VPN by Google One Security Assessment

Google Inc

Version 1.1 – December 14, 2022

While precautions have been taken in the preparation of this document, NCC Group the publisher, and the author(s) assume no responsibility for errors, omissions, or for damages resulting from the use of the information contained herein. Use of NCC Group's services does not guarantee the security of a system, or that computer intrusions will not occur.

Prepared By
Daniel Romero
Laura Garcia
Mario Rivas
Rafael Alfaro March
Shawn Fitzgerald

Prepared For
Google Inc

1 Executive Summary

Synopsis

During the summer of 2022, Google engaged NCC Group 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[1], 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

NCC Group'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.
- Ten findings rated low-severity.
- Nine findings rated as informational observations.

The most notable finding was related to the requirement of the Windows application to be executed with administrator privileges. While NCC Group 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.

Strategic 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] https://www.gstatic.com/subscriptions/marketing_page/vpn/white_paper_4f995ab5d7c7edc3d3f14f2e0593f790.pdf

2 Dashboard

Finding Breakdown

	Original Assessment	Remaining
Critical issues	0	0
High issues	0	0
Medium issues	3	2
Low issues	10	9
Informational issues	9	8

Category Breakdown

Access Controls	1	⬜
Configuration	6	🟨🟨🟨⬜⬜⬜
Cryptography	4	🟨🟨🟨🟨
Data Exposure	6	🟨🟨⬜⬜⬜⬜
Denial of Service	2	🟧🟧

Component Breakdown

Android Application	3	🟨⬜⬜
Windows Application	7	🟧🟨🟨🟨⬜⬜⬜
iOS Application	2	🟨⬜
macOS Application	6	🟧🟨🟨🟨⬜⬜
vpn-libraries	1	🟨

🟪 Critical 🟥 High 🟧 Medium 🟨 Low ⬜ Informational

3 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.

Android Application

Title	Status	ID	Risk
Lack of Certificate Pinning	Reported	WWL	Low
Missing Permissions on Android Receivers	Reported	XA6	Info
User Email Address Stored Without Application-Level Encryption	Reported	3JC	Info

Windows Application

Title	Status	ID	Risk
Weaknesses in Authentication Process	Reported	LXX	Medium
Lack of Privilege Separation	Fixed	9W7	Medium
Lack of Anti-Exploit Protections	Reported	MDL	Low
Sensitive Data Sent in the URL Using POST Method	Reported	MWM	Low
Lack of Certificate Pinning	Reported	J2Q	Low
Application Vulnerable to DLL Injection	Reported	DDJ	Info
Sensitive Information Written to Debug Logs	Fixed	BQL	Info
Application Binaries Not Obfuscated	Reported	GUM	Info
Binaries Contained Debug Information	Reported	UEX	Info

iOS Application

Title	Status	ID	Risk
Application Disables App Transport Security	Reported	VPB	Low
Mobile Application Data Storage Leaks GAIA ID in Log Files	Fixed	XUH	Low
Mobile Application Backgrounding Leaks Sensitive Info in Screenshots	Reported	RPK	Info

macOS Application

Title	Status	ID	Risk
Weaknesses in Authentication Process	Reported	G96	Medium
Sensitive Data Sent in the URL Using POST Method	Reported	6NH	Low
Application Vulnerable to DYLIB Hijacking	Reported	GMU	Low
Lack of Certificate Pinning	Reported	KE3	Low
Binaries Contained Debug Information	Not Fixed	B9L	Info
Application Binaries Not Obfuscated	Partially Fixed	DRA	Info

vpn-libraries

Title	Status	ID	Risk
Use of Deprecated and Internal Functions	Reported	AJK	Low

4 Architecture Review Analysis

During the first two weeks of the engagement, NCC Group 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. This phase of the engagement was performed independently of any specific implementation of the design, focusing specifically on the technical concepts described in the client provided documentation listed in Client Provided Documentation, and in person interviews. Additionally, NCC Group created architecture diagrams, which details the components, trust boundaries and communication paths.

Architecture/Platform 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.

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

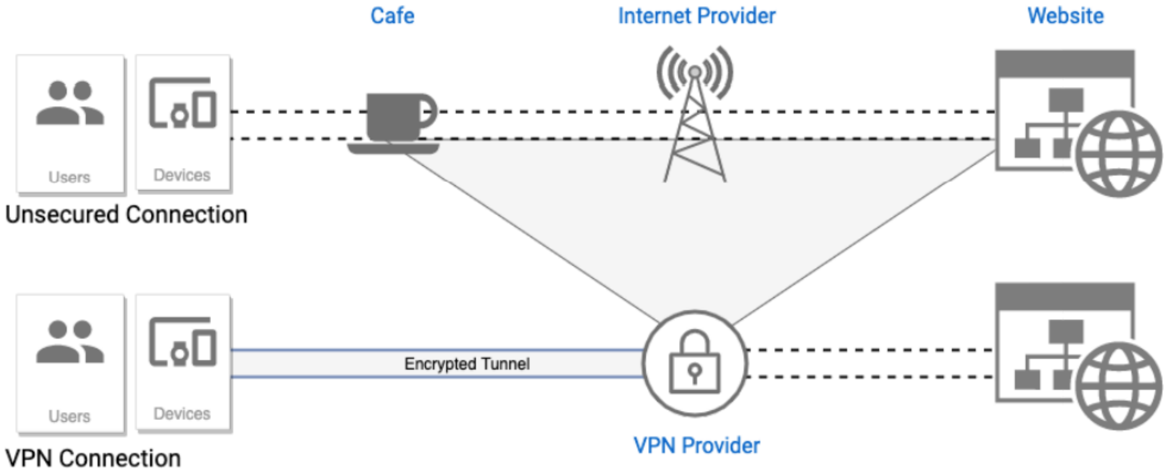


Figure 1: Typical network connection and using the VPN from the Google whitepaper

Scope and Architecture Diagrams

NCC Group 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, NCC Group 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.

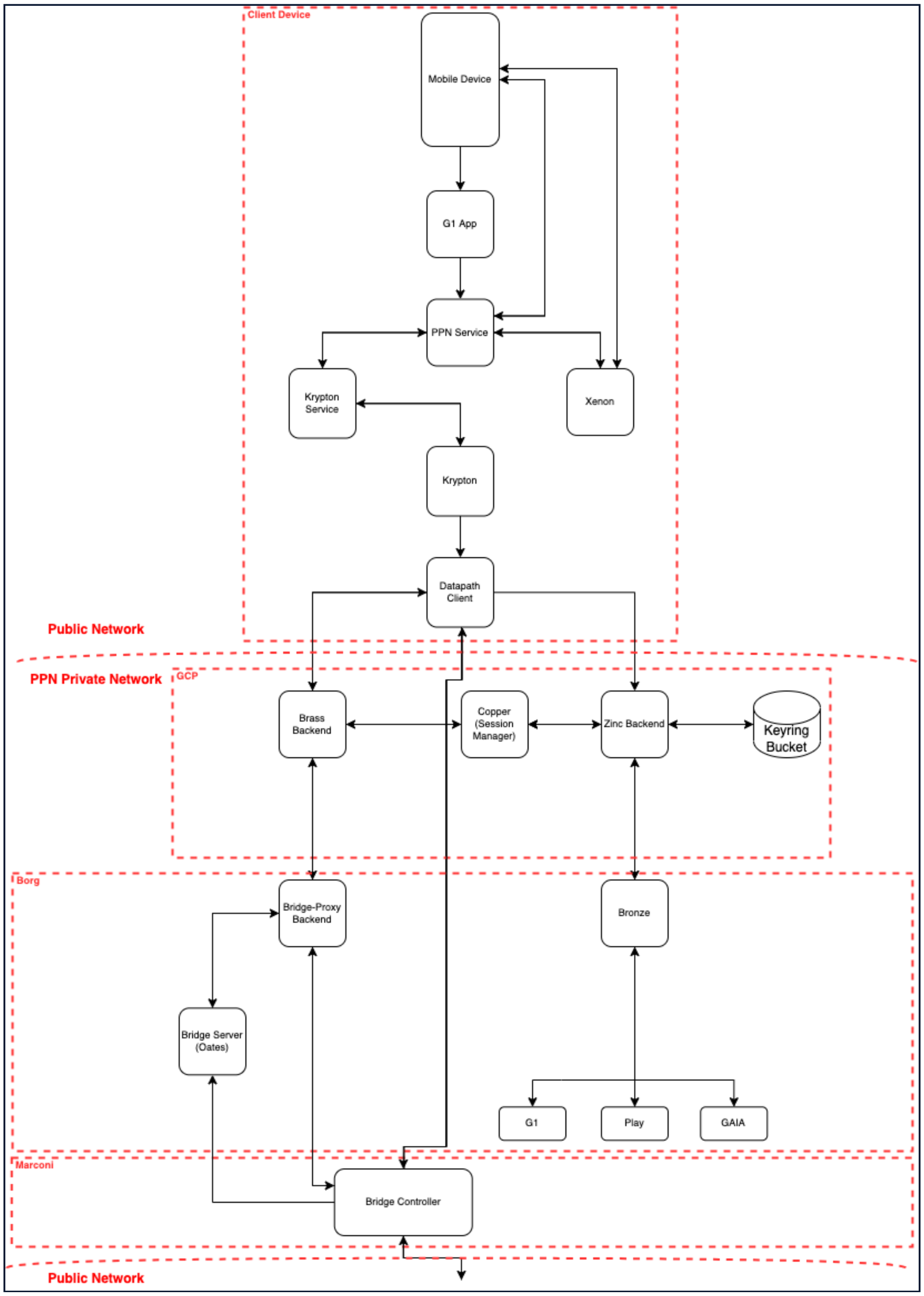


Figure 2: VPN Architecture Overview Diagram in Production

Each component is described below.

Client 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.

Current Google PPN architecture

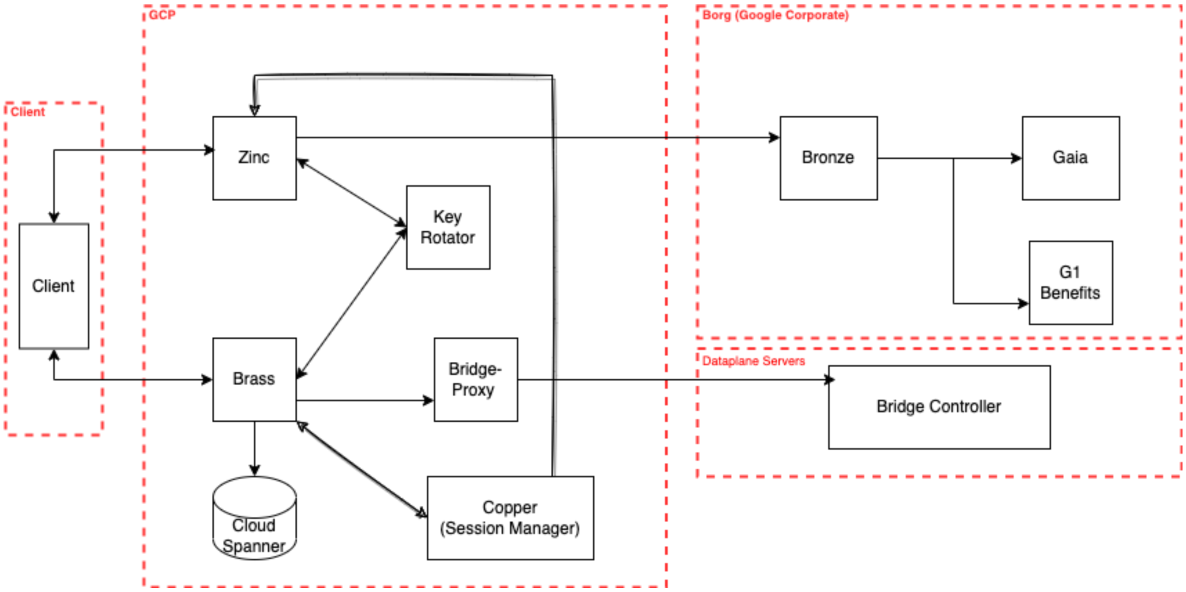


Figure 3: VPN Architecture Overview at current date

Components

- **Brass:** 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 that the dataplane generates to return to the client device.
- **Copper(Session Manager):** The exit node responsible for sending/receiving the packets from the device and the internet. Selects the node of the Session Manager service based on user preference, service policy and load balancing.
- **Zinc:** Authentication server responsible for authentication and authorization of the PPN service where it proxies GAIA auth to Bronze. It is also responsible for converting an OAuth token and blinded session token into a signed blinded session token.
- **Bridge-Proxy Backend:** Talks to Bridge server to get valid Bridge token and sends it to the bridge controller.
- **Bridge-Server:** Provides a valid token to the bridge-proxy.
- **Bridge Controller:** Handles requests to program the packet Processor.
- **Bronze:** Closed source authentication server that is responsible for GAIA and G1 service authorizations which the Zinc talks to and validates from.
- **G1 Benefits:** Tells if the user has a PPN subscription.
- **GAIA:** Responsible for service authorization.
- **Cloud Spanner DB:** Stores session tokens to protect Brass from token replay attacks.
- **Key Rotator:** Generates and rotates blind-signing keys periodically using a cron job.

Q3-2022 Google PPN architecture

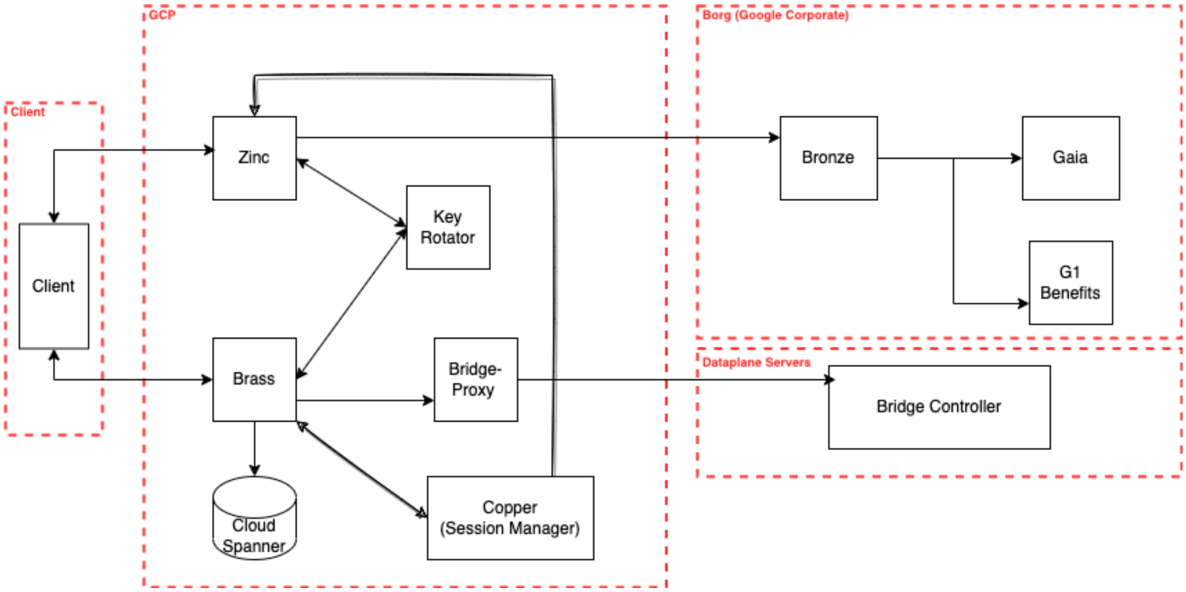


Figure 4: VPN Architecture Overview after migration planned at Q3

Components

The migration introduces Phosphor which replaces Zinc and Bronze; simplifying the client connection calling a public JSON API into Borg. The other components remain the same as described above.

