malicious employees or individual business units from making unauthorized changes. During the audit, NCC Group organized meetings to discuss deployment procedures implemented by Google which reduce the risk of insecure or non-compliant code being introduced to the product. Google attested that the following processes were in place:

- Code review: "At Google, our code reviews include inspection and approval from at least one engineer other than the author. Our code review process enforces a rule that, at a minimum, code modifications to any system must be

[2] https://sceweb.sce.uhcl.edu/yang/teaching/csci5234WebSecurityFall2011/Chaum-blind-signatures.PDF

approved by the owners of that system. Once the code is checked in, it is built."
- Cryptographically verified build system: "For each build, the system produces a verifiable build manifest – a signed certificate fully describing the sources that went into the build, the cryptographic hashes of any binaries or other build artifacts, and the full build parameters."
- Job identity: "For a service to access certain data or another service, its identity must have the necessary permissions."
- Immutable audit trails: "All administrative actions taken within the product production infrastructure environment are logged in a way which cannot be modified by the product team."

The statements above were made by the product engineering team, or directly retrieved from Google's publication on binary authorization. In both cases, during interviews the Google team attested that they apply to VPN client code and infrastructure.

Capability Categories

Manipulation of the client application

Summary of capability: In the future, Google could update the client application to facilitate attribution of traffic to users.

Example: Google could update the Krypton library on the client to report IPsec VPN connection parameters to Google servers along with the user's identity.

Analysis: Users of any commercial VPN should be aware that this is always a risk when using third-party software. Providers that distribute their own software package, particularly those which automatically update, have the potential to introduce unwanted changes into their software distributions. Active research is underway to address this risk through binary transparency.[3]

Additionally, should Google be compelled by a legal authority to analyze the traffic of a particular identity, Google is conceptually positioned to serve a unique software package to a targeted individual via the Play Store. Developer discussions indicated that Google systems are not organized to support this capability; further, NCC Group did not observe any indications that support this capability.

While client-side code can be more easily analyzed by users, it is infeasible for customers to perform such auditing on a continuous basis. However, the inability to serve unique software to a targeted individual makes client-side changes broadly visible. A potential future enhancement to alleviate this concern might incorporate third-party validation into client application and library distributions.

Manipulation of cryptographic protocols and parameters

Summary of capability: Google may be able to generate chosen cryptographic keys for users during the authorization flow which allows later stages to re-associate a VPN tunnel with a Google user identity. This risk is relevant to any VPN provider and may be more challenging to execute than the intentional code edits described above.

Example: Google's authentication server could choose to sign a specific user's blinded session token with a freshly rotated key such that (immediately) subsequent usage of the corresponding unblinded token may leak user identity.

Analysis: This concern is not related to the strength of the cryptographic protections offered by the product. During the course of the audit, NCC Group determined that Google used cryptography in alignment with industry best practices as discussed in the technical analysis of this report. This design concern does not represent a diminishing of cryptographic standards, but rather an opportunity for Google to instrument parameters generated by the server (such as keys) in a way that could be used to identify the user at later stages of the VPN connection setup. This general risk is also inherent in any use of third-party software.

A potential future enhancement to alleviate this concern might incorporate continuous third-party auditing into server-side key generation code.

[3] https://wiki.mozilla.org/Security/Binary_Transparency

Instrumentation of client device traffic

Summary of capability: After a VPN tunnel is established, Google could reassociate a tunnel with a Google identity by generating specific, identifiable network traffic via other services it operates.

Example:

1. An identity-aware Google application or website requests a resource via a tailored, unique domain name.
2. The target device requests the IP address of the associated server over DNS.
3. This DNS request is sent over the IPsec tunnel, and decrypted on the Copper server.
4. The Copper server analyzes the unique domain name request and reports it to Google identity backends.

Analysis: This activity would be difficult to attribute to Google without access to backend servers. A potential future enhancement to alleviate this concern might incorporate continuous third-party auditing into data-path infrastructure to ensure that this type of logging is not occurring.

Server-side analysis of network traffic

Summary of capability: Google could correlate networking metadata or cleartext traffic to develop strong associative metrics between an identity and tunneled traffic. This risk is also relevant to all VPN providers.

