Hypori Halo Client (Android) 4.3 Security Target

Version 1.0
February 15, 2024

Prepared for:
Hypori, Inc.
1801 Robert Fulton Drive, Suite 440
Reston, VA 20191

Prepared by:
Leidos Inc.
Common Criteria Testing Laboratory
6841 Benjamin Franklin Drive, Columbia, Maryland 21046

Copyright

LIST OF TABLES

1. Security Target Introduction

This section identifies the Target of Evaluation (TOE) along with identification of the Security Target (ST) itself. The section includes documentation organization, ST conformance claims, and ST conventions.

The TOE is the Hypori Client (Android) 4.3 component of the Hypori Platform provided by Hypori, Inc.

The Security Target contains the following additional sections:

- Security Target Introduction (Section 1)
- TOE Description (Section 2)
- Security Problem Definition (Section 3)
- Security Objectives (Section 4)
- IT Security Requirements (Section 5)
- TOE Summary Specification (Section 6)
- Protection Profile Claims (Section 7)
- Rationale (Section 8).
- Appendix: Android APIs (Section 9).
- Appendix: Java Library APIs (Section 10)

1.1 Security Target, TOE and CC Identification

ST Title – Hypori Halo Client (Android) 4.3 Security Target

ST Version – Version 1.0

ST Date – February 15, 2024

TOE Identification – Hypori Halo Client (Android) 4.3

TOE Developer – Hypori, Inc.

Evaluation Sponsor – Hypori, Inc.

CC Identification – *Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 5, April 2017*

1.2 Conformance Claims

This TOE is conformant to the following CC specifications:

This ST is conformant to the *Protection Profile for Application Software*, Version 1.4, 2021-10-07 [PP_APP_v1.4].

The following NIAP Technical Decisions apply to the security target or the evaluation assurance activities.

- TD0780: FIA_X509_EXT.1 Test 4 Clarification
- TD0756: Update for platform-provided full disk encryption
- TD0747: Configuration Storage Option for Android
- TD0743: FTP_DIT_EXT.1.1 Selection exclusivity
- TD0719: ECD for PP APP V1.3 and 1.4
- TD0717: Format changes for PP_APP_V1.4
- TD0664: Testing activity for FPT_TUD_EXT.2.2

The following NIAP Technical Decisions are list on the NIAP website, but are not applicable to this evaluation:

- TD0798: Static Memory Mapping Exceptions
    - The Security Target does not include any list of explicit exceptions in FPT_AEX_EXT.1.1.
- TD0736: Number of elements for iterations of FCS_HTTPS_EXT.1
    - The Security Target does not include FCS_HTTPS_EXT.1/Server.
- TD0650: Conformance claim sections updated to allow for MOD_VPNC_V2.3 and 2.4
    - The Security Target does not claim conformance to the PP-Module for VPN Clients.
- TD0628: Addition of Container Image to Package Format
    - The TOE is not a container image.

Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, Version 3.1, Revision 5, April 2017.

- Part 2 Extended

Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1 Revision 5, April 2017.

- Part 3 Extended

1.3 Conventions

The following conventions have been applied in this document:

- Security Functional Requirements – Part 2 of the CC defines the approved set of operations that may be applied to functional requirements: iteration, assignment, selection, and refinement.
    - Iteration: allows a component to be used more than once with varying operations. In the ST, iteration is indicated by a slash followed by a descriptor for the purpose of the iteration. For example, FCS_CKM.1/AK indicates that the FCS_CKM.1 requirement applies specifically to Asymmetric Key Generation functionality.
    - Assignment: allows the specification of an identified parameter. Assignments are indicated using bold and are surrounded by brackets (e.g., [**assignment**]). Note that an assignment within a selection would be identified in italics and with embedded bold brackets (e.g., [***[selected-assignment]***]).
    - Selection: allows the specification of one or more elements from a list. Selections are indicated using bold italics and are surrounded by brackets (e.g., [***selection***]).
    - Refinement: allows the addition of details. Refinements are indicated using bold, for additions, and strike-through, for deletions (e.g., "… **all** objects …" or "… ~~some~~ **big** things …"). Note that 'cases' that are not applicable in a given SFR have simply been removed without any explicit identification.
- Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions.

1.3.1 Terminology

[PP_APP_v1.4] provides definitions for terms specific to the application software technology as well as general Common Criteria terms. The technology-specific terms are:

- Address Space Layout Randomization
- Application
- Application Programming Interface
- Credential
- Data Execution Prevention

- Developer
- Mobile Code
- Operating System
- Personally Identifiable Information
- Platform
- Sensitive Data
- Stack Cookie
- Vendor

Terms from the Common Criteria are:

- Common Criteria
- Common Evaluation Methodology
- Protection Profile
- Security Target
- Target of Evaluation
- TOE Security Functionality
- TOE Summary Specification
- Security Functional Requirement
- Security Assurance Requirement

This ST does not include additional technology-specific terminology.

1.3.2 Abbreviations

This section identifies abbreviations and acronyms used in this ST.

API	Application Programming Interface
App	Software application
ASLR	Address Space Layout Randomization
CC	Common Criteria
CEM	Common Evaluation Methodology
FCM	Google's Firebase Cloud Messaging
gid	Group Identifier
MDM	Mobile Device Management
OS	Operating System
PII	Personally Identifiable Information
PP	Protection Profile
PP_APP_v1.4	Protection Profile for Application Software
SAR	Security assurance requirement
SFR	Security functional requirement
ST	Security Target
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSS	TOE Summary Specification
uid	User Identifier

2. TOE Description

After a brief overview of the Hypori Halo Client (Android) product, this section describes its Hypori Halo Client (Android) component, which is the Target of Evaluation (TOE). The description covers TOE architecture, logical boundaries, and physical boundaries.

2.1 Product Overview

In the Hypori Halo Client (Android) platform, end users running a Hypori Halo Client (Android) on their mobile device access a virtual Android device running on a server in the cloud. The virtual device on the server contains the operating system, the data, and the applications, using TLS 1.2 encryption to communicate securely with the Hypori Client (Android). The Hypori Android application provides secure access to the remote Android virtual device and brokers access between the mobile device and the applications executing in the virtual device on a Hypori server. The client applications on the Hypori server are indifferent to the version of Android executing on the physical device.

The following diagram illustrates the user data connection for the TOE.

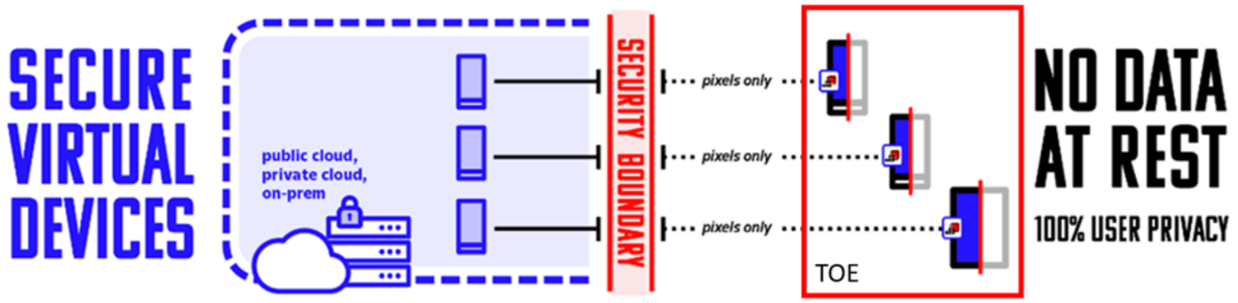

Figure 1 Hypori Halo Client (Android) User Data Flow

The user's physical device is a "window" to their virtualized smartphone residing in the cloud or on-premises. The Hypori Halo app captures touch and sensor data from any Android device. Encrypted pixels are transmitted to and from the physical device to access the enterprise applications in the cloud.

Hypori Halo delivers secure access to enterprise apps and data via a separate, secure virtual device from a smartphone or tablet. It uses cloud-based, zero-trust architecture, guarantees no data on the device, and 100% separation of personal and enterprise data.