- **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).
- **Attestation Service:** A service running within Phosphor instance that handles device attestation from Android and iOS devices. This service will prevent authentication from clients that have been modified.

General Conclusions

At the end of the review, NCC Group 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.

Privacy Claim Violation Opportunities

The subsections below enumerate the techniques that could be employed to associate the VPN traffic with subscriber's identity.

Manipulation of the client application

In the future, Google could update the client application to facilitate attribution of traffic to users. This type of compromise could occur at multiple levels within the application, such as modification of authentication and authorization flow or reporting the IPSec connection parameters.

Google implemented a new component called Attestation Service which will be added to the architecture in the upcoming migration planned to begin in Q3 2022. Attestation Service is designed to reject requests coming from manipulated client applications. Although this component is a step in the right direction, in order to prevent undesirable modifications by third parties, it can be assumed that since internal Google employees are responsible for these updates to the application; the protection given by Attestation Service can be bypassed as well. As Google operates the distribution platform (Play Store) and Attestation Service/Phosphor backend servers, an update of the client application might be performed to facilitate attribution of traffic to users.

Manipulation of cryptographic parameters

During the initial authorization phase, where the client application sends the blinded VPN authentication token along with the identity-attributable OAuth token, Google may be able to mark the blinded token in a way it can later identify by using a unique public key for the signing operation.

Instrumentation of client device traffic

After an IPSec connection is established, Google could reassociate a tunnel with a Google identity via generating specific, identifiable network traffic. An example of such a situation is outlined below, similar techniques are likely possible:

1. A 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 of network metadata and metrics

Google could correlate networking information such as device source IP address and connection times to establish an association between an identity and tunneled VPN traffic. 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. 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.

5 Finding Details – Android Application

Low

Lack of Certificate Pinning

Overall Risk	Low	**Finding ID**	NCC-GOLE021-WWL
Impact	Medium	**Component**	Android Application
Exploitability	Low	**Category**	Cryptography
		Status	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 years[2,3] , 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 Config[4] to define a list of trusted certificate hashes without manual checking being necessary. Information about this mechanism is available on Android's developer documentation[5].

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 code[6].

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.

[2] DigiNotar - Issuance of fraudulent certificates: https://en.wikipedia.org/wiki/DigiNotar#Issuance_of_fraudulent_certificates
[3] Comodo - Certificate hacking: https://en.wikipedia.org/wiki/Comodo_Group#Certificate_hacking
[4] Android Developers - Network Security Configuration: https://developer.android.com/training/articles/security-config
[5] Android Developers - CertificatePinning: https://developer.android.com/training/articles/security-config#CertificatePinning
[6] OkHttp library: https://square.github.io/okhttp/3.x/okhttp/okhttp3/CertificatePinner.html

In addition, it is considered a good practice to pin more than one certificate, especially when pinning individual host certificates, to reduce the risk of issues associated to a specific certificate, such as an expired certificate.

Reproduction Steps

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

Location

Google One for Android, version 1.157.459421718

Info

Missing Permissions on Android Receivers

Overall Risk	Informational	**Finding ID**	NCC-GOLE021-XA6
Impact	Low	**Component**	Android Application
Exploitability	None	**Category**	Access Controls
		Status	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:

```xml
<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.

```java
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.

Location

Android application Manifest

Info

User Email Address Stored Without Application-Level Encryption

Overall Risk	Informational	**Finding ID**	NCC-GOLE021-3JC
Impact	Low	**Component**	Android Application
Exploitability	Low	**Category**	Data Exposure
		Status	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 library[7] where possible. The Jetpack Security library is provided by the Android team to enable secure, standardized encryption mechanisms based on the Android Keystore system[8], and to ease the transition for developers.

Reproduction Steps

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
```

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

Location

- com.google.android.libraries.privacy.ppn.internal.PpnSettings

[7] Android Developers - Work with data more securely: https://developer.android.com/topic/security/data.md

[8] Android Developers - Android Keystore system: https://developer.android.com/training/articles/keystore

6 Finding Details – Windows Application

Medium Weaknesses in Authentication Process

Overall Risk	Medium	**Finding ID**	NCC-GOLE021-LXX
Impact	Medium	**Component**	Windows Application
Exploitability	Low	**Category**	Denial of Service
		Status	Reported

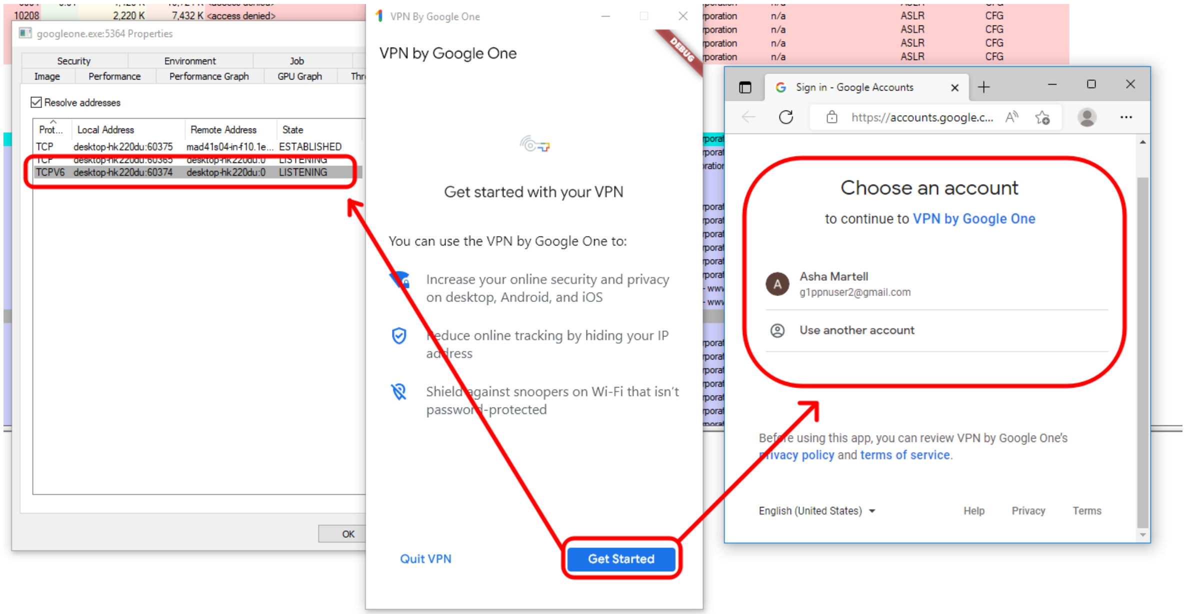
Impact

A local attacker could cause a Denial of Service condition by preventing users from successfully authenticating to the VPN service and may be able to compromise the OAuth token.

Description

NCC Group identified that by sending a crafted HTTP request to the open local port used for authentication, the application closes the port. Once this port is closed, an attacker can open this TCP port and masquerade as this process. This can allow a local attacker to obtain the OAuth token of the user attempting to authenticate to the VPN. Additionally, once the port is closed, a VPN user can no longer authenticate to the VPN service.

If the VPN's user authenticates by clicking the "Get Started" button, a new port is opened as the following image shows:



googleone.exe:5364 Properties

Prot...	Local Address	Remote Address	State
TCP	desktop-hk220du:60375	mad41s04-in-f10.1e...	ESTABLISHED
TCP	desktop-hk220du:60365	desktop-hk220du:0	LISTENING
TCPV6	desktop-hk220du:60374	desktop-hk220du:0	LISTENING

VPN by Google One

Get started with your VPN

You can use the VPN by Google One to:

- Increase your online security and privacy on desktop, Android, and iOS
- Reduce online tracking by hiding your IP address
- Shield against snoopers on Wi-Fi that isn't password-protected

Sign in - Google Accounts

https://accounts.google.c...

Choose an account

to continue to VPN by Google One

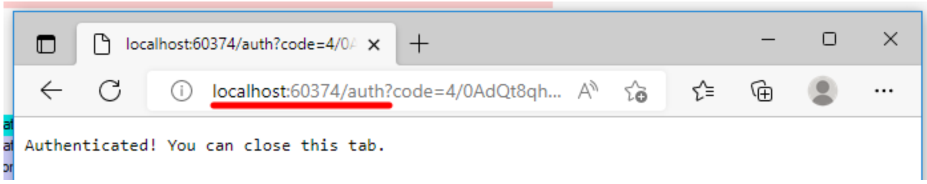
Asha Martell
g1ppnuser2@gmail.com

Use another account

Before using this app, you can review VPN by Google One's privacy policy and terms of service.

Figure 5: Authentication open port

Once the user is authenticated through the web browser, this connects to the localhost port to send the auth token:



localhost:60374/auth?code=4/0AdQt8qh...

```
Authenticated! You can close this tab.
```

Figure 6: Authentication successfully

By looking into the application's implementation, it was identified that if the port is reached with an authentication GET request without parameters (e.g. `http://localhost:60374/auth`), the application closes the listening port. Therefore, a malicious application with no administrator privileges could monitor the open ports, send a request to close them (causing a denial of service by not allowing the authentication to succeed) and open the same port again to capture the OAuth token.

Recommendation

Ensure that the authentication code can properly handle unexpected user data. Alternatively, implement the authentication directly in the Google One VPN application, which would avoid the use of external applications (such as the web browser) and the need of opening a local port to receive the OAuth token.

Location

- g1_windows/app/auth/lib/auth.dart

Medium

Lack of Privilege Separation

Overall Risk	Medium	**Finding ID**	NCC-GOLE021-9W7
Impact	Medium	**Component**	Windows Application
Exploitability	Medium	**Category**	Access Controls
		Status	Fixed

Impact

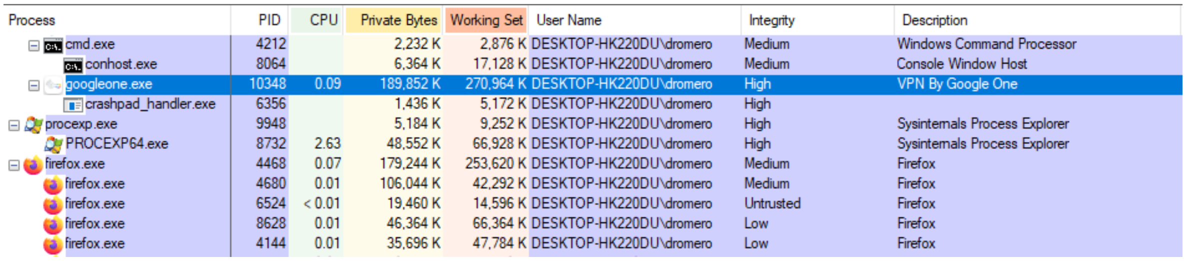
An attacker that manages to compromise the Google One VPN application's runtime process would gain administrative privileges within its host environment.

Description

The Google One VPN application process is required to be run with administration privileges. In systems based on Microsoft Windows, the user administrator has full access to the whole system without any restriction.

It is acknowledged that elevated privileges could be required by the Google One VPN software to manage or filter network packets, raw sockets or operating system functionalities. However, other elements such as configurations, logs, authentication communications or even parsers that introduce a large attack surface, can and should be run with limited privileges and within a sandbox (isolated environment). Under this model, even if an attacker were to find a vulnerability that could be exploited (for example in the authentication process), they would still need to identify a way to escape from the sandbox and escalate their privileges in order to obtain full control over the system.

As shown below, the `googleone.exe` process was running with high integrity level:



Process	PID	CPU	Private Bytes	Working Set	User Name	Integrity	Description
cmd.exe	4212		2,232 K	2,876 K	DESKTOP-HK220DU\dromero	Medium	Windows Command Processor
conhost.exe	8064		6,364 K	17,128 K	DESKTOP-HK220DU\dromero	Medium	Console Window Host
googleone.exe	10348	0.09	189,852 K	270,964 K	DESKTOP-HK220DU\dromero	High	VPN By Google One
crashpad_handler.exe	6356		1,436 K	5,172 K	DESKTOP-HK220DU\dromero	High	
procexp.exe	9948		5,184 K	9,252 K	DESKTOP-HK220DU\dromero	High	Sysinternals Process Explorer
PROCEXP64.exe	8732	2.63	48,552 K	66,928 K	DESKTOP-HK220DU\dromero	High	Sysinternals Process Explorer
firefox.exe	4468	0.07	179,244 K	253,620 K	DESKTOP-HK220DU\dromero	Medium	Firefox
firefox.exe	4680	0.01	106,044 K	42,292 K	DESKTOP-HK220DU\dromero	Medium	Firefox
firefox.exe	6524	< 0.01	19,460 K	14,596 K	DESKTOP-HK220DU\dromero	Untrusted	Firefox
firefox.exe	8628	0.01	46,364 K	66,364 K	DESKTOP-HK220DU\dromero	Low	Firefox
firefox.exe	4144	0.01	35,696 K	47,784 K	DESKTOP-HK220DU\dromero	Low	Firefox

Figure 7: Google One VPN application executed with high integrity level

Recommendation

A new design based on separation of privileges could be implemented in order to reduce the risk outlined above, although due to the nature of the Google One VPN subsystem and its current state, designing and implementing this may require a significant amount of time.

In principle, the new design should ensure that actions that require high privileges are handled separately from those that have lesser requirements (such as parsers, authentication requests or protocol communications).

Low

Lack of Anti-Exploit Protections

Overall Risk	Low	**Finding ID**	NCC-GOLE021-MDL
Impact	Low	**Component**	Windows Application
Exploitability	Low	**Category**	Configuration
		Status	Reported

Impact

The application may be exposed to memory corruption attacks.

Description

Some of the DLL libraries and binaries that are part of the Google One VPN product did not make use of all anti-exploit protections that are available in modem Windows environments. Specifically a platform security feature called `ControlFlowGuard`[9], which places restrictions on where and how an application can execute code. The omission of this protection can make the application and related binaries more vulnerable to exploitation.

The following is the list of configured binary protections for each of the binaries:

Main Path: “C:\Program Files\Google\VPN By Google One\1.0.2000.8”