Example: The source IP address of the initial authentication request containing the identity-attributable OAuth token will be the same as the one on tunnel traffic inbound to the Copper server. As the initial authentication request and tunnel establishment happen within a very short time frame, this represents a fairly strong associative metric should backend code in the data-path choose to report this information. IPv6 traffic would add further confidence to the association, as IPv6 addresses are more likely to be unique and not shared by multiple devices via NAT.

Analysis: Similar to the instrumentation of client traffic capability, this activity would be difficult to attribute to Google without access to backend servers. Note that by implementing cryptographic blind-signing as described above, Google has introduced significant barriers to this activity. In addition, the engagement placed a strong focus on the potential for logging at both ends of the VPN connection. NCC Group inspected the logging performed by the Google One application to confirm its consistency with whitepaper claims, and this is described in the next section. A potential future enhancement to alleviate this concern might incorporate continuous third-party auditing into data-path infrastructure to ensure that this type of logging is not occurring.

System Architecture Summary

The technical architecture and design of the Google One VPN product enables Google to create a VPN with a very robust implementation with respect to technical security and privacy goals, that benefits its users with immediate security enhancements for their network traffic. It also provides a framework within which the product can provide authentication and authorization for users without sending identifying information to the VPN nodes. Thus, it effectively facilitates the separation of Google user identities from their tunneled traffic. While opportunities to circumvent this separation exist, they were not observed to be part of the product implementation.

Introduction

NCC Group performed a technical code review on a variety of components within the Google One VPN product. The following subsections cover the four primary testing targets: the Android application, the client-side VPN library code, the corresponding server-side authentication-related code, and an external network penetration test of the VPN exit nodes. Each subsection begins with a brief overview and then describes NCC Group's testing methodology for the specific component followed by a high-level appraisal of the observed security posture at the time of the engagement.

Google One Android Application

Overview

The Google One Android application combines a variety of features such as allowing users to automatically back up their phone, management of their Google cloud storage, and access to Google experts for product and service support. A recent feature addition to the application is a VPN subsystem that manages the configuration, authentication and establishment of VPN sessions. The VPN service feature is activated when a user pays for a subscription to the application.

Relevant portions of the application source code were delivered via a secure file exchange for static testing. In addition, the complete Google One application with the package name `com.google.android.apps.subscriptions.red` was downloaded from the Google Play Store for dynamic testing. Dynamic testing was performed on a rooted Pixel 3 device running Android API 28.

A test Google account was created for testing the application and a subscription was purchased to activate the VPN functionality for that account. Note that the source code for static testing was a slightly newer version of that obtained from the Play Store. The VPN service can be turned on by the user by going to the "Enable VPN" page in the Google One Android Application, as shown below:

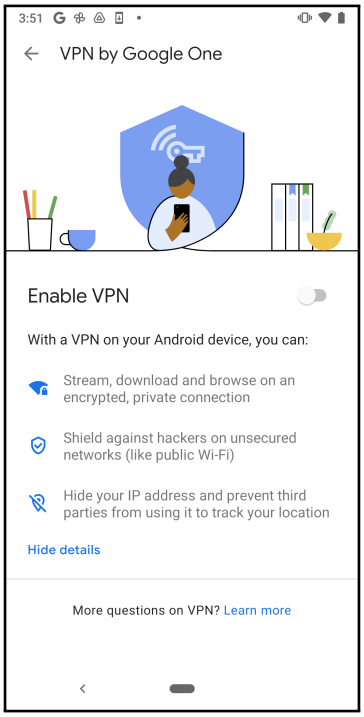

Figure 4: Screenshot of a user enabling VPN

Testing Methodology

The application was tested for a wide range of security controls, including:

- Networking vulnerabilities, both listening and active
- Insecure client-server communications
- Improper use of web request APIs
- Insufficient transport encryption
- Certificate or public key pinning
- Weak permissions, unsafe storage of sensitive data
- Transport security weaknesses
- Injection attacks (including SQL, LDAP, XML)
- Denial of service vulnerabilities
- Weaknesses in authentication, including:
  - Authentication bypass
  - Vulnerability to brute force attacks
  - Improper authorization checks
  - Data enumeration vulnerabilities
  - Improper token invalidation
  - Predictable session tokens
- Exposure of sensitive customer data

In addition, the Google One application was then tested for various Android-specific vulnerabilities. The application was tested using the `Mobile Security Framework`[4] and `Drozer`[5] security framework tools. The application's `apk` was decompiled using the `jadx` decompiler.[6] Decompiled code from the Google Play Store version of the application, as well as source code provided directly to NCC Group by Google, were reviewed for common Android application flaws, misconfigurations, logic issues, logging mechanisms, etc. The `Frida`[7] dynamic testing toolkit was also used to perform dynamic analysis and bypass certificate pinning.