The platform device which hosts the Hypori Halo application is not included in the TOE boundary.

The following diagram shows the Hypori system, including its components and networks. Unlike many software solutions, some of the Hypori servers are installed on virtual servers while others are installed on physical servers.

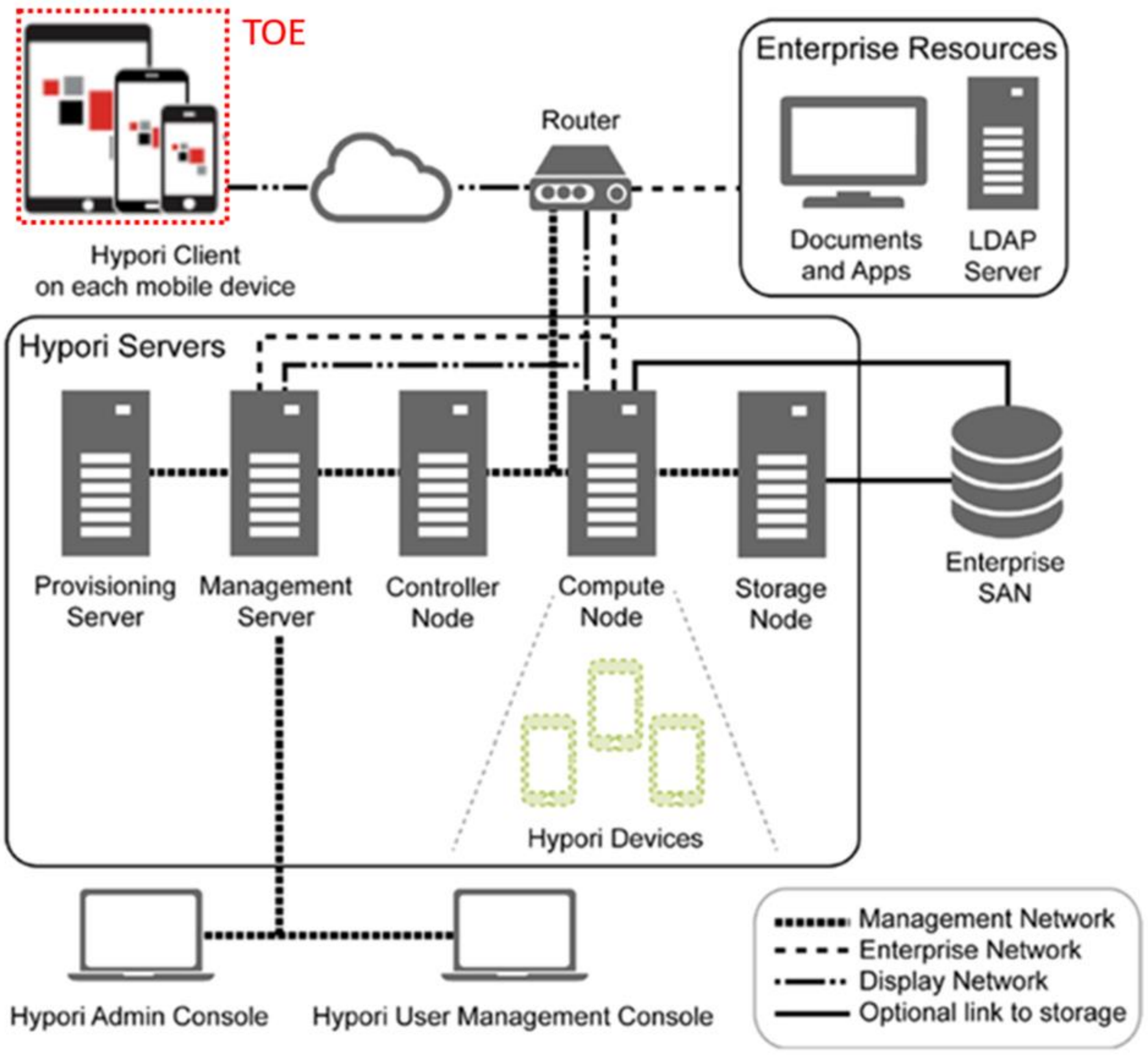

Figure 2 Hypori Solution

The Hypori solution includes the following components:

- **Hypori Halo Client:** This is an Android- application that installs on the end user's mobile device and communicates with the Hypori Virtual Device on the server through secure encrypted protocols. The platform device is not included in the TOE boundary.
- **Hypori Virtual Device:** This is an Android-based virtualized mobile device executing on a server in the cloud.
- **Hypori Servers:** This is the cloud server cluster that hosts the Hypori Virtual Devices.
- **Hypori Admin Console:** This is a browser-based administration user interface that is used to manage the Hypori system.
- **Hypori User Management Console**: A web application to manage users within a designated Hypori Halo environment.

The Hypori Virtual Device, Servers, User Management Console, and Admin Console are not included in the evaluation.

2.2 TOE Overview

The TOE is the Android-based Hypori Halo Client software application. The following diagram shows how the TOE interacts with a Hypori Device running applications on a Hypori Server. The Hypori Halo Client is an application that communicates only with a Hypori Virtual Device on a Hypori Server and not with other servers or applications.

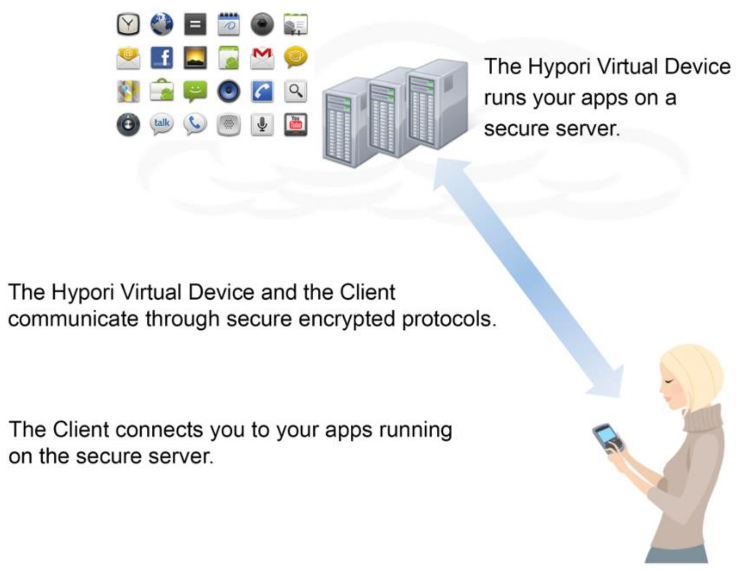
The Hypori Virtual Device runs your apps on a secure server.
The Hypori Virtual Device and the Client communicate through secure encrypted protocols.
The Client connects you to your apps running on the secure server.

Figure 3 Hypori Halo Client Communication with a Hypori Virtual Device on a Hypori Server

2.3 TOE Architecture

This section describes the TOE architecture including physical and logical boundaries. Figure 4 shows the relationship of the TOE to its operational environment along with the TOE boundary. The security functional requirements identify the libraries included in the application package.