```
FileName         : C:\Program Files\Google\VPN By Google One\1.0.2000.8\crashpad_handler.exe
ASLR             : True
DEP              : True
ControlFlowGuard : False
HighentropyVA    : True

FileName         : C:\Program Files\Google\VPN By Google One\1.0.2000.8\flutter_windows.dll
ASLR             : True
DEP              : True
ControlFlowGuard : False
HighentropyVA    : True

FileName         : C:\Program Files\Google\VPN By Google One\1.0.2000.8\googleone.exe
ASLR             : True
DEP              : True
ControlFlowGuard : False
HighentropyVA    : True

FileName         : C:\Program Files\Google\VPN By Google One\1.0.2000.8\googtun.dll
ASLR             : True
DEP              : True
ControlFlowGuard : True
HighentropyVA    : True

FileName         : C:\Program Files\Google\VPN By Google
↳ One\1.0.2000.9\VpnByGoogleOneService.exe
ASLR             : True
DEP              : True
ControlFlowGuard : False
HighentropyVA    : True
```

[9] Control Flow Guard: https://learn.microsoft.com/en-us/windows/win32/secbp/control-flow-guard

Recommendation

It is suggested that the Control Flow Guard mechanism is enabled when compiling the main application binary. CFG is a security feature that was created to combat memory corruption vulnerabilities, by placing restrictions on where an application can execute code. CFG extends previous exploit mitigation technologies such as /GS, DEP and ASLR.

Location

- crashpad_handler.exe
- flutter_windows.dll
- googleone.exe
- googtun.dll[10]
- VpnByGoogleOneService.exe

[10] It should be noted that the original assessment included the wintun.dll component. During retesting this had been updated by Google to googtun.dll. Other than retesting of the original finding, no other testing was performed on googtun.dll or the component that has included that dll.

Low

Sensitive Data Sent in the URL Using POST Method

Overall Risk	Low	**Finding ID**	NCC-GOLE021-MWM
Impact	Low	**Component**	Windows Application
Exploitability	Low	**Category**	Data Exposure
		Status	Reported

Impact

An attacker could intercept the communications between the thick application and the server, allowing them the ability to retrieve valid OAuth access tokens.

Description

The desktop client application sent requests to the authentication services located in `oauth2.googleapis.com` containing sensitive information in the URL. This exposed parameters such as `client_id`, `client_secret` and `refresh_token` to be stored in proxies.

```
POST /token?
↳ client_id=874847826917-0rfhjap59nhp8fpun56mm51sshkfjd2f.apps.googleusercontent.com&client_sec
↳ ret=2ndJjPpvzenW2pGSRy-KYddM&grant_type=refresh_token&refresh_token=1%2F%2F032SarWRZ702eCgYIA
↳ RAAGAMSNwF-L9Irklhy1lRgUARdbJdD2xbgHK3R0gVLrvD1kz98Jyrcb5Mf6bdnN2FE0fwHUd6KEs-fwPA HTTP/1.1
Connection: close
Content-Type: application/json; charset=utf-8
Accept: application/json
User-Agent: PPN HttpFetcher
Content-Length: 0
Host: oauth2.googleapis.com
```

The response also contained the current access token, making it possible to retrieve this information by reusing the same request while the refresh token is valid:

```
HTTP/1.1 200 OK
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Expires: Mon, 01 Jan 1990 00:00:00 GMT
Pragma: no-cache
Date: Wed, 31 Aug 2022 13:00:58 GMT
Content-Type: application/json; charset=utf-8
Vary: X-Origin
Vary: Referer
Server: scaffolding on HTTPServer2
X-XSS-Protection: 0
X-Frame-Options: SAMEORIGIN
X-Content-Type-Options: nosniff
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000,h3-Q050=":443"; ma=2592000,h3-Q046=":
↳ 443"; ma=2592000,h3-Q043=":443"; ma=2592000,quic=":443"; ma=2592000; v="46,43"
Accept-Ranges: none
Vary: Origin,Accept-Encoding
Connection: close
Content-Length: 487

{
  "access_token": "ya29.a0AVA9y1sTR9jfAFrFP8__t9lwxVxPwlXg0F99WYy-p-
  ↳ OZqcZ1P2HiJLQGaxGx1swwWkI59gW37Ekgy6JNrTfIGlJR-
  ↳ ZsJHj789EcxvuPFKrkVJgdJlsKA4D7z3iiUqvcdlUNfsB2oYqBLPQPZHMZdgl64CdzplQaCgYKATASAQASFQE65dr8K
  ↳ gfay1CYnCGItkoFmjk5hQ0165",
  "expires_in": 3599,
  "scope": "https://www.googleapis.com/auth/subscriptions https://www.googleapis.com/auth/
  ↳ experimentsandconfigs https://www.googleapis.com/auth/peopleapi.readonly https://
  ↳ www.googleapis.com/auth/cclog",
  "token_type": "Bearer"
}
```

This information should have been sent using the HTTP POST body. Note that the use of HTTPS is not an effective mitigation of this risk.

This issue has been rated to a low severity because a `blinded_token` is needed in order to perform the next request to the Zinc API.

Recommendation

While the application currently uses a POST request to send data, sensitive information (such as the `client_secret` and `refresh_token` parameters) should only be sent within the message body of the POST requests and not within the URL. Likewise, on the server-side, ensure that sensitive data is only accepted within the message body and never from the URL.

Reproduction Steps

1. Download and install Burp Suite/MitMProxy
2. Set the proxy up on 127.0.0.1:8080
3. Install the Burp/MitMProxy CA as Trusted Root Certification Authority on the local Windows machine.
4. Use a third-party application such as ProxyCap to force network traffic through the proxy.
5. Run the G1 application and select "Use VPN".
6. Intercept HTTPS traffic using the proxy.
7. Check that the POST request sent the parameters `client_secret` and `refresh_token` in the URL to `oauth2.googleapis.com` API.

Location

- **Krypton library**: DesktopOAuth::RefreshAccessToken() [krypton\desktop\desktop_oauth.cc - line 159]

Low

Lack of Certificate Pinning

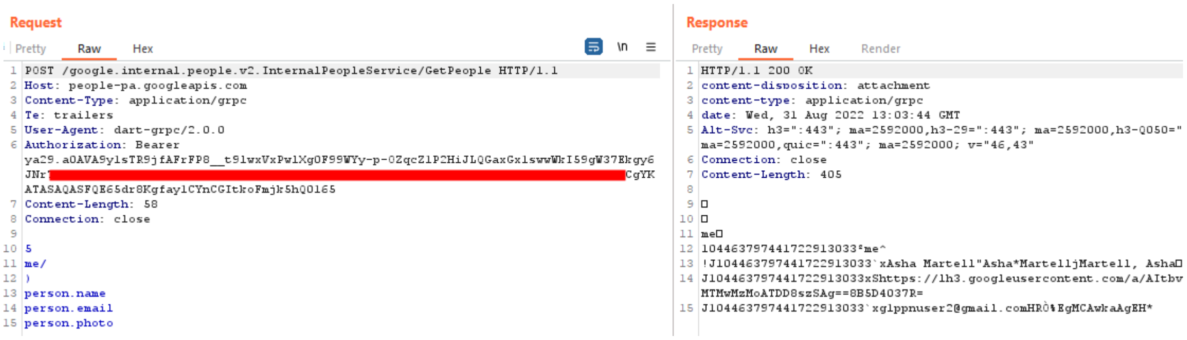
Overall Risk	Low	**Finding ID**	NCC-GOLE021-J2Q
Impact	Medium	**Component**	Windows Application
Exploitability	Low	**Category**	Cryptography
		Status	Reported

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 laptop 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.



```
POST /google.internal.people.v2.InternalPeopleService/GetPeople HTTP/1.1
Host: people-pa.googleapis.com
Content-Type: application/grpc
Te: trailers
User-Agent: dart-grpc/2.0.0
Authorization: Bearer
ya29.a0AVA9ylsTR9jfAFrFP8__t9lwxVxPwlXg0F99WYy-p-0ZqcZ1P2HiJLQGaxGxlswwWkI59gW37Ekgy6
JNr[redacted]CgYK
ATASAQASFQE65dr8Kgfay1CYnCGItkoFmjk5hQ0165
Content-Length: 58
Connection: close

5
me/
)
person.name
person.email
person.photo
```

```
HTTP/1.1 200 OK
content-disposition: attachment
content-type: application/grpc
date: Wed, 31 Aug 2022 13:03:44 GMT
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000,h3-Q050="
ma=2592000,quic=":443"; ma=2592000; v="46,43"
Connection: close
Content-Length: 405

□
□
me□
104463797441722913033²me^
!J104463797441722913033`xAsha Martell"Asha*MartelljMartell, Asha□
J104463797441722913033xShttps://lh3.googleusercontent.com/a/AItbv
MTMwMzMoATDD8szSAg==8B5D4037R=
J104463797441722913033`xglppnuser2@gmail.comHRÒ$EgMCAwkaAgEH*
```

Figure 8: Lack of certificate pinning

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 years[11,12], 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.

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 strong protection, as the attack surface is reduced to the specific CA pinned, and not the whole PKI infrastructure.

[11] DigiNotar - Issuance of fraudulent certificates: https://en.wikipedia.org/wiki/DigiNotar#Issuance_of_fraudulent_certificates
[12] Comodo - Certificate hacking: https://en.wikipedia.org/wiki/Comodo_Group#Certificate_hacking

In addition, it is considered a good practice to pin more than one certificate, especially when pinning individual host certificates, to reduce the risk of issues associated to a specific certificate, such as an expired certificate.

Reproduction Steps

1. Install the Burp Proxy CA in the Keychain.
2. Intercept the application’s SSL traffic to pass through an interception proxy such as Burp Proxy.
3. Verify that TLS traffic can be decrypted.

Info

Application Vulnerable to DLL Injection

Overall Risk	Informational	**Finding ID**	NCC-GOLE021-DDJ
Impact	Low	**Component**	Windows Application
Exploitability	None	**Category**	Configuration
		Status	Reported

Impact

Under specific circumstances the application could load and process malicious DLL files.

Description

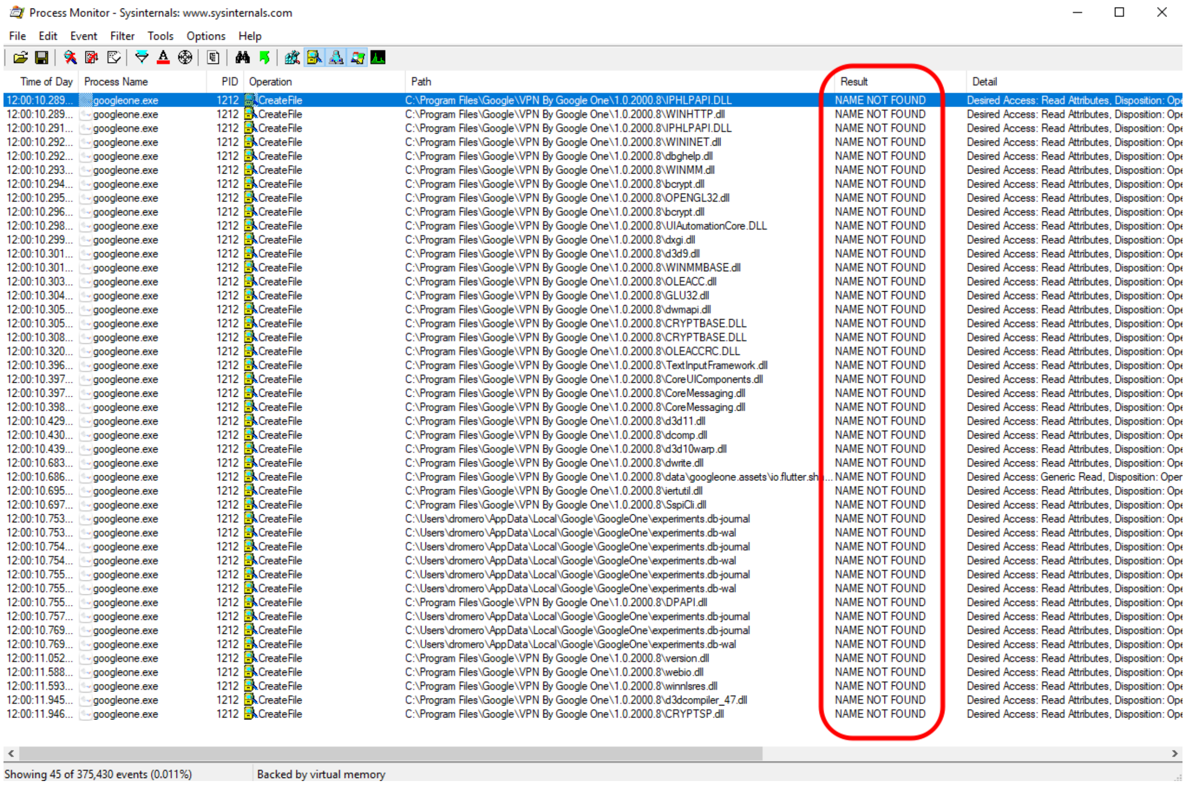
The main executable named `googleone.exe` was traced and analysed as part of the assessment, and was found potentially vulnerable to DLL injection as DLLs were loaded in a default, insecure manner.

When DLLs are loaded without a fully-qualified path name being specified, Windows will search through a known set of directories. Should an attacker place a file with the same name as the DLL in one of these directories, and the system finds it before the correct version, it will be loaded; providing a vector for possible privilege escalation or other attacks.

It should be noted that when an application needs to load a DLL it will go through the following order:

- The directory from which the application is loaded
- C:\Windows\System32
- C:\Windows\System
- C:\Windows
- The current working directory
- Directories in the system PATH environment variable
- Directories in the user PATH environment variable

The following picture from *Process Monitor* software shows the analysis of the application where a number of DLLs being loaded without success by the application when executed:



Process Monitor - Sysinternals: www.sysinternals.com
File Edit Event Filter Tools Options Help

Time of Day	Process Name	PID	Operation	Path	Result	Detail
12:00:10.289...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\IPHLPAPI.DLL	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.289...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\WINHTTP.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.291...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\IPHLPAPI.DLL	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.292...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\WININET.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.292...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\dbghelp.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.293...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\WINMM.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.294...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\bcrypt.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.295...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\OPENGL32.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.296...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\bcrypt.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.298...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\UIAutomationCore.DLL	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.299...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\dxgi.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.301...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\d3d9.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.301...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\WINMMBASE.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.303...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\OLEACC.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.304...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\GLU32.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.305...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\dwmapi.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.305...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\CRYPTBASE.DLL	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.308...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\CRYPTBASE.DLL	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.320...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\OLEACCRC.DLL	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.396...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\TextInputFramework.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.397...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\CoreUIComponents.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.397...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\CoreMessaging.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.398...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\CoreMessaging.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.429...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\d3d11.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.430...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\dcomp.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.439...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\d3d10warp.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.683...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\dwrite.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.686...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\data\googleone.assets\io.flutter.sh...	NAME NOT FOUND	Desired Access: Generic Read, Disposition: Oper
12:00:10.695...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\iertutil.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.697...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\SspiCli.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.753...	googleone.exe	1212	CreateFile	C:\Users\dromero\AppData\Local\Google\GoogleOne\experiments.db-journal	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.753...	googleone.exe	1212	CreateFile	C:\Users\dromero\AppData\Local\Google\GoogleOne\experiments.db-wal	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.754...	googleone.exe	1212	CreateFile	C:\Users\dromero\AppData\Local\Google\GoogleOne\experiments.db-journal	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.754...	googleone.exe	1212	CreateFile	C:\Users\dromero\AppData\Local\Google\GoogleOne\experiments.db-wal	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.755...	googleone.exe	1212	CreateFile	C:\Users\dromero\AppData\Local\Google\GoogleOne\experiments.db-journal	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.755...	googleone.exe	1212	CreateFile	C:\Users\dromero\AppData\Local\Google\GoogleOne\experiments.db-wal	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.755...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\DPAPI.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.757...	googleone.exe	1212	CreateFile	C:\Users\dromero\AppData\Local\Google\GoogleOne\experiments.db-journal	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.769...	googleone.exe	1212	CreateFile	C:\Users\dromero\AppData\Local\Google\GoogleOne\experiments.db-journal	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:10.769...	googleone.exe	1212	CreateFile	C:\Users\dromero\AppData\Local\Google\GoogleOne\experiments.db-wal	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:11.052...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\version.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:11.588...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\webio.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:11.593...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\winnlsres.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:11.945...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\d3dcompiler_47.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope
12:00:11.946...	googleone.exe	1212	CreateFile	C:\Program Files\Google\VPN By Google One\1.0.2000.8\CRYPTSP.dll	NAME NOT FOUND	Desired Access: Read Attributes, Disposition: Ope

Showing 45 of 375,430 events (0.011%) Backed by virtual memory

Figure 9: DLLs loaded upon execution of the main executable

It can be observed that the installer was looking for a number of DLLs placed in the location where the installer is run with `NAME NOT FOUND` results, which means that if an attacker is able to place a malicious file named as `IPHLPAPI.DLL` (as highlighted in the above evidence) within the same folder (such as `C:\Program Files\Google\VPN By Google One\1.0.2000.8` in this case) an attacker could perform malicious activities; such as remotely obtain control over the user's computer.[13] [14] [15] [16]

All DLL files were obtained from `C:\Windows\System32`, as none of them were missing (which may cause the file to be obtained from non-privileged folders) it would require administrator privileges to perform this attack and therefore this finding was rated as informational.

Recommendation

Several methods exist for ensuring that only the correct DLL is loaded. When using `LoadLibrary`, `LoadLibraryEx`, `CreateProcess`, or `ShellExecute`, specify fully-qualified paths:

- Use the `LOAD_LIBRARY_SEARCH` flag with `LoadLibraryEx`.

[13] Dynamic-Link Library Security - http://msdn.microsoft.com/en-us/library/windows/desktop/ff919712(v=vs.85).aspx
[14] Dynamic-Link Library Search Order - http://msdn.microsoft.com/en-us/library/windows/desktop/ms682586(v=vs.85).aspx
[15] Dynamic-Link Library Redirection (Windows) - http://msdn.microsoft.com/en-us/library/windows/desktop/ms682600(v=vs.85).aspx
[16] Microsoft Sysinternals Suite - https://docs.microsoft.com/en-us/sysinternals/downloads/sysinternals-suite

- Use the `LOAD_LIBRARY_SEARCH` flag `SetDefaultDllDirectories`. This will allow you to establish the order in which the process will search for DLLs. `AddDllDirectory` or `SetDllDirectory` can be used to alter this list.
- Use DLL redirection. This involves creating an empty file called «appname».local, which should be placed in the application directory, as should the DLLs to be used.

Info

Sensitive Information Written to Debug Logs

Overall Risk	Informational	**Finding ID**	NCC-GOLE021-BQL
Impact	Undetermined	**Component**	Windows Application
Exploitability	None	**Category**	Data Exposure
		Status	Fixed

Impact

An attacker who obtains access to the application’s log files would be able to view service access information such as authentication and authorization tokens, and sensitive parameters such as client_secret, client_id, and PKCE codes.

This information could be used to authorize another application to use the VPN service while the OAuth token remains valid.

Description

Authentication (OAuth) and authorization (blinded) tokens, as well as other sensitive information were written to debug log files. A suitably-placed attacker or malicious user would have access to those files, gaining access to the service for an undetermined amount of time.

An example of sensitive data being written to log files can be seen below, where the OAuth-related requests are registered:

```
I0902 07:52:52.686767    9868 http_fetcher.cc:92] Requesting WinHttpConnect for url: https://
↳ prod.zinc.cloud.cupronickel.goog/auth
I0902 07:52:52.686955    9868 http_fetcher.cc:103] WinHttpConnect successful
I0902 07:52:52.687215    9868 http_fetcher.cc:119] WinHttpOpenRequest successful
I0902 07:52:52.687399    9868 http_fetcher.cc:143] WinHttpAddRequestHeaders successful
I0902 07:52:52.687615    9868 http_fetcher.cc:148] Json_body: {"blinded_token":
↳ ["gM6gBYrXF4wZPh0owyfbGLY+qEOaxHK/U5kQPWECQ+IhQIFPL0gl/
↳ UleSD7mNslEmzWvINvWg8udRKfwr5jnU9FDXJRElZhQhHUQu/
↳ TWXzrl5ENHoKxMFMNKwI7bxH6KlGcx6Z2JIlrGcfbNQSMn8ydrJssiA85hv1C+H7Zn2fq4bRMWqLSvA9WBUdSgPqYdz+0
↳ QtKPr4sZL4pYEAViTzEVmUS6B3lQs6/hXRoVU3cAGRb11JNkEIkQQaeVHzB0h0Ndy+Hv/ml/
↳ p7YIL02g8EWVia2Zeh6HBI9O1eb6aMOfUpGActtzgAtuPUIRHKF5kDTsxVOdFVIjPRCABtXSzZQ=="],"oauth_token"
↳ :"ya29.a0AVA9y1tTVcH_lsjZo4kRB79M6nrsZqVf2sYPlgFna19RSkYnK1egb0BmRZkhPwgMzQ87tZYhmbsz2vsd2xxG
↳ _O-REdOEs12gMB6tvLK3-_L-1eKxSc3DnWhJb6xKLL4bXtgu1vNHI7-
↳ U91yw5qS7RdG39cAuwwaCgYKATASAQASFQE65dr8qmcgRYsGHNgUyI8ipRNEXQ0165","public_key_hash":"xy0GxT
↳ H9BnDrPD/OEIG3+UErjPFBlSe4q5Rr5Dbx5bw=","service_type":"g1"}
 Length:681
I0902 07:52:52.887660    9868 http_fetcher.cc:157] WinHttpSendRequest successful
```

There are other instances where client information is also stored in debug logs:

```
I0902 07:53:09.709673    1816 network_monitor.cc:493] The best network interface has not
↳ changed.
I0902 07:56:55.740545    1508 http_fetcher.cc:51] Calling HttpFetcher postJson Windows method
↳ to https://oauth2.googleapis.com/token?
↳ client_id=874847826917-0rfhjap59nhp8fpun56mm51sshkfjd2f.apps.googleusercontent.com&client_sec
↳ ret=2ndJjPpvzenW2pGSRy-
↳ KYddM&grant_type=refresh_token&refresh_token=1%2F%2F032SarWRZ702eCgYIARAAGAMSNwF-
↳ L9Irklhy1lRgUARdbJdD2xbgHK3R0gVLrvD1kz98Jyrcb5Mf6bdnN2FE0fwHUd6KEs-fwPA
I0902 07:56:55.741139    1508 http_fetcher.cc:77] WinHttpCrackUrl successful
```

The severity for this issue was set to informational because this was only observed in debug logs.

Recommendation

Review the discovered instances and ensure that sensitive information is suitably stripped before being committed to log files.

Investigate whether static analysis could be utilized to identify new instances of this class of issue during development.

Reproduction Steps

1. Install and run the Google One application normally (including the VPN service).
2. Navigate to the debug log folder:
    - (`C:\Users\<USER>\AppData\Local\Google\GoogleOne\debug\`).
3. Search sensitive information, such as “oAuth” or “token” in all files.

Location

C:\Users\<USER>\AppData\Local\Google\GoogleOne\debug\ppn_debug_20220902_075251.9400897.txt

Info

Application Binaries Not Obfuscated

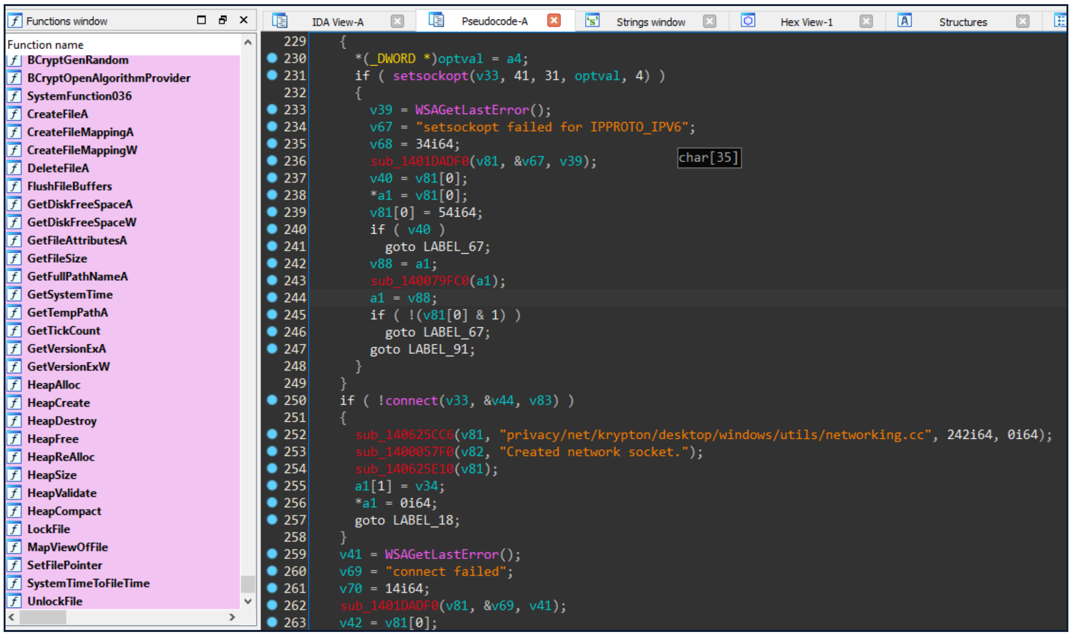
Overall Risk	Informational	**Finding ID**	NCC-GOLE021-GUM
Impact	Undetermined	**Component**	Windows Application
Exploitability	Undetermined	**Category**	Configuration
		Status	Reported

Impact

The lack of code obfuscation means that it is relatively easy to determine the structure and functionality of the application, and to analyse the binary for suitable locations to modify the application behavior.

Description

Application binaries and libraries were not obfuscated by using an executable binary obfuscator, as shown in the image below.



```
Function name
BCryptGenRandom
BCryptOpenAlgorithmProvider
SystemFunction036
CreateFileA
CreateFileMappingA
CreateFileMappingW
DeleteFileA
FlushFileBuffers
GetDiskFreeSpaceA
GetDiskFreeSpaceW
GetFileAttributesA
GetFileSize
GetFullPathNameA
GetSystemTime
GetTempPathA
GetTickCount
GetVersionExA
GetVersionExW
HeapAlloc
HeapCreate
HeapDestroy
HeapFree
HeapReAlloc
HeapSize
HeapValidate
HeapCompact
LockFile
MapViewOfFile
SetFilePointer
SystemTimeToFileTime
UnlockFile
```