The VPN authentication and setup flow contains multiple requests to `<REDACTED-1>` and `<REDACTED-2>` hosts. When the VPN is turned ON, the application gets a public key from the `<REDACTED-1>` server and sends back an `Oauth_token` and `blinded_token` to the server along with the hash of the public key. The VPN configuration is received from the `<REDACTED-2>` host which takes in the `unblinded_token` along with some additional VPN config.

The following DNS servers were configured on the device by VPNManager when the Google One VPN was started: `8.8.8.8, 8.8.8.4, 2001:4860:4860::8888, 2001:4860:4860::8844`

```
01-27 11:31:58.210 14658 15344 W VpnManager: Adding DNS: 8.8.8.8
01-27 11:31:58.210 14658 15344 W VpnManager: Adding DNS: 8.8.8.4
01-27 11:31:58.210 14658 15344 W VpnManager: Adding DNS: 2001:4860:4860::8888
01-27 11:31:58.210 14658 15344 W VpnManager: Adding DNS: 2001:4860:4860::8844
01-27 11:31:58.269 14658 15344 W VpnManager: Establishing Tun FD
01-27 11:31:58.275  1172  1567 D VpnJni  : Address added on tun0: 100.72.146.191/32
01-27 11:31:58.275  1172  1567 D VpnJni  : Address added on tun0: 2604:ca00:1ea:2f9d::260:5c29/64
```

Figure 5: DNS and IP addresses being assigned and logged on the device

Additionally, it was observed that the application was not able to determine that it is running on a rooted device. The testing device was rooted using Magisk installer v8.0.7.

Findings

The assessment identified two vulnerabilities that are arguably outside of the VPN subsystem but within the overall user experience.

1. One low-severity finding in the Google One Android application currently published on the Google Play Store. The application was observed to be using public directories that are world readable/writable. Note that this finding was not found in the code shared directly by Google, as it corresponds to an updated version of the application that is not yet published, indicating that this issue has already been fixed upstream.

[4] https://mobsf.github.io/docs/#/
[5] https://labs.f-secure.com/tools/drozer/
[6] https://github.com/skylot/jadx
[7] https://frida.re/

2. One low-severity finding in the Google One Android application currently published on the Google Play Store. The application leaks plain-text passwords via the recent apps task switcher when the application is backgrounded while logging in to a Google account. The low-severity rating is primarily due to the unlikely user-driven scenario, timing constraints, need for physical access and other preconditions. This vulnerability was observed to exist in the updated code that was shared for testing.

The application does perform logging, but it was observed that no sensitive information is being logged. All logging functionality used generic messages and no sensitive information was being leaked. The source code was also reviewed for potential logging escapes.

No suspicious activity or information leakage was observed in the DNS requests on the `tun0` interface configured by the VPN. Packets were captured on all interfaces with and without VPN using `tcpdump`[8] tools and further analyzed.

Client-side VPN Library

Overview

A central component of the Google One Android application's VPN functionality resides in the open-source VPN library[9] code publicly available on GitHub. This code is effectively integrated within the Google One application described above. The repository contains both higher-level Android-specific code written in Java, and also lower-level authentication, cryptographic and session-establishment code written in C++.

Testing Methodology

Manual and assisted source code review was performed on all code within the repository. This effort spanned a wide range of analysis including input validation, output encoding, authentication/authorization, memory management, state management, protocol analysis, sources/uses of randomness, corner cases, error handling and general secure coding practices along with Android-specific security techniques. The Java code had some overlap with the overall Android application testing effort.