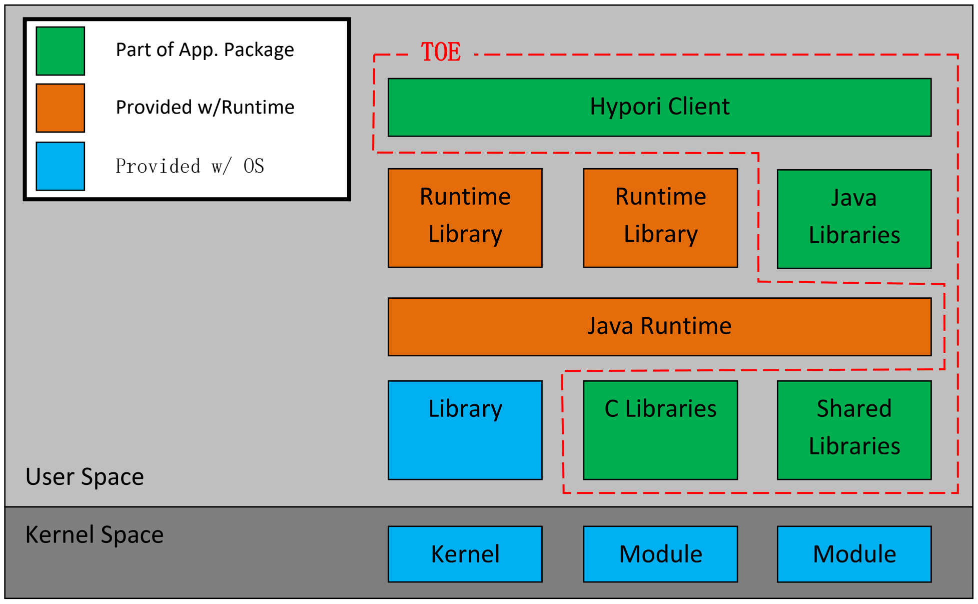

Figure 4 TOE Boundary for Android Devices

2.3.1 Physical Boundaries

The TOE consists of a Hypori Halo Client software application available in the Hypori Halo Client installation package from the Google Play Store. The Hypori Halo Client is an Android application that only communicates with the Hypori Virtual Device on the Hypori server. The Hypori server, applications running on the Hypori server, the hardware mobile device, and any functions not specified in this security target are outside the scope of the TOE.

2.3.1.1 Software Requirements

The TOE was evaluated on Android releases 12 and 13.

2.3.1.2 Hardware Requirements

The TOE imposes no hardware requirements beyond the Android operating system requirements.

2.3.2 Logical Boundaries

This section summarizes the security functions provided by the TOE:

- Cryptographic support
- User data protection
- Identification and Authentication
- Security management
- Privacy
- Protection of the TSF
- Trusted path/channels

2.3.2.1 Cryptographic support

The TOE establishes secure communication with the Hypori Virtual Device on the server using TLS and cryptographic services provided by the platform. TOE stores credentials and certificates for mutual authentication in the platform's key chain.

2.3.2.2 User data protection

The TOE informs a user of hardware and software resources the TOE accesses.

The user initiates a secure network connection to the Hypori Virtual Device on the server using the TOE. In general, sensitive data resides on the Hypori server and not the Hypori Client, although the client does store credentials as per section 2.3.2.1.

2.3.2.3 Identification and authentication

The TOE supports X.509 certificate validation as part of establishing TLS connections. The TOE relies on platform-provided functionality to support certificate validity checking methods, including the checking of certificate revocation status using OCSP. If the validity status of a certificate cannot be determined, the certificate will not be accepted.

2.3.2.4 Security management

Security management consists of setting Hypori Client configuration options and applying configuration policies from the Hypori Server. The TOE stores the configuration settings and policies encrypted using cryptographic services provided by the platform.

2.3.2.5 Privacy

The TOE does not transmit PII over a network.

2.3.2.6 Protection of the TSF

The TOE uses security features and APIs that the platform provides. The TOE leverages package management for secure installation and updates. The TOE package includes only those third-party libraries necessary for its intended operation.

2.3.2.7 Trusted path/channels

The TOE invokes the platform-provided functionality to encrypt all transmitted data using TLS 1.2 for all communication with the Hypori Virtual Device on the Hypori server.

2.4 TOE Documentation

Hypori provides the following product documentation in support of the installation and secure use of the TOE.

- Hypori Halo Client User Guide Common Criteria Configuration and Operation Version 4.3
- Hypori Halo Administrator Guide, Version 1.18

3. Security Problem Definition

This security target includes by reference the Security Problem Definition from the [PP_APP_v1.4]. The Security Problem Definition consists of threats that a conformant TOE is expected to address and assumptions about the operational environment of the TOE.

In general, the [PP_APP_v1.4] has presented a Security Problem Definition appropriate for application software that runs on mobile devices, as well as on desktop and server platforms. The Hypori Halo Client is an Android application running on a mobile device. As such, the [PP_APP_v1.4] Security Problem Definition applies to the TOE.

4. Security Objectives

Like the Security Problem Definition, this security target includes by reference the Security Objectives from the [PP_APP_v1.4]. The [PP_APP_v1.4] security objectives for the operational environment are reproduced below, since these objectives characterize technical and procedural measures each consumer must implement in their operational environment.

In general, the [PP_APP_v1.4] has presented a Security Objectives statement appropriate for application software that runs on mobile devices, as well as on desktop and server platforms. Consequently, the [PP_APP_v1.4] security objectives are suitable for the Hypori Halo Client (Android) TOE.

4.1 Security Objectives for the Operational Environment

OE.PLATFORM	The TOE relies upon a trustworthy computing platform for its execution. This includes the underlying operating system and any discrete execution environment provided to the TOE.
OE.PROPER_USER	The user of the application software is not willfully negligent or hostile, and uses the software within compliance of the applied enterprise security policy.
OE.PROPER_ADMIN	The administrator of the application software is not careless, willfully negligent or hostile, and administers the software within compliance of the applied enterprise security policy.

5. IT Security Requirements

This section defines the Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) that serve to represent the security functional claims for the Target of Evaluation (TOE) and to scope the evaluation effort.

The security functional requirements have all been drawn from: *Protection Profile for Application Software*, Version 1.4, 2021-10-07 [PP_APP_v1.4]. As a result, any selection, assignment, or refinement operations already performed by that PP on the claimed SFRs are not identified here (i.e., they are not formatted in accordance with the conventions specified in section 1.3 of this ST). Formatting conventions are only applied on SFR text that was chosen at the ST author's discretion.

The security assurance requirements are the set of SARs specified in [PP_APP_v1.4].

5.1 Extended Requirements

All of the extended requirements in this ST have been drawn from the [PP_APP_v1.4]. The [PP_APP_v1.4] defines the following extended security requirements. Since these security requirements are not redefined in this ST, readers should consult [PP_APP_v1.4] for more information in regard to these CC extensions.

- FCS_CKM_EXT.1 Cryptographic Key Generation Services
- FCS_RBG_EXT.1 Random Bit Generation Services
- FCS_STO_EXT.1 Storage of Credentials
- FDP_DAR_EXT.1 Encryption Of Sensitive Application Data
- FDP_NET_EXT.1 Network Communications
- FDP_DEC_EXT.1 Access to Platform Resources
- FIA_X509_EXT.1 X.509 Certificate Validation
- FIA_X509_EXT.2 X.509 Certificate Authentication
- FMT_MEC_EXT.1 Supported Configuration Mechanism
- FMT_CFG_EXT.1 Secure by Default Configuration
- FPR_ANO_EXT.1 User Consent for Transmission of Personally Identifiable Information
- FPT_AEX_EXT.1 Anti-Exploitation Capabilities
- FPT_API_EXT.1 Use of Supported Services and APIs
- FPT_IDV_EXT.1 Software Identification and Versions
- FPT_LIB_EXT.1 Use of Third Party Libraries
- FPT_TUD_EXT.1 Integrity for Installation and Update
- FPT_TUD_EXT.2 Integrity for Installation and Update
- FTP_DIT_EXT.1 Protection of Data in Transit
- ALC_TSU_EXT.1 Timely Security Updates

5.2 TOE Security Functional Requirements

The following table identifies the SFRs that are satisfied by the Hypori Client TOE.