```
    {
      *(_DWORD *)optval = a4;
      if ( setsockopt(v33, 41, 31, optval, 4) )
      {
        v39 = WSAGetLastError();
        v67 = "setsockopt failed for IPPROTO_IPV6";
        v68 = 34i64;
        sub_1401DADF0(v81, &v67, v39);
        v40 = v81[0];
        *a1 = v81[0];
        v81[0] = 54i64;
        if ( v40 )
          goto LABEL_67;
        v88 = a1;
        sub_140079FC0(a1);
        a1 = v88;
        if ( !(v81[0] & 1) )
          goto LABEL_67;
        goto LABEL_91;
      }
    }
    if ( !connect(v33, &v44, v83) )
    {
      sub_140625CC6(v81, "privacy/net/krypton/desktop/windows/utils/networking.cc", 242i64, 0i64);
      sub_1400057F0(v82, "Created network socket.");
      sub_140625E10(v81);
      a1[1] = v34;
      *a1 = 0i64;
      goto LABEL_18;
    }
    v41 = WSAGetLastError();
    v69 = "connect failed";
    v70 = 14i64;
    sub_1401DADF0(v81, &v69, v41);
    v42 = v81[0];
```

Figure 10: Application binaries are not obfuscated

Recommendation

Consider using a binary or source-level obfuscation system to obfuscate the application. Additionally, commercially available binary obfuscation systems may also provide runtime protection of memory contents.

However, it is worth noting that all binary obfuscation can be defeated, and only serves to deter casual attackers and slow down skilled and motivated attackers. Additionally, due to the way binary executable obfuscation works, using it may remove platform-level mitigations and thus weaken the authentication system's security posture in other ways. It is worth noting that code signing will typically be mutually exclusive with binary obfuscation, requiring a business decision to be made as to whether to make the authentication system more difficult to reverse engineer or more difficult to modify.

Info

Binaries Contained Debug Information

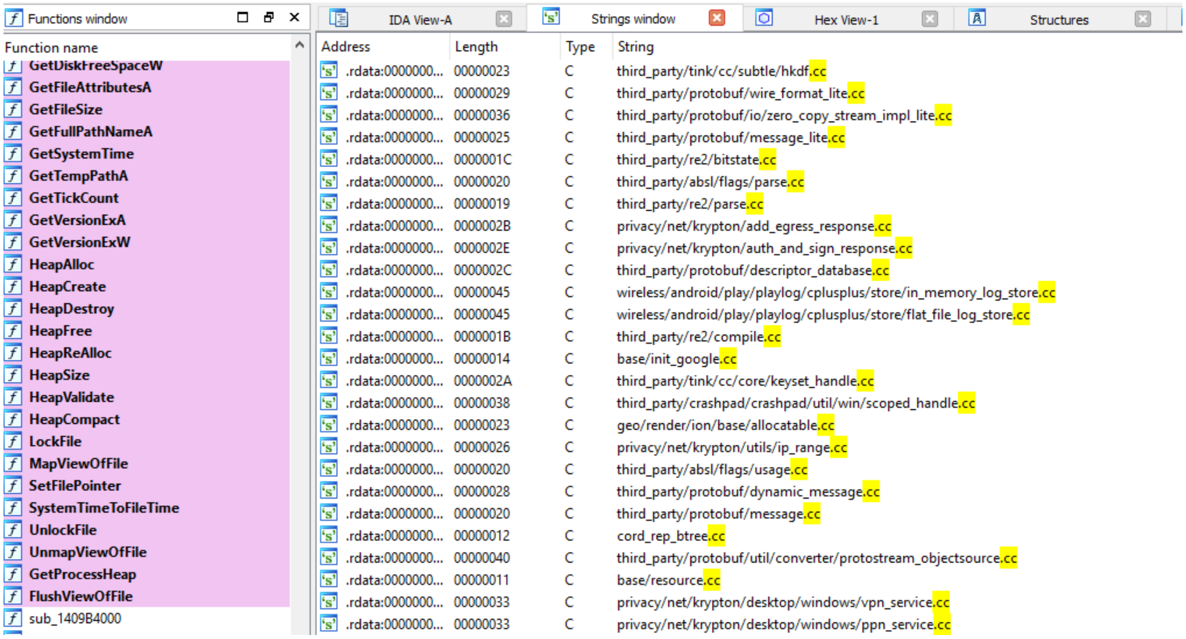
Overall Risk	Informational	**Finding ID**	NCC-GOLE021-UEX
Impact	Undetermined	**Component**	Windows Application
Exploitability	Undetermined	**Category**	Data Exposure
		Status	Reported

Impact

While not a direct security issue, the presence of this information in the binaries makes reverse engineering efforts much simpler. Reverse engineering all or part of a binary is often a key step in producing a reliable exploit.

Description

Application binaries and DLLs contained debug information. Evidence of the issue can be seen in the screenshot below:



Function name
GetDiskFreeSpaceW
GetFileAttributesA
GetFileSize
GetFullPathNameA
GetSystemTime
GetTempPathA
GetTickCount
GetVersionExA
GetVersionExW
HeapAlloc
HeapCreate
HeapDestroy
HeapFree
HeapReAlloc
HeapSize
HeapValidate
HeapCompact
LockFile
MapViewOfFile
SetFilePointer
SystemTimeToFileTime
UnlockFile
UnmapViewOfFile
GetProcessHeap
FlushViewOfFile
sub_1409B4000

Address	Length	Type	String
.rdata:0000000...	00000023	C	third_party/tink/cc/subtle/hkdf.cc
.rdata:0000000...	00000029	C	third_party/protobuf/wire_format_lite.cc
.rdata:0000000...	00000036	C	third_party/protobuf/io/zero_copy_stream_impl_lite.cc
.rdata:0000000...	00000025	C	third_party/protobuf/message_lite.cc
.rdata:0000000...	0000001C	C	third_party/re2/bitstate.cc
.rdata:0000000...	00000020	C	third_party/absl/flags/parse.cc
.rdata:0000000...	00000019	C	third_party/re2/parse.cc
.rdata:0000000...	0000002B	C	privacy/net/krypton/add_egress_response.cc
.rdata:0000000...	0000002E	C	privacy/net/krypton/auth_and_sign_response.cc
.rdata:0000000...	0000002C	C	third_party/protobuf/descriptor_database.cc
.rdata:0000000...	00000045	C	wireless/android/play/playlog/cplusplus/store/in_memory_log_store.cc
.rdata:0000000...	00000045	C	wireless/android/play/playlog/cplusplus/store/flat_file_log_store.cc
.rdata:0000000...	0000001B	C	third_party/re2/compile.cc
.rdata:0000000...	00000014	C	base/init_google.cc
.rdata:0000000...	0000002A	C	third_party/tink/cc/core/keyset_handle.cc
.rdata:0000000...	00000038	C	third_party/crashpad/crashpad/util/win/scoped_handle.cc
.rdata:0000000...	00000023	C	geo/render/ion/base/allocatable.cc
.rdata:0000000...	00000026	C	privacy/net/krypton/utils/ip_range.cc
.rdata:0000000...	00000020	C	third_party/absl/flags/usage.cc
.rdata:0000000...	00000028	C	third_party/protobuf/dynamic_message.cc
.rdata:0000000...	00000020	C	third_party/protobuf/message.cc
.rdata:0000000...	00000012	C	cord_rep_btree.cc
.rdata:0000000...	00000040	C	third_party/protobuf/util/converter/protostream_objectsource.cc
.rdata:0000000...	00000011	C	base/resource.cc
.rdata:0000000...	00000033	C	privacy/net/krypton/desktop/windows/vpn_service.cc
.rdata:0000000...	00000033	C	privacy/net/krypton/desktop/windows/ppn_service.cc

Figure 11: Binaries contained debug information

Recommendation

Software build rules should be modified to remove debug symbols in production releases. Debug strings should be out of the production code. If this is a concern for diagnosing field returns, then consider replacing them with macros that revert to simple file and line number prints (for example using **FILE** and **LINE** macros). Post-processing of the now obfuscated logs can then be used to recover the original print statements for debugging purposes. Failure messages should be simplified so as not to reveal the detailed operation of the program. Avoid using function names and consider simple error numbers.

Location

Windows Application

7 Finding Details – iOS Application

Low

Application Disables App Transport Security

Overall Risk	Low	**Finding ID**	NCC-GOLE021-VPB
Impact	Medium	**Component**	iOS Application
Exploitability	Low	**Category**	Cryptography
		Status	Reported

Impact

When ATS is disabled by the application, an attacker is more easily able to intercept HTTP traffic between the mobile application and back-end services.

Description

As of iOS version 9, a new App Transport Security (ATS) feature was added that can enhance the security afforded to data in transit. When ATS is fully enabled, a mobile application's HTTP connections must use HTTPS and meet certain minimum certificate, protocol, and cipher suite criteria. In iOS this feature is enabled by default, but was found to be explicitly disabled for the Google One application.

This is demonstrated by observing that within the application's `plist` file, the `NSAllowsArbitraryLoads` flag was set to `true`, as shown below:

<prod.app/Info.plist>

```
<key>NSAppTransportSecurity</key>
<dict>
    <key>NSAllowsArbitraryLoads</key>
    <true/>
</dict>
```

Recommendation

The `NSAllowsArbitraryLoads` flag should be explicitly set to `false`.[17]

Reproduction Steps

Run objection framework[18]:

1. $ objection -g com.google.one explore
2. com.google.one on (iPhone: 14.7.1) [usb] # ios plist cat Info.plist
3. Search for NSAppTransportSecurity, check that NSAllowsArbitraryLoads is set to = 1.

Location

ios/Info.plist

[17] App Transport Security: https://developer.apple.com/library/archive/documentation/General/Reference/InfoPlistKeyReference/Articles/CocoaKeys.html#//apple_ref/doc/uid/TP40009251-SW33,https://developer.apple.com/library/archive/documentation/General/Reference/InfoPlistKeyReference/Articles/CocoaKeys.html#//apple_ref/doc/uid/TP40009251-SW34

[18] Objection https://github.com/sensepost/objection.

Low

Mobile Application Data Storage Leaks GAIA ID in Log Files

Overall Risk	Low	**Finding ID**	NCC-GOLE021-XUH
Impact	Low	**Component**	iOS Application
Exploitability	Low	**Category**	Data Exposure
		Status	Fixed

Impact

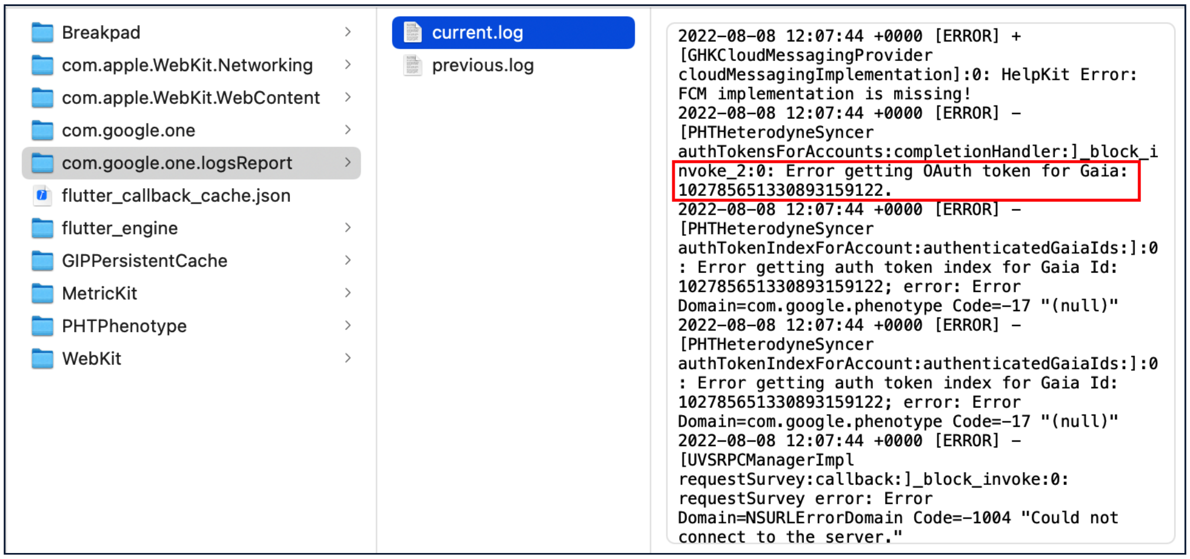
By knowing a user's GAIA ID, an attacker can search it in other Google services, and find out more details about the user.

Description

Potentially sensitive data such as the GAIA ID was observed being logged by the application. Please note that on modern devices, sandboxing prevents apps from reading the logs of other apps on the device, limiting the exploitability of this issue.

Historically, mobile operating systems provided relatively weak protections against attacks across different mobile applications, and attacks that assume physical access to a locked device. All recent versions of iOS now guarantee strong sandboxing and device-level encryption which, in most cases, completely eliminates the impact of those attack vectors under a typical threat model. For most applications, no additional protections are needed beyond making use of the recommended data storage functionality provided by the operating system. If an attacker gains access to a user's unlocked mobile device, they may be able to read sensitive application data that would not otherwise be accessible.

The iOS application stored the user's GAIA ID in log files in the device, as shown below:


Breakpad
com.apple.WebKit.Networking
com.apple.WebKit.WebContent
com.google.one
com.google.one.logsReport
flutter_callback_cache.json
flutter_engine
GIPPersistentCache
MetricKit
PHTPhenotype
WebKit
current.log
previous.log