Additionally, the cryptographic aspects of the C++ code were carefully analyzed to ensure correct operation without any unintentional weaknesses or data leakage. This involved the following methodologies:

- Selection of appropriate cryptographic primitives, algorithms and security parameters.
- Protocol resistance to message replay, drops, modification and side-channel timing attacks.
- Full key life-cycle management processes while avoiding extraneous key reuse.
- Review of session ID message construction via a full domain hash[10] scheme.
- Sources of client-side randomness and "flooding" rather than rejection sampling.
- Mathematical corner cases such as $r = 0, 1$, co-prime tests and signature malleability.
- The potential to use instances of authentication key rotation timing to leak user subset information.
- Ability for malicious client sharing of an unblinded session ID and corresponding signature.
- Distinguishing between an expired token and an attempted forgery.
- The potential for malicious server-generated session IDs such as $0, 1$.
- Intermediate validation of authentication signature on the client before interacting with the data-path server.

Findings

A variety of largely cryptographic related issues were uncovered in the code and reported to Google; see the Table of Findings on page 19 for additional details. Subsequently, Google delivered a fresh commit for retesting with the vast majority of issues resolved. As it currently stands, the C++ code implements the RSA blinded signature algorithm that supports the structural separation of user-vs-usage information between the authentication and data-path servers.

[8] https://www.tcpdump.org/
[9] https://github.com/google/vpn-libraries/tree/9deaf79ab67d23f7cd6472be35ed3f8c2a9c8a2e
[10] https://www.iacr.org/archive/crypto2000/18800229/18800229.pdf

Server-side Authentication

Overview

Selected server-side code components from Brass and Zinc that interact with the VPN library described above were reviewed along with additional components involved in key management and blind signing. This proprietary code is written in Golang and central to the cryptographic blinding operations that separate user information on the authentication and authorization servers from usage information on the VPN data-path servers. The code utilizes a range of cryptographic primitives from Google's BoringSSL library.[11]

Testing Methodology

Manual source code review was performed on these components in conjunction with the VPN library above. As noted, the review included a broad range of secure coding practices along with the cryptographic-specific tests; these are outlined in the preceding section above.

Findings

Similar to the VPN library, a variety of issues were uncovered and reported to Google; see the Table of Findings on page 19 for additional details. Subsequently, Google delivered a fresh commit for retesting with the vast majority of issues resolved. As it currently stands, the server-side Golang code complements the client-side code implemented in the VPN library.

External Network Assessment On Exit Node

Overview

An exit node sits in between a client's device and the Internet. It is responsible for encrypting/decrypting data packets and providing NAT functionality so that traffic between a client and the Internet can be transmitted securely and correctly. It also obfuscates the client's IP address. Compromise of an exit node can break this security design, leaving client data at risk. The purpose of this section of the assessment was to identify any security gaps existing on the VPN exit nodes from an external attacker's perspective.

Testing Methodology

Upon kickoff of the engagement, NCC Group conducted a reconnaissance scan of the target host IPs provided by Google. The purpose of this scan is to quickly and efficiently identify all ports, services, and operating systems within the scope of the assessment. The effort included Open Source information gathering, DNS reconnaissance, port, service and user enumeration. In addition, the consultant performed an automated vulnerability scanning and discovery against the target hosts and services to ensure thoroughness and redundancy. Cross checking the results helped to verify the completeness of the initial scan and further identify areas to focus in on during manual testing efforts.

Subsequently, a full TCP probe and a full UDP probe were performed against the provided exit node. Different firewall/IDS evasion techniques were attempted in order to probe more services running on the exit node. For example, scanning with fragmentation, MAC address spoofing, source port number specification, or data length manipulation were attempted.

Finally, manual penetration testing was performed against the provided host by interrogating identified ports and focusing on the services, operating systems, and applications identified in the initial perimeter and vulnerability scans. The primary focus during this phase was on the following security areas:

- Correlation of service fingerprint, version, and behavior to evaluate susceptibility to publicly known vulnerabilities
- Manual inspection of potential unreported vulnerabilities
- Access control circumvention
- Client/Server impersonation and spoofing attacks
- Platform configuration issues
- Exploitation of identified vulnerabilities in order to gain access to sensitive information or systems

[11] https://boringssl.googlesource.com/boringssl/

- Post-exploitation efforts where possible

Findings

The final phase of the assessment consisted of the consultant analyzing and researching the vulnerabilities and potential resolutions. The findings were then documented with recommended remediation steps and report provided to Google. There were no major risks uncovered during this portion of the assessment. Only ports 1849/TCP and 2153/TCP were found to be exposed externally for the target exit node, both of which are used to build and maintain a secure connection between the client and the VPN server.