Table 1 TOE Security Functional Components

Requirement Class	Requirement Component
	FCS_CKM_EXT.1 Cryptographic Key Generation Services

Requirement Class	Requirement Component
FCS: Cryptographic support	FCS_CKM.1/AK Cryptographic Asymmetric Key Generation
	FCS_CKM.2 Cryptographic Key Establishment
	FCS_RBG_EXT.1 Random Bit Generation Services
	FCS_STO_EXT.1 Storage of Credentials
FDP: User data protection	FDP_DAR_EXT.1 Encryption of Sensitive Application Data
	FDP_DEC_EXT.1 Access to Platform Resources
	FDP_NET_EXT.1 Network Communications
FIA: Identification and authentication	FIA_X509_EXT.1 X.509 Certificate Validation
	FIA_X509_EXT.2 X.509 Certificate Authentication
FMT: Security management	FMT_CFG_EXT.1 Secure by Default Configuration
	FMT_MEC_EXT.1 Supported Configuration Mechanism
	FMT_SMF.1 Specification of Management Functions
FPR: Privacy	FPR_ANO_EXT.1 User Consent for Transmission of Personally Identifiable Information
FPT: Protection of the TSF	FPT_AEX_EXT.1 Anti-Exploitation Capabilities
	FPT_API_EXT.1 Use of Supported Services and APIs
	FPT_IDV_EXT.1 Software Identification and Versions
	FPT_LIB_EXT.1 Use of Third Party Libraries
	FPT_TUD_EXT.1 Integrity for Installation and Update
	FPT_TUD_EXT.2 Integrity for Installation and Update
FTP: Trusted path/channels	FTP_DIT_EXT.1 Protection of Data in Transit

5.2.1 Cryptographic Support (FCS)

5.2.1.1 Cryptographic Key Generation Services (FCS_CKM_EXT.1)

FCS_CKM_EXT.1.1[1] The application shall [***invoke platform-provided functionality for asymmetric key generation***].

5.2.1.2 Cryptographic Asymmetric Key Generation (FCS_CKM.1/AK)

FCS_CKM.1.1/AK[2] The application shall [

- ***invoke platform-provided functionality***

] to generate asymmetric cryptographic keys in accordance with a specified cryptographic key generation algorithm [

- ***[RSA schemes] using cryptographic key sizes of [2048-bit or greater] that meet the following: [FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3"],***
- ***[ECC schemes] using ["NIST curves" P-384 and [P-256, P-521]]that meet the following: [FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4],***

].

[1] Modified per TD0717.

[2] Modified per TD0717.

5.2.1.3 Cryptographic Key Establishment (FCS_CKM.2)

FCS_CKM.2.1 The application shall [***invoke platform-provided functionality***] to perform cryptographic key establishment in accordance with a specified cryptographic key establishment method:

[

- ***[RSA-based key establishment schemes] that meets the following: [NIST Special Publication 800-56B, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography"] ,***
- ***[Elliptic curve-based key establishment schemes] that meets the following: [NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"] ,***

].

5.2.1.4 Random Bit Generation Services (FCS_RBG_EXT.1)

FCS_RBG_EXT.1.1 The application shall [***use no DRBG functionality***] for its cryptographic operations.

5.2.1.5 Storage of Credentials (FCS_STO_EXT.1)

FCS_STO_EXT.1.1 The application shall [***invoke the functionality provided by the platform to securely store [user TLS client private key]***] to non-volatile memory.

5.2.2 User Data Protection (FDP)

5.2.2.1 Encryption of Sensitive Application Data (FDP_DAR_EXT.1)

FDP_DAR_EXT.1.1 The application shall [***protect sensitive data in accordance with FCS_STO_EXT.1***] in nonvolatile memory.

5.2.2.2 Access to Platform Resources (FDP_DEC_EXT.1)

FDP_DEC_EXT.1.1 The application shall restrict its access to [

- ***network connectivity,***
- ***camera,***
- ***microphone,***
  location services,
- ***Bluetooth,***
- ***[Fingerprint scanner]***].

FDP_DEC_EXT.1.2 The application shall restrict its access to [

- ***no sensitive information repositories***

].

5.2.2.3 Network Communications (FDP_NET_EXT.1)

FDP_NET_EXT.1.1 The application shall restrict network communication to [

- ***user-initiated communication for [connecting to the Virtual Device on the Hypori server],***
- ***[polling the Hypori server for notifications]***

].

5.2.3 Identification and authentication (FIA)

5.2.3.1 X.509 Certificate Validation (FIA_X509_EXT.1)

FIA_X509_EXT.1.1 The application shall [***invoke platform-provided functionality***] to validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a trusted CA certificate.
- The application shall validate a certificate path by ensuring the presence of the basicConstraints extension, that the CA flag is set to TRUE for all CA certificates. and that any path constraints are met
- The application shall validate that any CA certificate includes caSigning purpose in the key usage field
- The application shall validate the revocation status of the certificate using [***OCSP as specified in RFC 6960***].
- The application shall validate the extendedKeyUsage (EKU) field according to the following rules:
    - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
    - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the EKU field.
    - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the EKU field.
    - S/MIME certificates presented for email encryption and signature shall have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the EKU field.[3]
    - OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the EKU field.
    - Server certificates presented for EST shall have the CMC Registration Authority (RA) purpose (id-kp-cmcRA with OID 1.3.6.1.5.5.7.3.28) in the EKU field.[4]

FIA_X509_EXT.1.2 The application shall treat a certificate as a CA certificate only if the basicConstraints extension is present and the CA flag is set to TRUE.

5.2.3.2 X.509 Certificate Authentication (FIA_X509_EXT.2)

FIA_X509_EXT.2.1 The application shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [***TLS***].

FIA_X509_EXT.2.2 When the application cannot establish a connection to determine the validity of a certificate, the application shall [***not accept the certificate***].

5.2.4 Security Management (FMT)

5.2.4.1 Secure by Default Configuration (FMT_CFG_EXT.1)

FMT_CFG_EXT.1.1 The application shall provide only enough functionality to set new credentials when configured with default credentials or no credentials.

FMT_CFG_EXT.1.2 The application shall be configured by default with file permissions which protect the application's binaries and data files from modification by normal unprivileged users.

[3] The Hypori Client does not check extended key usage for Email Protection, since the Hypori Client does not perform email encryption or email signature verification.

[4] The Hypori Client does not check extended key usage for CMC Registration Authority, since the Hypori Client does not perform Enrollment over Secure Transport.

5.2.4.2 Supported Configuration Mechanism (FMT_MEC_EXT.1)

FMT_MEC_EXT.1.1 The application shall

- [*invoke the mechanisms recommended by the platform vendor for storing and setting configuration options and Hypori Halo Client policies from the Hypori Server*].

5.2.4.3 Specification of Management Functions (FMT_SMF.1)

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions [***[***

- ***setting configuration options***
- ***applying configuration policies from the Hypori Server]***

].

5.2.5 Privacy (FPR)

5.2.5.1 User Consent for Transmission of Personally Identifiable Information (FPR_ANO_EXT.1)

FPR_ANO_EXT.1.1 The application shall [***not transmit PII over a network***].

5.2.6 Protection of the TSF (FPT)

5.2.6.1 Anti-Exploitation Capabilities (FPT_AEX_EXT.1)

FPT_AEX_EXT.1.1 The application shall not request to map memory at an explicit address except for [**no exceptions**].

FPT_AEX_EXT.1.2 The application shall [***not allocate any memory region with both write and execute permissions***].

FPT_AEX_EXT.1.3 The application shall be compatible with security features provided by the platform vendor.

FPT_AEX_EXT.1.4 The application shall not write user-modifiable files to directories that contain executable files unless explicitly directed by the user to do so.

FPT_AEX_EXT.1.5 The application shall be built with stack-based buffer overflow protection enabled.

5.2.6.2 Use of Supported Services and APIs (FPT_API_EXT.1)

FPT_API_EXT.1.1 The application shall use only documented platform APIs.

5.2.6.3 Software Identification and Versions (FPT_IDV_EXT.1)

FPT_IDV_EXT.1.1 The application shall be versioned with [***[Android application version identifier, internal build information]***].