```
2022-08-08 12:07:44 +0000 [ERROR] +
[GHKCloudMessagingProvider
cloudMessagingImplementation]:0: HelpKit Error:
FCM implementation is missing!
2022-08-08 12:07:44 +0000 [ERROR] -
[PHTHeterodyneSyncer
authTokensForAccounts:completionHandler:]_block_i
nvoke_2:0: Error getting OAuth token for Gaia:
102785651330893159122.
2022-08-08 12:07:44 +0000 [ERROR] -
[PHTHeterodyneSyncer
authTokenIndexForAccount:authenticatedGaiaIds:]:0
: Error getting auth token index for Gaia Id:
102785651330893159122; error: Error
Domain=com.google.phenotype Code=-17 "(null)"
2022-08-08 12:07:44 +0000 [ERROR] -
[PHTHeterodyneSyncer
authTokenIndexForAccount:authenticatedGaiaIds:]:0
: Error getting auth token index for Gaia Id:
102785651330893159122; error: Error
Domain=com.google.phenotype Code=-17 "(null)"
2022-08-08 12:07:44 +0000 [ERROR] -
[UVSRPCManagerImpl
requestSurvey:callback:]_block_invoke:0:
requestSurvey error: Error
Domain=NSURLErrorDomain Code=-1004 "Could not
connect to the server."
```

Figure 12: iOS current logs

Recommendation

On iOS, all application data can generally be stored using the built-in Data Protection classes[19] which the OS provides for automatic file-based encryption. Critically-sensitive data such as passwords can be stored in the Keychain[20] using Keychain Item Attributes[21] to restrict access to the secret.

Reproduction Steps

1. $ sftp root@IP_jailbreak_device
2. $ cd /var/mobile/Containers/Data/Application/APP_ID/Caches/com.google.one.logsReport/
3. Download current.log file stored locally in the device

Location

`/var/mobile/Containers/Data/Application/APP_ID/Caches/com.google.one.logsReport/current.log`

[19] Apple Platform Security - Data Protection classes: https://support.apple.com/guide/security/data-protection-classes-secb010e978a/1/web/1
[20] Apple Developer Documentation - Keychain Services: https://developer.apple.com/documentation/security/keychain_services
[21] Apple Developer Documentation - Item Attribute Keys and Values: https://developer.apple.com/documentation/security/keychain_services/keychain_items/item_attribute_keys_and_values

Info

Mobile Application Backgrounding Leaks Sensitive Info in Screenshots

Overall Risk	Informational	**Finding ID**	NCC-GOLE021-RPK
Impact	None	**Component**	iOS Application
Exploitability	None	**Category**	Data Exposure
		Status	Reported

Impact

The application did not hide screenshots shown in the list of recent applications. With physical access to an unlocked device, an attacker may be able to see information in these screenshots.

Description

When an iOS application is closed (or backgrounded), iOS takes a snapshot of the current screen content and stores it in an encrypted, sandboxed file on the device. Information on the screen at the time of closing is therefore disclosed through this screenshot and can be read by an attacker who has physical access to the unlocked device.

Some snapshots of the current screen were stored locally in the file path `/Library/SplashBoard/Snapshot/sceneID/com.google.one-default/` on the device, as shown below:

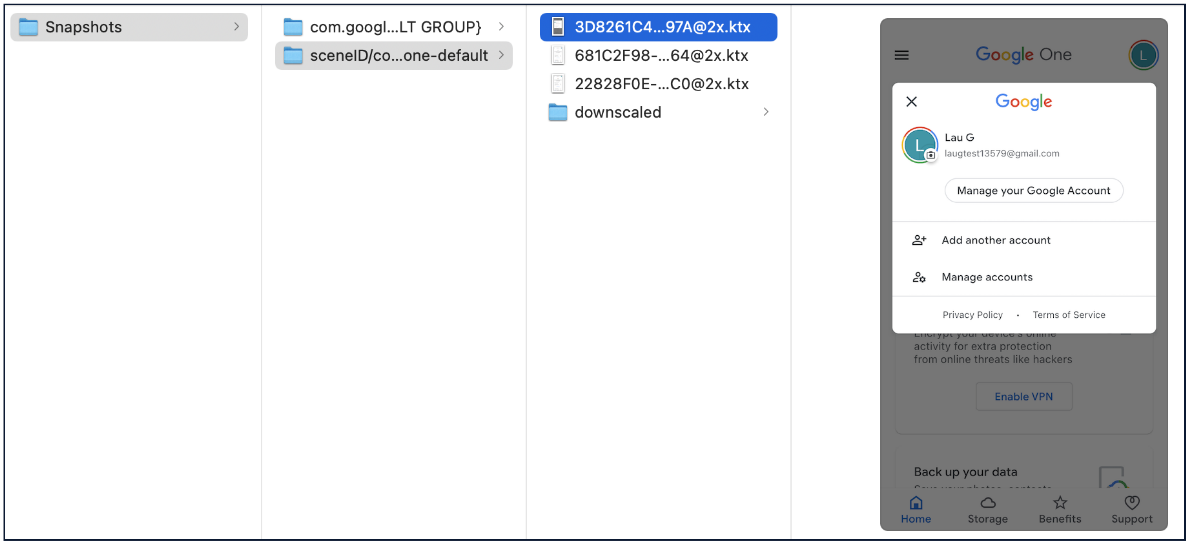


Figure 13: Mobile Application Backgrounding Leaks Info in Screenshots

Recommendation

On iOS, there are three unique events that need to be handled (in iOS 10+):

1. `func applicationWillResignActive(UIApplication)` – Triggered when the app is about to become inactive (for example, on a double-tap of the home button to bring up the task switcher)
2. `func applicationDidEnterBackground(UIApplication)` – Triggered when the app enters the background state (for example, when the home button is pressed)

3. `func applicationProtectedDataWillBecomeUnavailable(UIApplication)` – Triggered when protected files become unavailable (for example, when the screen lock button is pressed)

Event handlers should be used for all three events in order to hide sensitive information from being captured in screenshots when the application makes various state transitions. More information can be found on these events in Apple's documentation on the `UIApplicationDelegate`[22] object.

Reproduction Steps

1. $ sftp root@IP_jailbreak_device
2. $ cd /var/mobile/Containers/Data/Application/APP_ID/Library/SplashBoard/Snapshot/sceneID/com.google.one-default/
3. Download the snapshots stored locally in the device

Location

`/var/mobile/Containers/Data/Application/APP_ID/Library/SplashBoard/Snapshot/sceneID/com.google.one-default/`

[22] Apple Developer Documentation - UIApplicationDelegate: https://developer.apple.com/documentation/uikit/uiapplicationdelegate

8 Finding Details – macOS Application

Medium

Weaknesses in Authentication Process

Overall Risk	Medium	**Finding ID**	NCC-GOLE021-G96
Impact	Medium	**Component**	macOS Application
Exploitability	Low	**Category**	Denial of Service
		Status	Reported

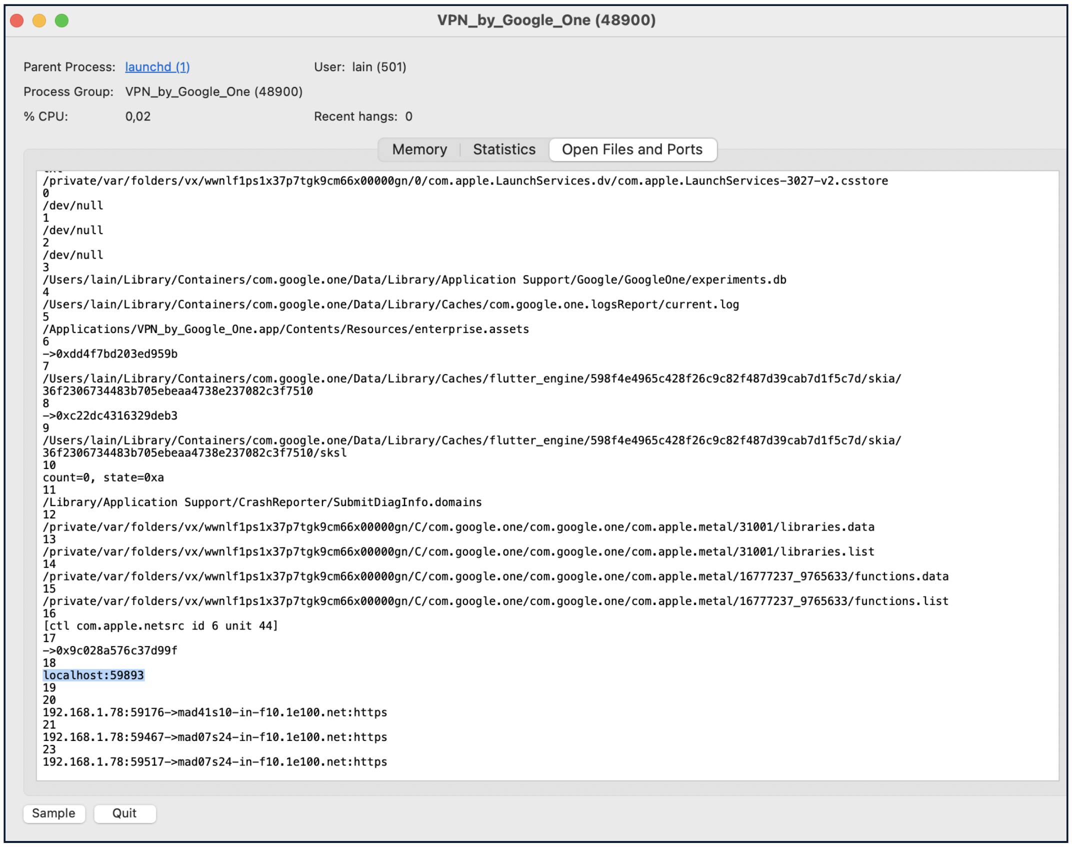
Impact

A local attacker could cause a Denial of Service condition by preventing users from successfully authenticating to the VPN service and may be able to compromise the OAuth token.

Description

NCC Group identified that by sending a crafted HTTP request to the open local port used for authentication, the application closes the port. Once this port is closed, an attacker can open this TCP port and masquerade as this process. This can allow a local attacker to obtain the OAuth token of the user attempting to authenticate to the VPN. Additionally, once the port is closed, a VPN user can no longer authenticate to the VPN service.

The following image shows a new port being opened after a user authenticates to the VPN service.



Parent Process: launchd (1)   User: lain (501)

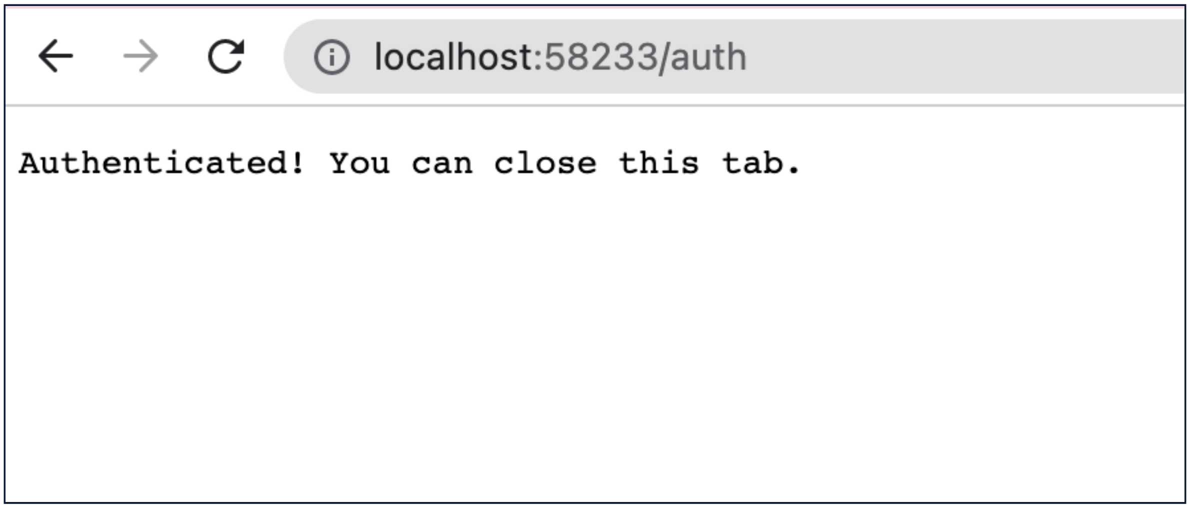
Process Group: VPN_by_Google_One (48900)

% CPU: 0,02   Recent hangs: 0

```
/private/var/folders/vx/wwnlf1ps1x37p7tgk9cm66x00000gn/0/com.apple.LaunchServices.dv/com.apple.LaunchServices-3027-v2.csstore
0
/dev/null
1
/dev/null
2
/dev/null
3
/Users/lain/Library/Containers/com.google.one/Data/Library/Application Support/Google/GoogleOne/experiments.db
4
/Users/lain/Library/Containers/com.google.one/Data/Library/Caches/com.google.one.logsReport/current.log
5
/Applications/VPN_by_Google_One.app/Contents/Resources/enterprise.assets
6
->0xdd4f7bd203ed959b
7
/Users/lain/Library/Containers/com.google.one/Data/Library/Caches/flutter_engine/598f4e4965c428f26c9c82f487d39cab7d1f5c7d/skia/
36f2306734483b705ebeaa4738e237082c3f7510
8
->0xc22dc4316329deb3
9
/Users/lain/Library/Containers/com.google.one/Data/Library/Caches/flutter_engine/598f4e4965c428f26c9c82f487d39cab7d1f5c7d/skia/
36f2306734483b705ebeaa4738e237082c3f7510/sksl
10
count=0, state=0xa
11
/Library/Application Support/CrashReporter/SubmitDiagInfo.domains
12
/private/var/folders/vx/wwnlf1ps1x37p7tgk9cm66x00000gn/C/com.google.one/com.google.one/com.apple.metal/31001/libraries.data
13
/private/var/folders/vx/wwnlf1ps1x37p7tgk9cm66x00000gn/C/com.google.one/com.google.one/com.apple.metal/31001/libraries.list
14
/private/var/folders/vx/wwnlf1ps1x37p7tgk9cm66x00000gn/C/com.google.one/com.google.one/com.apple.metal/16777237_9765633/functions.data
15
/private/var/folders/vx/wwnlf1ps1x37p7tgk9cm66x00000gn/C/com.google.one/com.google.one/com.apple.metal/16777237_9765633/functions.list
16
[ctl com.apple.netsrc id 6 unit 44]
17
->0x9c028a576c37d99f
18
localhost:59893
19
20
192.168.1.78:59176->mad41s10-in-f10.1e100.net:https
21
192.168.1.78:59467->mad07s24-in-f10.1e100.net:https
23
192.168.1.78:59517->mad07s24-in-f10.1e100.net:https
```

Figure 14: Authentication open port

Once the user is authenticated through the web browser, it connects to the localhost port to send the authentication token:


localhost:58233/auth

Authenticated! You can close this tab.

Figure 15: Authentication successfully

By looking into the application's implementation, NCC Group identified that if the port is reached with an authentication GET request without parameters (e.g. `http://localhost:58233/auth`), the application closes the listening port. Therefore, a malicious application without administrator privileges could monitor the open ports, send a request to close them (causing a denial of service by not allowing the authentication to succeed) and then open the same port again to capture the OAuth token.

Recommendation

Ensure that the authentication code can properly handle unexpected user data. Alternatively, implement the authentication directly in the Google One VPN application, which would avoid the use of external applications (such as the web browser) and the need of opening a local port to receive the OAuth token.

Location

- g1_macos/app/auth/lib/auth.dart

Low

Sensitive Data Sent in the URL Using POST Method

Overall Risk	Low	**Finding ID**	NCC-GOLE021-6NH
Impact	Low	**Component**	macOS Application
Exploitability	Low	**Category**	Data Exposure
		Status	Reported

Impact

An attacker could intercept the communications between the thick application and the server, allowing them the ability to retrieve valid OAuth access tokens.

Description

Authentication requests to the OAuth2 API `oauth2.googleapis.com` were submitted in the URL using the HTTP POST method. This resulted in secrets such as `client_secret` and `refresh_token` being sent as plaintext in the URL. These secrets should be sent in the body of the POST request instead of the URL.

This meant that secrets could be stored on any proxy servers between the thick application and the server.

Note that the use of HTTPS is not an effective mitigation of this risk.

This issue has been rated to a low severity because a `blinded_token` is needed in order to perform the next request to the Zinc API.

Recommendation

While the application currently uses a POST request to send data, sensitive information (such as the `client_secret` and `refresh_token` parameters) should only be sent within the message body of the POST requests and not within the URL. Likewise, on the server-side, ensure that sensitive data is only accepted within the message body and never from the URL.

Reproduction Steps

1. Download and install Burp Suite
2. Set the proxy up on 127.0.0.1:8080
3. Install the Burp CA as Trusted Root on the Keychain
4. Go to: to macOS System Preferences -> Network -> Proxies and configured HTTP
5. HTTPS to proxy through 127.0.0.1:8080
6. Run the G1 application and "Use VPN"
7. Intercept HTTPS traffic using Burp Proxy
8. Check that the POST request sent the parameters `client_secret` and `refresh_token` in the URL to `oauth2.googleapis.com` API.

Request:

```
POST /token?
↳ client_id=874847826917-0rfhjap59nhp8fpun56mm51sshkfjd2f.apps.googleusercontent.com&client_sec
↳
↳
```

```
↪ ret=2ndS...YddM&grant_type=refresh_token&refresh_token=1%2F%2F03eNeAznG2d..XOKuZmF51No956DVFH
↪ w-ciY5WfHZRGDK_Ug0HfoyC4enPrC9J8 HTTP/1.1
Host: oauth2.googleapis.com
Accept: */*
Content-Type: application/json
Content-Length: 0
User-Agent: com.google.one.NetworkExtension/1.0.081013.0 MacOSX/12.5.1
Accept-Language: en-GB,en;q=0.9
Accept-Encoding: gzip, deflate
Connection: close
```

Response:

```
HTTP/2 200 OK
Date: Mon, 29 Aug 2022 10:24:32 GMT
Pragma: no-cache
Expires: Mon, 01 Jan 1990 00:00:00 GMT
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Content-Type: application/json; charset=utf-8
Vary: Origin
Vary: X-Origin
Vary: Referer
Server: scaffolding on HTTPServer2
Content-Length: 487
X-Xss-Protection: 0
X-Frame-Options: SAMEORIGIN
X-Content-Type-Options: nosniff
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000,h3-Q050=":443"; ma=2592000,h3-Q046=":
↪ 443"; ma=2592000,h3-Q043=":443"; ma=2592000,quic=":443"; ma=2592000; v="46,43"

{
  "access_token": "ya29.a0AVA9y1v-cGns6vS0jaOaPTS3CxbfI9...npez0-
  ↪ bStpQ3vkewaCgYKATASAQASFQE65dr8GVGwnu4qDJj8a8jiVHFy1g0165",
  "expires_in": 3599,
  "scope": "https://www.googleapis.com/auth/experimentsandconfigs https://www.googleapis.com/
  ↪ auth/peopleapi.readonly https://www.googleapis.com/auth/cclog https://www.googleapis.com/
  ↪ auth/subscriptions",
  "token_type": "Bearer"
}
```

Location

- macOS Application

Low

Application Vulnerable to DYLIB Hijacking

Overall Risk	Low	**Finding ID**	NCC-GOLE021-GMU
Impact	Low	**Component**	macOS Application
Exploitability	Low	**Category**	Configuration
		Status	Reported

Impact

Under specific circumstances the application could load and process malicious DYLIB files.

Description

The main executable named `VPN_by_Google_One` was traced and analyzed as part of the assessment, and was found potentially vulnerable to DYLIB injection as the DYLIB files were loaded in a default, insecure manner.

The following scenarios could be a vector for possible privilege escalation or other attacks:

- LC_LOAD_WEAK_DYLIB that reference a non-existent DYLIB.
- LC_LOAD*_DYLIB with @rpath'd import and multiple LC_RPATHs with the run-path dependent library not found in a primary run-path search path.

The following command shows all the DYLIBs being loaded by the application when executed:[23]

```
$ otool -l /Applications/VPN_by_Google_One.app/Contents/MacOS/VPN_by_Google_One

...
Load command 20
         cmd LC_LOAD_WEAK_DYLIB
      cmdsize 112
         name /System/Library/Frameworks/SystemExtensions.framework/Versions/A/
         ↳SystemExtensions (offset 24)
   time stamp 2 Thu Jan  1 01:00:02 1970
      current version 1.0.0
compatibility version 1.0.0
...
```

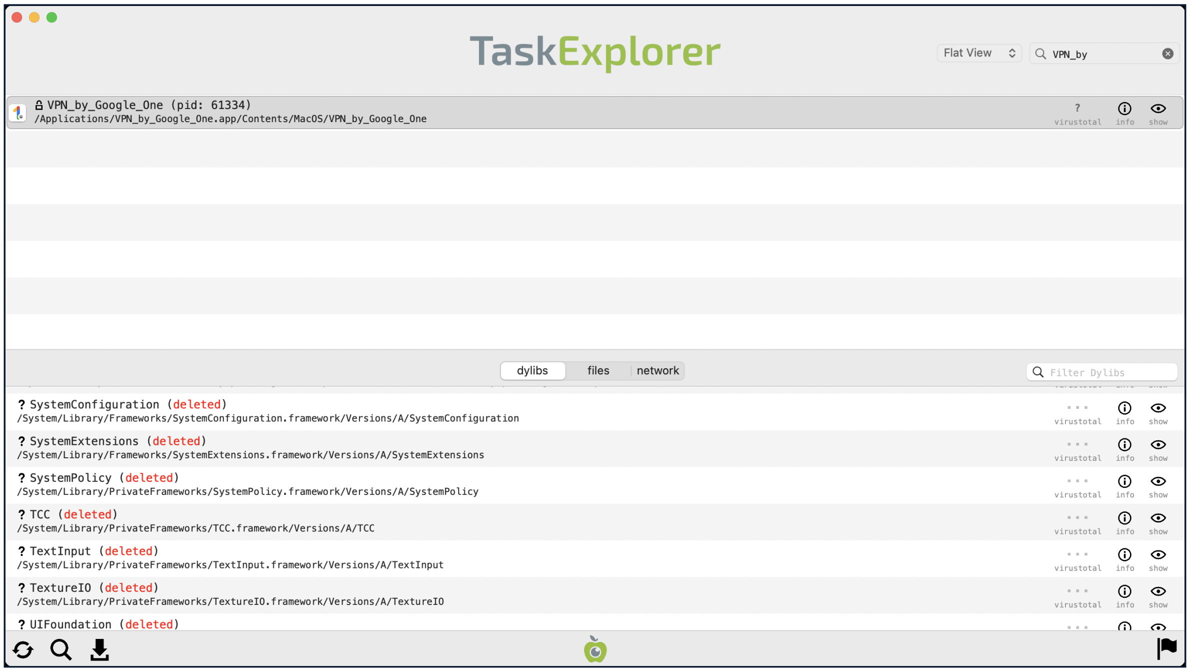
A file search shows that the DYLIB was not found in the system:

```
$ ls /System/Library/Frameworks/SystemExtensions.framework/Versions/A/SystemExtensions
ls: /System/Library/Frameworks/SystemExtensions.framework/Versions/A/SystemExtensions: No such
↳file or directory
```

If weak linking is used, such as the LC_LOAD_WEAK_DYLIB function, an application will still execute even if an expected DYLIB is not present. Weak linking enables developers to run an application on multiple macOS versions as new APIs are added.[24]

The following picture from *TaskExplorer* [25] software shows the analysis of the application where a number of DYLIBs being loaded without success by the application when executed:

[23] Dylib-hijacking-os-x
[24] MITRE Dylib Hijacking
[25] TaskExplorer



```
VPN_by_Google_One (pid: 61334)
/Applications/VPN_by_Google_One.app/Contents/MacOS/VPN_by_Google_One

? SystemConfiguration (deleted)
/System/Library/Frameworks/SystemConfiguration.framework/Versions/A/SystemConfiguration
? SystemExtensions (deleted)
/System/Library/Frameworks/SystemExtensions.framework/Versions/A/SystemExtensions
? SystemPolicy (deleted)
/System/Library/PrivateFrameworks/SystemPolicy.framework/Versions/A/SystemPolicy
? TCC (deleted)
/System/Library/PrivateFrameworks/TCC.framework/Versions/A/TCC
? TextInput (deleted)
/System/Library/PrivateFrameworks/TextInput.framework/Versions/A/TextInput
? TextureIO (deleted)
/System/Library/PrivateFrameworks/TextureIO.framework/Versions/A/TextureIO
? UIFoundation (deleted)
```

Figure 16: TaskExplorer

Recommendation

Set directory access controls to prevent file writes to the search paths for applications, both in the folders where applications are run from and the standard DYLIB folders.[26]

Monitor for dynamic libraries being loaded. Run path dependent libraries can include LC_LOAD_DYLIB, LC_LOAD_WEAK_DYLIB, and LC_RPATH. Other special keywords recognized by the macOS loader are @rpath, @loader_path, and @executable_path.[27] These loader instructions can be examined for individual binaries or frameworks using the otool -l command. Objective-See's Dylib Hijacking Scanner can be used to identify applications vulnerable to DYLIB hijacking

Location

- VPN_by_Google_One.app

[26] MITTRE Dylib Hijacking
[27] Run-Path Dependent Libraries

Low

Lack of Certificate Pinning

Overall Risk	Low	**Finding ID**	NCC-GOLE021-KE3
Impact	Medium	**Component**	macOS Application
Exploitability	Low	**Category**	Cryptography
		Status	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 laptop 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 years[28,29] , 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.

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 strong protection, as the attack surface is reduced to the specific CA pinned, and not the whole PKI infrastructure.

In addition, it is considered a good practice to pin more than one certificate, especially when pinning individual host certificates, to reduce the risk of issues associated to a specific certificate, such as an expired certificate.

Reproduction Steps

1. Install the Burp Proxy CA in the Keychain
2. Intercept the application's SSL traffic to pass through an interception proxy such as Burp Proxy
3. Verify that TLS traffic can be decrypted

[28] DigiNotar - Issuance of fraudulent certificates: https://en.wikipedia.org/wiki/DigiNotar#Issuance_of_fraudulent_certificates

[29] Comodo - Certificate hacking: https://en.wikipedia.org/wiki/Comodo_Group#Certificate_hacking

Location

- VPN_by_Google_One.app

Info

Binaries Contained Debug Information

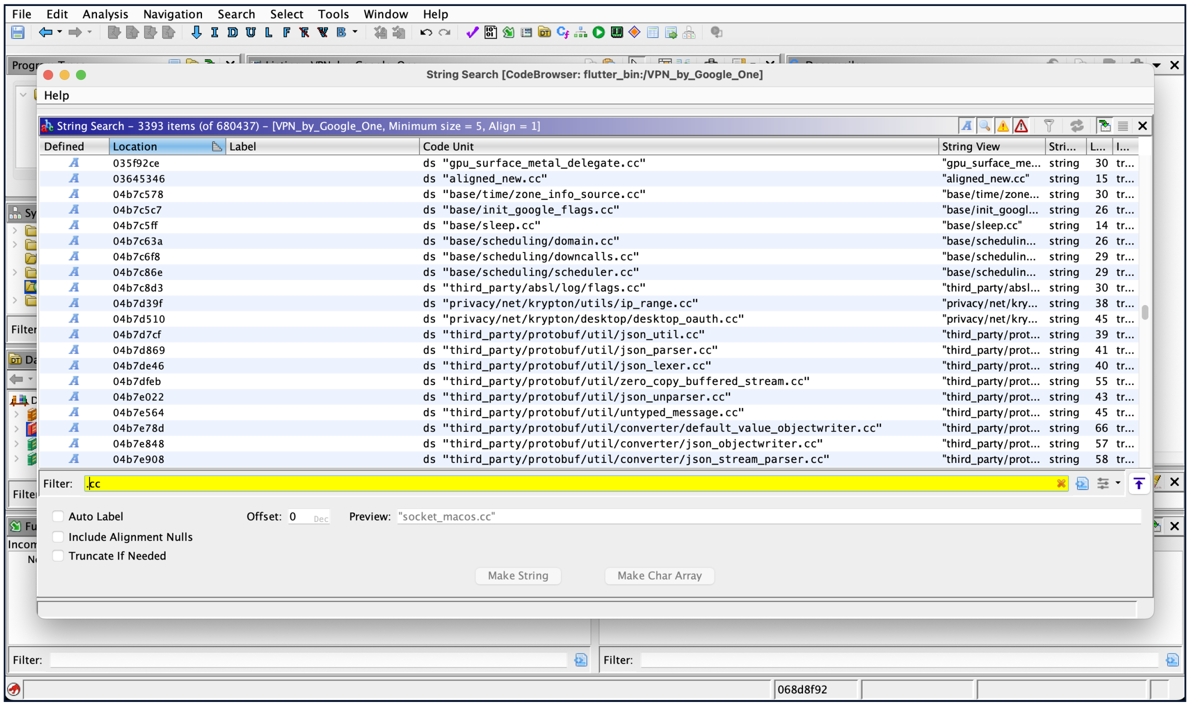
Overall Risk	Informational	**Finding ID**	NCC-GOLE021-B9L
Impact	Undetermined	**Component**	macOS Application
Exploitability	Undetermined	**Category**	Data Exposure
		Status	Not Fixed

Impact

While not a direct security issue, the presence of this information in the binaries makes reverse engineering efforts much simpler. Reverse engineering all or part of a binary is often a key step in producing a reliable exploit.

Description

Application binaries and DLLs contained debug information. Evidence of the issue can be seen in the screenshot below:



String Search [CodeBrowser: flutter_bin:/VPN_by_Google_One]

String Search – 3393 items (of 680437) – [VPN_by_Google_One, Minimum size = 5, Align = 1]

Defined	Location	Label	Code Unit	String View	Stri...	L...	I...
A	035f92ce		ds "gpu_surface_metal_delegate.cc"	"gpu_surface_me...	string	30	tr...
A	03645346		ds "aligned_new.cc"	"aligned_new.cc"	string	15	tr...
A	04b7c578		ds "base/time/zone_info_source.cc"	"base/time/zone...	string	30	tr...
A	04b7c5c7		ds "base/init_google_flags.cc"	"base/init_googl...	string	26	tr...
A	04b7c5ff		ds "base/sleep.cc"	"base/sleep.cc"	string	14	tr...
A	04b7c63a		ds "base/scheduling/domain.cc"	"base/schedulin...	string	26	tr...
A	04b7c6f8		ds "base/scheduling/downcalls.cc"	"base/schedulin...	string	29	tr...
A	04b7c86e		ds "base/scheduling/scheduler.cc"	"base/schedulin...	string	29	tr...
A	04b7c8d3		ds "third_party/absl/log/flags.cc"	"third_party/absl...	string	30	tr...
A	04b7d39f		ds "privacy/net/krypton/utils/ip_range.cc"	"privacy/net/kry...	string	38	tr...
A	04b7d510		ds "privacy/net/krypton/desktop/desktop_oauth.cc"	"privacy/net/kry...	string	45	tr...
A	04b7d7cf		ds "third_party/protobuf/util/json_util.cc"	"third_party/prot...	string	39	tr...
A	04b7d869		ds "third_party/protobuf/util/json_parser.cc"	"third_party/prot...	string	41	tr...
A	04b7de46		ds "third_party/protobuf/util/json_lexer.cc"	"third_party/prot...	string	40	tr...
A	04b7dfeb		ds "third_party/protobuf/util/zero_copy_buffered_stream.cc"	"third_party/prot...	string	55	tr...
A	04b7e022		ds "third_party/protobuf/util/json_unparser.cc"	"third_party/prot...	string	43	tr...
A	04b7e564		ds "third_party/protobuf/util/untyped_message.cc"	"third_party/prot...	string	45	tr...
A	04b7e78d		ds "third_party/protobuf/util/converter/default_value_objectwriter.cc"	"third_party/prot...	string	66	tr...
A	04b7e848		ds "third_party/protobuf/util/converter/json_objectwriter.cc"	"third_party/prot...	string	57	tr...
A	04b7e908		ds "third_party/protobuf/util/converter/json_stream_parser.cc"	"third_party/prot...	string	58	tr...

Filter: .cc

□ Auto Label
□ Include Alignment Nulls
□ Truncate If Needed

Offset: 0 Dec

Preview: "socket_macos.cc"

Filter:

Filter:

068d8f92

Figure 17: Binary contained debug information

Recommendation

Software build rules should be modified to remove debug symbols in production releases. Debug strings should be out of the production code. Post-processing of the now obfuscated logs can then be used to recover the original print statements for debugging purposes. Failure messages should be simplified so as not to reveal the detailed operation of the program. Avoid using function names and consider simple error numbers.

Reproduction Steps

1. Open VPN_by_Google_One binary using a Reverse Engineering framework
2. Analyse the VPN_by_Google_One binary
3. Search for strings like ".cc"

Location

- VPN_by_Google_One.app/Contents/MacOS/VPN_by_Google_One

Retest Results

2022-10-21 – Not Fixed

No changes were seen in the new binaries tested, all the information shown in the evidence above was still present.

Info

Application Binaries Not Obfuscated

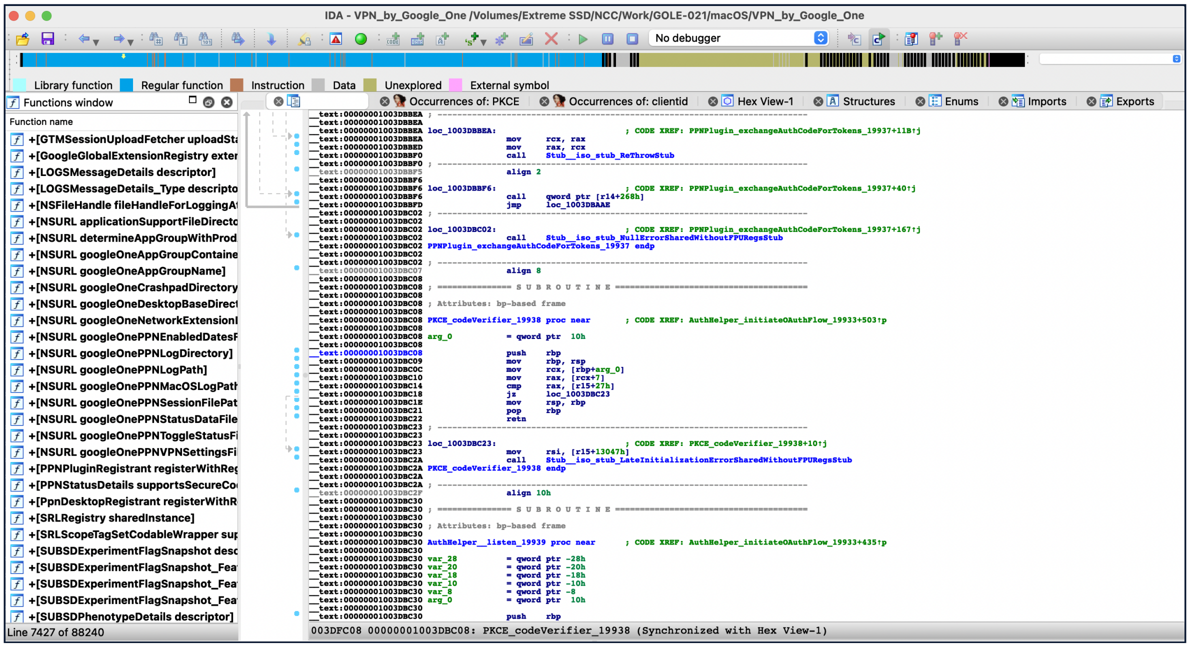
Overall Risk	Informational	**Finding ID**	NCC-GOLE021-DRA
Impact	Undetermined	**Component**	macOS Application
Exploitability	Undetermined	**Category**	Configuration
		Status	Partially Fixed

Impact

The lack of code obfuscation means that it is relatively easy to determine the structure and functionality of the application, and to analyse the binary for suitable locations to modify the application behavior.

Description

Application binaries and libraries were not obfuscated by using an executable binary obfuscator, as shown in the image below.