Technical Component Summary

While the source code demonstrated good structure, attention to detail and a robust test suite, the technical assessment uncovered fourteen findings in total. The development team subsequently delivered updated code and all fourteen findings were retested. Ten findings were found to be 'Fully Fixed'. Of the remaining four:

- One medium-severity finding, requiring a protocol change that would increase communication steps, was considered to be an acceptable risk by Google.
- One low-severity finding was considered to be an acceptable risk by Google.
- One informational observation was confirmed to be 'Partially Fixed' while a second was considered to be an acceptable risk by Google. *Note that informational observations do not indicate immediate security concerns*.

The two Android-related findings may be arguably outside of the VPN subsystem, but remain part of the overall user experience.

Conclusions

Security

NCC Group's assessment included an in-depth analysis of the product's use of technology to provide a VPN system which is secure for its users and Google. The consulting team provided a list of findings during the first phase of the engagement, the bulk of which Google addressed promptly.

By the conclusion of the assessment and remediation period, NCC Group found the product to have a very robust security posture. The consulting team determined that the use of modern operating system libraries and strong, openly standardized cryptographic protocols enabled Google to provide a VPN which benefits its users with immediate security enhancements for their network traffic.

For more detailed analysis and conclusions with regards to technical security, see the System Architecture Review on page 5 and Technical Component Analysis on page 11 of the document.

Privacy

A substantial amount of engagement time was allocated to the analysis of Google's ability to fulfill the privacy claims outlined in the product's whitepaper. The consulting team considered the system's capacity to deliver on these claims from a technical, design, and procedural perspective.

Ultimately, NCC Group determined that while the supplemental cryptographic privacy protections did not categorically eliminate the opportunity for Google to violate its privacy claims, they did provide a framework within which the application can provide authentication and authorization for users without sending identifying information to the VPN nodes. NCC Group did not observe claims circumvention to be part of the project's strategy or implementation during the assessment. Furthermore, Google demonstrated business processes which would impede malicious employees or individual business units from making unauthorized changes. During the audit, NCC Group organized meetings to discuss deployment procedures implemented by Google which further reduces the risk of insecure or non-compliant code being introduced to the product.

For more detailed analysis and conclusions with regards to both security and privacy, see the System Architecture Review on page 5 and Technical Component Analysis on page 11 of the document.

Finding Field Definitions

The following sections describe the risk rating and category assigned to issues NCC Group identified.

Risk Scale

NCC Group 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 NCC Group'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.

Overall Risk

Overall risk reflects NCC Group's estimation of the risk that a finding poses to the target system or systems. It takes into account the impact of the finding, the difficulty of exploitation, and any other relevant factors.

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.

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.

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.

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

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

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.

Table of Findings

For each finding, NCC Group 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. For an explanation of NCC Group's risk rating and finding categorization, see Finding Field Definitions on page 17.

Title	Status	ID	Risk
`Unblind()` Does Not Validate Server Signature	Fixed	009	High
Usage of Non-Cryptographic Randomness	Fixed	001	Medium
RSA Public Key Size, Exponent Not Validated	Fixed	008	Medium
Unblinded Token Structure Not Validated	Fixed	010	Medium
Potential for User Tracking via Targeted Shared Secret Generation	Risk Accepted	013	Medium
Backgrounding Screenshots Enabled	Risk Accepted	002	Low
Overly Tolerant JSON Validation During Decode	Fixed	004	Low
Insecure and/or Extraneous Defaults	Fixed	005	Low
Insufficient Validation of Token Expiration	Fixed	006	Low
Google One Android Application Uses Public Directory	Fixed	007	Low
Missing Private-to-Public Key Conversion of Retired Key Array	Fixed	011	Low
JSON Web Token Remnants	Partially Fixed	003	Informational
Non Constant-Time RSA Modular Exponentiation	Risk Accepted	012	Informational
Trivially Loose Bound on Blinding Factor `r`	Fixed	014	Informational