5.2.6.4 Use of Third Party Libraries (FPT_LIB_EXT.1)

FPT_LIB_EXT.1.1 The application shall be packaged with only [

- **Opus Audio Codec v1.1**
- **Protobuf v3.21.1**
- **Zxing core 3.3.0**
- **Yubikit v1.0.0**
- **AppAuth 0.9.1**
- **Moshi 1.13.0**
- **BouncyCastle 1.70**

- **Hasher 1.2**
- **Kotlin standard library 1.8.10**
- **kotlin-reflect 1.8.10**
- **kotlinx-coroutines 1.6.4**
- **Dagger Hilt v2.45**

].

5.2.6.5 Integrity for Installation and Update (FPT_TUD_EXT.1)

FPT_TUD_EXT.1.1 The application shall [***leverage the platform***] to check for updates and patches to the application software.

FPT_TUD_EXT.1.2 The application shall [***provide the ability***] to query the current version of the application software.

FPT_TUD_EXT.1.3 The application shall not download, modify, replace or update its own binary code.

FPT_TUD_EXT.1.4 Application updates shall be digitally signed such that the application platform can cryptographically verify them prior to installation.

FPT_TUD_EXT.1.5 The application is distributed [***as an additional software package to the platform OS***].

5.2.6.6 Integrity for Installation and Update (FPT_TUD_EXT.2)

FPT_TUD_EXT.2.1 The application shall be distributed using the format of the platform-supported package manager.

FPT_TUD_EXT.2.2 The application shall be packaged such that its removal results in the deletion of all traces of the application, with the exception of configuration settings, output files, and audit/log events.

FPT_TUD_EXT.2.3 The application installation package shall be digitally signed such that its platform can cryptographically verify them prior to installation.

5.2.7 Trusted path/channels (FTP)

5.2.7.1 Protection of Data in Transit (FTP_DIT_EXT.1)

FTP_DIT_EXT.1.1[5] The application shall [

- i***nvoke platform-provided functionality to encrypt all transmitted data with [TLS] for [communication with the virtual Hypori Device running applications on a Hypori Server]***] between itself and another trusted IT product.

5.3 TOE Security Assurance Requirements

The security assurance requirements in Table 2 are included in this ST by reference from the [PP_APP_v1.4].

Table 2 Assurance Components

Requirement Class	Requirement Component
ADV: Development	ADV_FSP.1 Basic functional specification
AGD: Guidance documents	AGD_OPE.1: Operational user guidance
	AGD_PRE.1: Preparative procedures
ALC: Life-cycle support	ALC_CMC.1 Labelling of the TOE

[5] Modified per TD0743.

	ALC_CMS.1 TOE CM coverage
	ALC_TSU_EXT.1 Timely Security Updates
ATE: Tests	ATE_IND.1 Independent testing - conformance
AVA: Vulnerability assessment	AVA_VAN.1 Vulnerability survey

These assurance requirements imply the following requirements from CC class ASE: Security Target Evaluation.

- ASE_CCL.1 Conformance claims
- ASE_ECD.1 Extended components definition
- ASE_INT.1 ST introduction
- ASE_OBJ.1 Security objectives for the operational environment
- ASE_REQ.1 Stated security requirements
- ASE_TSS.1 TOE summary specification

Consequently, the assurance activities specified in [PP_APP_v1.4] apply to the TOE evaluation.

6. TOE Summary Specification

This chapter describes the security functions:

- Cryptographic support
- User data protection
- Identification and authentication
- Security management
- Privacy
- Protection of the TSF
- Trusted path/channels

Additionally, this chapter describes the function "Timely Security Updates" that maps to the ALC_TSU_EXT.1 assurance component requirement.

6.1 Cryptographic support

The Hypori Client makes use of the platform for cryptographic services. The Hypori Halo Client uses platform TLS services for secure communication with the Hypori Virtual Device on the Hypori server, including mutual authentication. The client uses TLS client certificates and keys along with a CA certificate for the server. The user stores these certificates in the platform's key store during installation. The user need not install a CA certificate when the CA is a platform trusted CA.

The TOE relies on the platform to provide all of its cryptographic functionality. The following Android evaluations are conformant to the Common Criteria for IT Security Evaluation (ISO Standard 15408) and are listed at the National Information Assurance Partnership (NIAP) Product Compliant List.

The TOE was tested on the following Android 12 and Android 13 platforms:

Android 13

https://www.niap-ccevs.org/Product/Compliant.cfm?PID=11342

Samsung Electronics Co., Ltd. Samsung Galaxy Devices on Android 13- Spring

Validation Report Number: CCEVS-VR-VID11342-2023

Certificate Date: 2023.04.26

Device Name	Chipset Vendor	SoC	Architecture	Kernel Version	Kernel Crypto Version
Galaxy A53 5G	Samsung	Exynos 1280	ARMv8	5.10	2.2

Table 3 Android 13

CAVP Certificates

The Samsung SCrypto v2.6 Trusted Execution Environment (TrustZone) (TEE) library provides the following algorithms. Note that the TEE performs RSA signing/decryption (using the private key), while the BoringSSL library performs public key verification/encryption.

The Samsung vendor-affirms that the TOE's RSA key establishment follows 800- 56B. The TOE implementation meets RFC 3526 Section 3. The user and administrator need take no special configuration of the TOE as the TOE automatically generates the keys needed for negotiated TLS ciphersuites. Because the TOE only acts as a TLS client, the TOE only performs 800-56B encryption (specifically the encryption of the Pre-Master Secret using the Server's RSA public key) when participating in TLS_RSA_* based TLS handshakes. Thus, the TOE does not perform 800-56B decryption. However, the TOE's TLS client correctly handles other cryptographic errors (for example, invalid checksums, incorrect certificate types, corrupted certificates) by sending a TLS fatal alert.

SFR	Algorithm	NIST Standard	CAVP Certificate
FTP_DIT_EXT.1	AES CBC/GCM 128/256	FIPS 197, SP 800-38A/D	A915
FTP_DIT_EXT.1	DRBG AES-256 CTR_DRBG	SP 800-90A	A915
FCS_CKM.1/AK FTP_DIT_EXT.1	ECDSA PKG/PKV/SigGen/SigVer P-256/384/521	FIPS 186-4	A915
FTP_DIT_EXT.1	HMAC SHA-1/256/384/512	FIPS 198-1 & 180-4	A915
FCS_CKM.1/AK FTP_DIT_EXT.1	RSA KeyGen and SigGen (no verification) 2048 bits	FIPS 186-4	A915
FTP_DIT_EXT.1	SHS SHA-1/256/384/512	FIPS 180-4	A915

Table 4 Android 13: Samsung SCrypto TEE

The platform BoringSSL v1.7 library (with both Processor Algorithm Accelerators (PAA) and without PAA) provides the following algorithms.

SFR	Algorithm	NIST Standard	CAVP Certificate
FTP_DIT_EXT.1	AES 128/256 CBC, GCM	FIPS 197, SP 800-38A/D/F	A3285
FCS_CKM.2 FTP_DIT_EXT.1	KAS ECC P-256/384/521	SP 800-56A	A3285
FTP_DIT_EXT.1	DRBG CTR – AES-256	SP 800-90A	A3285
FCS_CKM.1/AK FTP_DIT_EXT.1 The “PKG/PKV” in the algorithm description means public key generation/public key verification. The CAVP Certificate also covers ECC Key Gen/Key Verification.	ECDSA PKG/PKV/SigGen/SigVer P-256/384/521	FIPS 186-4	A3285
FCS_CKM.2 FTP_DIT_EXT.1	RSA-based Key Exchange	Vendor affirm 800-56B	Vendor Affirmed
FTP_DIT_EXT.1	HMAC SHA-1/256/384/512	FIPS 198-1 & 180-4	A3285
FCS_CKM.1/AK FTP_DIT_EXT.1	RSA KeyGen/SigGen/SigVer 2048/3072/4096	FIPS 186-4	A3285
FTP_DIT_EXT.1	SHS SHA-1/256/384/512	FIPS 180-4	A3285