```
Function name
+[GTMSessionUploadFetcher uploadSt
+[GoogleGlobalExtensionRegistry exter
+[LOGSMessageDetails descriptor]
+[LOGSMessageDetails_Type descripto
+[NSFileHandle fileHandleForLoggingA
+[NSURL applicationSupportFileDirecto
+[NSURL determineAppGroupWithProd
+[NSURL googleOneAppGroupContaine
+[NSURL googleOneAppGroupName]
+[NSURL googleOneCrashpadDirectory
+[NSURL googleOneDesktopBaseDirect
+[NSURL googleOneNetworkExtension
+[NSURL googleOnePPNEnabledDatesF
+[NSURL googleOnePPNLogDirectory]
+[NSURL googleOnePPNLogPath]
+[NSURL googleOnePPNMacOSLogPath
+[NSURL googleOnePPNSessionFilePat
+[NSURL googleOnePPNStatusDataFile
+[NSURL googleOnePPNToggleStatusFi
+[NSURL googleOnePPNVPNSettingsFi
+[PPNPluginRegistrant registerWithReg
+[PPNStatusDetails supportsSecureCo
+[PpnDesktopRegistrant registerWithR
+[SRLRegistry sharedInstance]
+[SRLScopeTagSetCodableWrapper sup
+[SUBSDExperimentFlagSnapshot desc
+[SUBSDExperimentFlagSnapshot_Fea
+[SUBSDExperimentFlagSnapshot_Fea
+[SUBSDExperimentFlagSnapshot_Fea
+[SUBSDPhenotypeDetails descriptor]
Line 7427 of 88240
```

```
__text:00000001003DBBEA ; ---------------------------------------------------------------------------
__text:00000001003DBBEA
__text:00000001003DBBEA loc_1003DBBEA:                          ; CODE XREF: PPNPlugin_exchangeAuthCodeForTokens_19937+11B↑j
__text:00000001003DBBEA                 mov     rcx, rax
__text:00000001003DBBED                 mov     rax, rcx
__text:00000001003DBBF0                 call    Stub__iso_stub_ReThrowStub
__text:00000001003DBBF0 ; ---------------------------------------------------------------------------
__text:00000001003DBBF5                 align 2
__text:00000001003DBBF6
__text:00000001003DBBF6 loc_1003DBBF6:                          ; CODE XREF: PPNPlugin_exchangeAuthCodeForTokens_19937+40↑j
__text:00000001003DBBF6                 call    qword ptr [r14+268h]
__text:00000001003DBBFD                 jmp     loc_1003DBAAE
__text:00000001003DBC02 ; ---------------------------------------------------------------------------
__text:00000001003DBC02
__text:00000001003DBC02 loc_1003DBC02:                          ; CODE XREF: PPNPlugin_exchangeAuthCodeForTokens_19937+167↑j
__text:00000001003DBC02                 call    Stub__iso_stub_NullErrorSharedWithoutFPURegsStub
__text:00000001003DBC02 PPNPlugin_exchangeAuthCodeForTokens_19937 endp
__text:00000001003DBC02
__text:00000001003DBC02 ; ---------------------------------------------------------------------------
__text:00000001003DBC07                 align 8
__text:00000001003DBC08
__text:00000001003DBC08 ; =============== S U B R O U T I N E =======================================
__text:00000001003DBC08
__text:00000001003DBC08 ; Attributes: bp-based frame
__text:00000001003DBC08
__text:00000001003DBC08 PKCE_codeVerifier_19938 proc near       ; CODE XREF: AuthHelper_initiateOAuthFlow_19933+503↑p
__text:00000001003DBC08
__text:00000001003DBC08 arg_0           = qword ptr  10h
__text:00000001003DBC08
__text:00000001003DBC08                 push    rbp
__text:00000001003DBC09                 mov     rbp, rsp
__text:00000001003DBC0C                 mov     rcx, [rbp+arg_0]
__text:00000001003DBC10                 mov     rax, [rcx+7]
__text:00000001003DBC14                 cmp     rax, [r15+27h]
__text:00000001003DBC18                 jz      loc_1003DBC23
__text:00000001003DBC1E                 mov     rsp, rbp
__text:00000001003DBC21                 pop     rbp
__text:00000001003DBC22                 retn
__text:00000001003DBC23 ; ---------------------------------------------------------------------------
__text:00000001003DBC23
__text:00000001003DBC23 loc_1003DBC23:                          ; CODE XREF: PKCE_codeVerifier_19938+10↑j
__text:00000001003DBC23                 mov     rsi, [r15+13047h]
__text:00000001003DBC2A                 call    Stub__iso_stub_LateInitializationErrorSharedWithoutFPURegsStub
__text:00000001003DBC2A PKCE_codeVerifier_19938 endp
__text:00000001003DBC2A
__text:00000001003DBC2A ; ---------------------------------------------------------------------------
__text:00000001003DBC2F                 align 10h
__text:00000001003DBC30
__text:00000001003DBC30 ; =============== S U B R O U T I N E =======================================
__text:00000001003DBC30
__text:00000001003DBC30 ; Attributes: bp-based frame
__text:00000001003DBC30
__text:00000001003DBC30 AuthHelper__listen_19939 proc near      ; CODE XREF: AuthHelper_initiateOAuthFlow_19933+435↑p
__text:00000001003DBC30
__text:00000001003DBC30 var_28          = qword ptr -28h
__text:00000001003DBC30 var_20          = qword ptr -20h
__text:00000001003DBC30 var_18          = qword ptr -18h
__text:00000001003DBC30 var_10          = qword ptr -10h
__text:00000001003DBC30 var_8           = qword ptr -8
__text:00000001003DBC30 arg_0           = qword ptr  10h
__text:00000001003DBC30
__text:00000001003DBC30                 push    rbp
003DFC08 00000001003DBC08: PKCE_codeVerifier_19938 (Synchronized with Hex View-1)
```

Figure 18: No obfuscation

Recommendation

Consider using a binary or source-level obfuscation system to obfuscate the application. Additionally, commercially available binary obfuscation systems may also provide runtime protection of memory contents.

However, it is worth noting that all binary obfuscation can be defeated, and only serves to deter casual attackers and slow down skilled and motivated attackers. Additionally, due to the way binary executable obfuscation works, using it may remove platform-level mitigations and thus weaken the authentication system's security posture in other ways. It is worth noting that code signing will typically be mutually exclusive with binary obfuscation, requiring a business decision to be made as to whether to make the authentication system more difficult to reverse engineer or more difficult to modify.

Location

- VPN_by_Google_One.app/Contents/MacOS/VPN_by_Google_One

Retest Results

2022-10-21 – Partially Fixed

The VPN by Google One binary shown that some of the function names were obfuscated, but a number of them were still visible:

```
+[NSFileHandle(GTMFileHandleLogWriter)_fileHandleForLoggingAtPath:mode:]
+[NSURL(AppGroupContainer)_applicationSupportFileDirectory]
+[NSURL(AppGroupContainer)_googleOneAppGroupContainer]
+[NSURL(AppGroupContainer)_googleOneAppGroupName]
+[NSURL(AppGroupContainer)_googleOneCrashpadDirectoryPath]
+[NSURL(AppGroupContainer)_googleOneNetworkExtensionDataDirectory]
+[NSURL(AppGroupContainer)_googleOnePPNEnabledDatesFilePath]
+[NSURL(AppGroupContainer)_googleOnePPNLogDirectory]
+[NSURL(AppGroupContainer)_googleOnePPNLogPath]
+[NSURL(AppGroupContainer)_googleOnePPNMacOSLogPath]
+[NSURL(AppGroupContainer)_googleOnePPNSessionFilePath]
+[NSURL(AppGroupContainer)_googleOnePPNStatusDataFilePath]
+[NSURL(AppGroupContainer)_googleOnePPNToggleStatusFilePath]
+[NSURL(AppGroupContainer)_googleOnePPNVPNSettingsFilePath]
```

9 Finding Details – vpn-libraries

Low

Use of Deprecated and Internal Functions

Overall Risk	Low	**Finding ID**	NCC-GOLE021-AJK
Impact	Medium	**Component**	vpn-libraries
Exploitability	Low	**Category**	Configuration
		Status	Reported

Impact

Use of internal library functions could result in security checks being modified or rendered ineffective if the library is modified over time.

Description

The `SetBlindingPublicKey` function uses Tink functions such as `ValidateRsaModulusSize`, `ValidateRsaPublicExponent` and `BoringSslRsaFromRsaPublicKey` to validate and set public key properties. While the validation is correct, these functions are intended to be internal to the Tink library and are not for external consumption. As an example `ValidateRsaPublicExponent` contains the following comments in subtle_util_boringssl.h :

```
ABSL_DEPRECATED("Use of this function is dicouraged outside Tink.")
static inline crypto::tink::util::Status ValidateRsaPublicExponent(
    absl::string_view exponent) {
  return internal::ValidateRsaPublicExponent(exponent);
}
```

Using internal functions should be avoided since such functions may disappear at any time, leading to application crashes or public key validation being rendered ineffective.

Recommendation

These Tink functions should not be called directly and it is recommended to transition to publicly maintained validation or, include these checks in the VPN library code directly.

Location

`/VPN Library Code/krypton/crypto/session_crypto.cc`

10 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.

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.
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	

Rating	Description
	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.

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.

11 Client Provided Documentation

The following design documentation was provided to NCC Group by Google sharing documents with a Partner Google account set for this project:

- Debug logs details and monitoring procedures: `Debug and Production_Monitoring`
- Guide to instance deployment and rollback in production and locally: `Deployment and Rollback`
- Krypton and Xenon in depth architecture: `Krypton and Xenon Details`
- Migration plan to a new authentication infrastructure: `Phosphor Migration Plan.docx`
- Phosphor in depth architecture: `Phosphor_ PPN ServiceType mux.docx`
- Design definition for the authentication in the Borg internal network: `PPN Auth on Borg_ Detailed Design.docx`
- Authentication design and integration with Google One services: `PPN Authentication and G1 Integration.docx`
- Key rotation procedure in depth: `PPN Key Rotation - Markdown Export.docx`
- Attestation Service definition and expectations: `PPN Mithril.docx`
- Resources for newcomers and general architecture: `PPN Onboarding Resources and Architecture.docx`
- Key rotation overview and glossary: `PPN Signing Key Rotation, Glossary, DNS and APN Sections.docx`
- Local setup for Windows and MacOS tests: `PPN Windows and MacOS Setup and Marconi Process.docx`
- Proposal to implement a new authentication method: `Proposal_ PPN Borg Authentication.docx`
- Public Google PPN VPN whitepaper: `white_paper_4f995ab5d7c7edc3d3f14f2e0593f790.pdf`
- How is enforced the quota during PPN authentication to prevent service abuse: `PPN per-user Quota.docx`

The following documents were used as working documents for writing questions and answers:

- `NCC Questions-27-7-2022`
- `NCC Questions-28-7-2022`
- `NCC Questions-29-7-2022`
- `NCC Questions-2-8-2022`