Table 5 Android 13: BoringSSL v1.7

Android 12

https://www.niap-ccevs.org/Product/Compliant.cfm?PID=11239

Google Pixel Phones on Android 12.0

Validation Report Number: CCEVS-VR-VID11239-2022

Certificate Date: 2022.02.28

Device Name	Model #	SoC	Kernel Version
Google Pixel 6	GR1YH, GB7N6, G9S9B	Google Tensor (Mali-G78 MP20)	5.10

Table 6 Android 12

The platform BoringSSL Library (version dcdc7bbc6e59ac0123407a9dc4d1f43dd0d117cd) provides the following algorithms as validated on Android 12:

SFR	Algorithm	NIST Standard	CAVP Certificate
FCS_CKM.1/AK FTP_DIT_EXT.1	RSA Key Generation 2048/3072/4096	FIPS 186-4, RSA	A1109
FCS_CKM.1/AK FTP_DIT_EXT.1	ECDSA ECC Key Generation P-256, P-384, P-521	FIPS 186-4, ECDSA	A1109
FCS_CKM.2 FTP_DIT_EXT.1	KAS ECC P-256/384/521	SP 800-56A, CVL KAS ECC	A1109
FCS_CKM.2 FTP_DIT_EXT.1	RSA-based Key Exchange	Vendor affirm 800-56B	Vendor Affirmed
FTP_DIT_EXT.1	AES 128/256 CBC, GCM	FIPS 197, SP 800-38A/D/F	A1109
FTP_DIT_EXT.1	SHA Hashing SHA-1/256/384/512	FIPS 180-4	A1109
FTP_DIT_EXT.1	RSA Sign/Verify 2048/3072/4096	FIPS 186-4, RSA	A1109
FTP_DIT_EXT.1	ECDSA Sign/Verify P-256, P-384, P-521	FIPS 186-4, ECDSA	A1109
FTP_DIT_EXT.1	HMAC-SHA 1/256/384/512	FIPS 198-1 & 180-4	A1109
FTP_DIT_EXT.1	DRBG Bit Generation AES-256	SP 800-90A (Counter)	A1109

Table 7 Android 12: Boring SSL Cryptographic Algorithms

6.1.1 FCS_CKM_EXT.1

The TOE requires asymmetric key generation services to provide secure communications to the Virtual Device on the Hypori Server. The platform generates all ephemeral TLS keys without direct Hypori Halo Client action.

6.1.2 FCS_CKM.1/AK

The Android 12 and 13 platforms call the BoringSSL libraries for the platform to create the ECC and RSA keys. Both BoringSSL libraries generate the P-256, P-384, P-521 Elliptic Curve keys and the RSA 2048, 3072, and 4096 key sizes. The TOE invokes asymmetric key generation for establishing communications to the Virtual Device on the Hypori server.

The Hypori Server informs the Hypori Halo Client which ciphersuites are to be used via the installed client certificates.

6.1.3 FCS_CKM.2

The TOE invokes platform provided RSA and ECC key establishment schemes for establishing communications to the Virtual Device on the Hypori server.

6.1.4 FCS_RBG_EXT.1

The Hypori Halo Client relies on the platform for cryptographic services. Consequently, the Hypori Client itself uses no DRBG functions.

6.1.5 FCS_STO_EXT.1

Error! Reference source not found. lists each Hypori Halo Client persistent credential along with how the client uses and stores each credential.

Table 8 Persistent Credential Use and Storage

Credential	Purpose	Storage
User TLS client private key	Authenticates Hypori Halo Client when establishing TLS connection to Hypori server	Android Keystore System

6.2 User data protection

The Hypori Client uses the platform's permission mechanisms to inform the user of hardware and software resources the client accesses. A user initiates network connections to the Hypori server. In general, sensitive data resides on the Hypori server and is not stored on the Hypori Client. Sensitive data on the Hypori Client is limited to credentials, which the client stores as described in section 6.1.5. The client does not maintain Personally Identifiable Information (PII).

6.2.1 FDP_DAR_EXT.1

Hypori Halo Client sensitive data consist of the user TLS client private key credential. FCS_STO_EXT.1 Storage of Secrets specifies the platform's Android Keystore System for protecting keys and credentials (see https://developer.android.com/training/articles/keystore for details on the Android Keystore System). In accordance with FCS_STO_EXT.1, the Hypori Client stores the user TLS client private key in the platform's Android Keystore System as described in Section 6.1.5. Administrators can decide to provision credentials using the Android Keystore System (either the system-wide Android KeyChain or the application-only Android Keystore Provider).

6.2.2 FDP_NET_EXT.1

The Hypori Halo Client relies on user-initiated network communication to connect to the Hypori Virtual Device.

The Hypori Halo Client uses application-initiated network communication to periodically check for notifications and display them to the user when the system is configured for notification polling.

6.2.3 FDP_DEC_EXT.1

The installer presents to the user the permissions required by the Hypori Halo Client. A user must accept the permissions to complete installation. The permissions identified with a "*" are the permissions required for installation.

Table shows the permissions required by the Hypori Halo Client:

Table 9 Permissions

Permission	Description
INTERNET	Open network sockets.
USE_FINGERPRINT (Deprecated)	Use fingerprint hardware.
USE_BIOMETRIC	Enable Biometric Authentication Factors
WAKE_LOCK	Prevent device from sleeping.
RECORD_AUDIO *	Enable audio recording.
ACCESS_COARSE_LOCATION *	Access rough location.
ACCESS_FINE_LOCATION *	Access precise location.
ACCESS_LOCATION_EXTRA_COMMANDS	Access extra location provider commands.
READ_SYNC_SETTINGS	Read the sync settings.
WRITE_SYNC_SETTINGS	Write the sync settings.
ACCESS_NETWORK_STATE	Access information about networks.
CHANGE_NETWORK_STATE	Change network connectivity state.
ACCESS_WIFI_STATE	Access information about Wi-Fi networks.
MODIFY_AUDIO_SETTINGS	Modify global audio settings.
READ_PHONE_STATE	Read only access to phone state, including the phone number of the device, current cellular network information, the status of any ongoing calls. The READ_PHONE_STATE is used to make a call to the OS to determine the type of mobile data connection (WiFi or Cellar Network) to be used by the phone.
CAMERA *	Access the mobile device's camera.
INSTALL_SHORTCUT	Install a shortcut in the Launcher.
UNINSTALL_SHORTCUT	Uninstall a shortcut in the Launcher.
BLUETOOTH	Connect to paired Bluetooth devices.
BLUETOOTH_ADMIN	Discover and pair Bluetooth devices.
BLUETOOTH_CONNECT	This permission is used by our app to provide information about the Bluetooth device connected to the physical device. This is critical for the operation our application. The user is asked if they want to grant this permission to the app and the user can revoke this permission at any time.
RECEIVE_BOOT_COMPLETED	Receive notification after the system finishes booting.
CALL_PHONE *	Initiate a phone call bypassing the Dialer interface to confirm the call.

Permission	Description
VIBRATE	Access to the mobile device’s vibrator.
FLASHLIGHT	Access to the mobile device’s flashlight.
GET_ACCOUNTS (Deprecated)	Access to the list of accounts in the Accounts Service.
MANAGE_ACCOUNTS	Allow app to add and remove accounts.
AUTHENTICATE_ACCOUNTS (Deprecated)	Use the account authenticator capabilities of the AccountManager.
GET_TASKS (Deprecated)	Allow the app to retrieve information about currently and recently running tasks.
POST_NOTIFICATIONS *	This permission is used by the Hypori Halo Client to display local notifications in the notification tray.
com.google.android.c2dm.permission.RECEIVE	This permission allows the Hypori Halo Client to receive messages sent by the app’s service.

Updates to the Hypori Halo Client may automatically add additional capabilities. A user must accept new permissions to complete any update that includes permissions not in the list above.

A user initiates a network connection to the Virtual Device on the Hypori server by starting the Hypori Client and entering account information.

After the Hypori Halo Client connects to the Hypori server, the applications the user accesses run on the Hypori Device in the Hypori server, not on the mobile device. The Hypori Halo Client does not listen on any ports for inbound connection requests. The Hypori Client interacts only with the Hypori server. When a Hypori Device application needs information from a server (such as a map server), the Hypori Device – not the Hypori Halo Client – communicates with the server (which may be an internal, enterprise server).

The Hypori Halo Client does not maintain PII. Hence, it does not transmit PII over any network. As per the claimed PP, the TOE is not considered to maintain PII unless it provides an interface intended specifically to collect such data; general-purpose communications interfaces may contain PII supplied by the user that the TSF is not expected to treat in a special manner.

The TOE does not contain sensitive information repositories nor does it access any sensitive information platform repositories as defined in the [PP_APP_v1.4].

6.3 Identification and authentication

The Android platform follows RFC 5280 Internet X.509 Public Key Infrastructure for certification path validation[6]. The Hypori Halo Client uses the Android certification validation services to authenticate the X.509 certificate the Hypori server presents as part of the establishing a TLS connection. The TOE relies on the platform for revocation status of the certificate using the Online Certificate Status Protocol (OCSP) as specified in RFC 6960.

6.3.1 FIA_X509_EXT.1

The Android platform performs certificate path validation in accordance with RFC 5280 as part of the TLS service. It recursively builds certificate chains until a valid chain is found or all possible paths are exhausted. The chain begins at the leaf certificate and ends in the final trusted root certificate.

The Hypori Client relies on the platform for TLS services. Hence, the platform checks extended key usage for Server Authentication, Client Authentication, and Code Signing purposes. The platform validates the revocation status of

[6] The platform certificate path algorithm is described by its Android platform source code, available at: https://cs.android.com/android/platform/superproject/main/+/main:external/conscrypt/common/src/main/java/org/conscrypt/TrustManagerImpl.java;drc=4e0deaa2d05fbda3fa32ce892ba1debc3f3a4158;l=393?q=TrustManagerImpl&ss=android%2Fplatform%2Fsuperproject%2Fmain. See lines 380 and line 393 for the algorithm.

the certificate using OCSP as specified in RFC 6960. When the platform cannot establish a connection to the OCSP server to determine revocation status, the platform will reject the connection.

Neither the platform or the Hypori Client performs email encryption, email signature verification, and Enrollment over Secure Transport. Consequently, no check is made for extended key usage Email Protection and CMC Registration Authority purposed.

6.3.2 FIA_X509_EXT.2

The Hypori Halo Client can be used to contact the Hypori provisioning portal and download the user's credentials and store them into either the Android KeyChain or via the application-specific Android Keystore provider. The destination is controlled by administrator policies. The user's credentials are stored in the Android KeyChain or via the application-specific Android Keystore provider using the following methods:

- Configure the Hypori Halo Client Account using a QR Code
    - The user will receive an enrollment email from the Hypori Halo administrator titled "Your Hypori account is ready". This email contains the QR code required to add the account as well as the account information. After the Hypori Halo Client is installed and launched, the application will ask for permission to allow the Hypori Halo Client access to your physical device's camera. The camera will automatically scan the QR code, proceed with the installation process, and save the credential information in the key store and connect the user to virtual workspace.
- Configure the Hypori Halo Client using the One-Time Password (OTP) Method
    - The OTP method of account provisioning is mostly used by users who may have the camera disabled on their physical device. The user will receive an enrollment email from the Hypori Halo administrator titled "Your Hypori account is ready". After the Hypori Halo Client is installed and launched, the application will ask for the user to populate fields using the information provided in the "Your Hypori account is ready" email: Email Address or Login Name, Server URL/Port Number, One-Time Password (OTP). The user can optionally change the Account Name. The installation process, will save the credential information in the key store and connect the user to virtual workspace.

The Hypori Client presents the TLS client certificate and public key to the Hypori server to authenticate a TLS connection. The TLS client certificate is an X.509 certificate.

The user stores a CA certificate for the server certificates in the platform's key store during installation. (The user need not install a CA certificate when the CA is a platform trusted CA). On Android devices, the Hypori Client uses Android platform certificate path validation services with the CA certificate to validate the certificate presented by the Hypori server. The Hypori Client extracts the OCSP information from the certificate and performs the revocation checking to ensure that the certificate has not been revoked.

If the OCSP server fails to respond or there is an error, the Hypori Client will not accept the certificate (invalid) and not establish the connection.

6.4 Security management

Security management consists of setting Hypori Client configuration options. The client uses Android mechanisms for storing the configuration settings.

6.4.1 FMT_CFG_EXT.1

Hypori Halo Client credentials consist of the user TLS client private key. The Hypori Halo Client installer does not include a default client key. The TOE obtains and stores the certificate and private key from the server during initial configuration. The user is not able to access any TOE functionality prior to the installation of the TLS client certificate and private key.

The default file permissions protect the application's binaries and data files from modification by normal unprivileged users.

The “default permissions” are those provided by Android. In particular, the base apk permission is 644, which breaks down to the following:

- it is able to be read + written to by owner
- it is able to be read by group,
- it is able to be read by others.

The shared library files permissions are 755 as required by Android, which break down to:

- the owner which is root will read, write and execute in the directory,
- the group will only read and execute in the directory,
- others will only read and execute in the directory.

All of these files are owned by system/system.

Preferences/options files are stored in the shared_prefs directory with permissions 660, which breaks down to the following:

- the owner can read and write but not execute,
- the group is able to read and write but not execute,
- others cannot read, write or execute,

The preferences/options files are owned by the uid/gid associated with the application (varies per installation). These are all defined by Android. The preferences/options files contain encrypted key value pairs.

6.4.2 FMT_MEC_EXT.1

The Hypori Halo Client invokes the recommended Android mechanisms for storing account settings files. EncryptedSharedPreferences is the platformed provided mechanism for saving configuration data for the application. EncryptedSharedPreferences wraps the SharedPreferences class.

The account options stored in EncryptedSharedPreferences consists of the Hypori Server hostname (URL), port number of the Hypori Server, Account Name, and the email address. The androidx.security.crypto.EncryptedSharedPreferences API is invoked without any user intervention.

The Hypori Halo Client policies downloaded from the Hypori server are also stored using EncryptedSharedPreferences. Client policies are downloaded from the server during initial configuration and every time the client authenticates.

The description of the Android EncryptedSharedPreferences API is found at: https://developer.android.com/reference/androidx/security/crypto/EncryptedSharedPreferences.

6.4.3 FMT_SMF.1

For each account, the Hypori Halo Client provides the capability to initially set the Hypori server URL, Hypori server port, account name, email address, and TLS client certificate (private key). Except for the private key, these values are provided to the user and either manually entered during initial configuration or obtained by the TOE when the user scans the QR Code. The TOE acquires the TLS client certificate (private key) from the Hypori backend server during the account setup process. After the account has been set-up, the user only has the capability to change the account name.

Client policies are automatically downloaded from the Hypori Server and are applied during initial configuration. After initial configuration, the Hypori Halo Client policies are downloaded and refreshed every time the Hypori Halo Client authenticates to the Hypori Server, whether or not any changes have been made on the Hypori Server. The Hypori Halo Client does not listen on any ports for inbound connection requests. The Hypori Halo Client interacts only with the Hypori server. When a virtual Hypori Device application needs information from a server (such as a map server), the virtual Hypori Device – not the Hypori Halo Client – communicates with the server (which may be an internal, enterprise server).

6.5 Privacy

The Hypori Halo Client does not transmit PII over a network.

6.5.1 FPR_ANO_EXT.1

The Hypori Halo Client does not transmit PII over a network.

6.6 Protection of the TSF

The Hypori Halo Client uses security features and APIs that the platform provides. This includes address space layout randomization, data execution protection, Security Enhancements for Android, and stack-based buffer overflow protection. The client leverages Android package management for secure installation and updates. The Hypori Halo Client package includes only those third-party libraries necessary for its intended operation.

6.6.1 FPT_AEX_EXT.1

Hypori enables address space layout randomization (ASLR) in the Android Hypori Client using `-fpic` when building the application with Android Native Development Kit (NDK r15c) using gcc. The Hypori Client is a Java application that includes Java Native Interface (JNI) libraries. Hypori enables stack-based buffer overflow protection using `-fstack-protector-strong`. The Hypori Client does not invoke `mmap` or `mprotect` from the Android NDK.

6.6.2 FPT_API_EXT.1

The Hypori Halo Client uses the Android APIs listed in Section 9 Appendix: Android APIs and Section 10. Appendix: Java Library APIs.

6.6.3 FPT_IDV_EXT.1

The TOE is the Hypori Halo Client (Android) v 4.3. The TOE is identified and versioned by the Android application version identifiers as well as the internal Hypori build information.

The values listed below are provided for illustration purposes:

- Version name: 4.3.0 (on Play Store and in App)
- Version code: 403140004 (hidden inside app, not exposed to user)
- Internal build code (created by build system, shown in App)

Below is an example of version string as shown in Hypori UI:

- 4.3.0 (403000019-a53904e)

The Hypori (Android) Client 4.3.0 version is interpreted as a major.minor.maintenance-release format.

6.6.4 FPT_LIB_EXT.1

The Hypori Client package includes only the third-party libraries listed below:

- Opus Audio Codec v1.1
- Protobuf v3.21.1
- Zxing core 3.3.0
- Yubikit v1.0.0
- AppAuth 0.9.1
- Moshi 1.13.0
- BouncyCastle 1.70
- Hasher 1.2
- Kotlin standard library 1.8.10
- kotlin-reflect 1.8.10

- kotlinx-coroutines 1.6.4
- Dagger Hilt v2.45

6.6.5 FPT_TUD_EXT.1, FPT_TUD_EXT.2

Hypori distributes the Hypori Halo Client as a .APK file for Android devices. A user may obtain the installation package through Google Play or the enterprise IT group of the user. A user obtains Hypori Halo Client updates using the platform's update mechanism or from the user's IT group. Hypori digitally signs the installation package as well as updates with a unique certificate and corresponding private key; and includes the corresponding public key certificate in the package. Android verifies the digital signature on the package using the public key in the certificate. The installation or software update process will only occur if the signature validation is successful. It can be delivered via the Google Play store, MDM, or other enterprise app stores.

To verify the version of the Hypori Halo Client, open the Hypori Halo Client, but do not connect to the Virtual Device. On the Hypori Client Accounts screen, select the ellipses menu and click on 'About'. The About screen will display the version number, build information and copyright.

6.7 Trusted path/channels

The Hypori Halo Client uses TLS 1.2 for all communication with Hypori server.

6.7.1 FTP_DIT_EXT.1

The Hypori server is the only trusted IT product the Hypori Halo Client communicates with. For all communication with the Hypori server, the Hypori Halo Client connects to the server using TLS 1.2 provided by the platform.

The TOE uses the platform android.net.SSLCertificateSocketFactory and javax.net.ssl.SSLSocket calls to invoke the functionality.

6.8 Timely Security Updates

6.8.1 ALC_TSU_EXT.1

Hypori provides customers with timely updates. A customer chooses their preferred communication. The Hypori Support Department will notify customers of updates using each customer's preferred communication mechanism. Application changes may be pushed to end users via the Google Play Store like any other application or via an enterprise application store internal to a customer. Typical delivery times for security updates are 5 to 10 business days.

Hypori maintains a Security Portal online. Every customer is registered with the Support Portal. Hypori notifies each customer of a new security report on the Support portal using the customers preferred communication mechanism. Hypori secures the Support Portal via SSL and user authentication. Each customer contact must log in with their specific credentials in order to see the security reports.

7. Protection Profile Claims

This ST conforms to the *Protection Profile for Application Software,* Version 1.4, 2021-10-07 [PP_APP_v1.4].

As explained in Section 3, Security Problem Definition, the Security Problem Definition of the [PP_APP_v1.4] has been included by reference into this ST.

As explained in Section 4, Security Objectives, the Security Objectives of the [PP_APP_v1.4] have been included by reference into this ST.

The following table identifies all the security functional requirements in this ST. Each SFR is reproduced from the [PP_APP_v1.4] and operations completed as appropriate.

Table 10 SFR Protection Profile Sources

Requirement Class	Requirement Component	Source
FCS: Cryptographic support	FCS_CKM_EXT.1 Cryptographic Key Generation Services	[PP_APP_v1.4]
	FCS_CKM.1/AK Cryptographic Asymmetric Key Generation	[PP_APP_v1.4]
	FCS_CKM.2 Cryptographic Key Establishment	[PP_APP_v1.4]
	FCS_RBG_EXT.1 Random Bit Generation Services	[PP_APP_v1.4]
	FCS_STO_EXT.1 Storage of Credentials	[PP_APP_v1.4]
FDP: User data protection	FDP_DAR_EXT.1 Encryption of Sensitive Application Data	[PP_APP_v1.4]
	FDP_DEC_EXT.1 Access to Platform Resources	[PP_APP_v1.4]
	FDP_NET_EXT.1 Network Communications	[PP_APP_v1.4]
FIA: Identification and authentication	FIA_X509_EXT.1 X.509 Certificate Validation	[PP_APP_v1.4]
	FIA_X509_EXT.2 X.509 Certificate Authentication	[PP_APP_v1.4]
FMT: Security management	FMT_CFG_EXT.1 Secure by Default Configuration	[PP_APP_v1.4]
	FMT_MEC_EXT.1 Supported Configuration Mechanism	[PP_APP_v1.4]
	FMT_SMF.1 Specification of Management Functions	[PP_APP_v1.4]
FPR: Privacy	FPR_ANO_EXT.1 User Consent for Transmission of Personally Identifiable Information	[PP_APP_v1.4]
FPT: Protection of the TSF	FPT_AEX_EXT.1 AntiExploitation Capabilities	[PP_APP_v1.4]
	FPT_API_EXT.1.1 Use of Supported Services and APIs	[PP_APP_v1.4]
	FPT_IDV_EXT.1 Software Identification and Versions	[PP_APP_v1.4]
	FPT_LIB_EXT.1 Use of Third Party Libraries	[PP_APP_v1.4]
	FPT_TUD_EXT.1 Integrity for Installation and Update	[PP_APP_v1.4]
	FPT_TUD_EXT.2 Integrity for Installation and Update	[PP_APP_v1.4]
FTP: Trusted path/channels	FTP_DIT_EXT.1 Protection of Data in Transit	[PP_APP_v1.4]

8. Rationale

This security target includes by reference the [PP_APP_v1.4] Security Problem Definition, Security Objectives, and Security Assurance Requirements. The security target makes no additions to the [PP_APP_v1.4] assumptions. [PP_APP_v1.4] security functional requirements have been reproduced with the [PP_APP_v1.4] operations completed. Operations on the security requirements follow [PP_APP_v1.4] application notes and assurance activities. Consequently, [PP_APP_v1.4] rationale applies but is incomplete. The TOE Summary Specification rationale below serves to complete the rationale required for the security target.

8.1 Dependency Rationale

The Protection Profile for Application Software [PP_APP_v1.4] contains all the requirements claimed in this Security Target. As such, the dependencies are not applicable since the PP has been approved.

8.2 TOE Summary Specification Rationale

Each subsection in Section 6, the TOE Summary Specification, describes a security function of the TOE. Each description is followed with rationale that indicates which requirements are satisfied by aspects of the corresponding security function. The security functions work together to satisfy all of the security functional requirements. Furthermore, all of the security functions are necessary in order for the TSF to provide the required security functionality.

This section in conjunction with Section 6 TOE Summary Specification provides evidence that the security functions are suitable to meet the TOE security requirements. The collection of security functions works together to provide all of the security requirements. The security functions described in the TOE summary specification are all necessary for the required security functionality in the TSF. Table 11 demonstrates the relationship between security requirements and security functions.

Table 11 Security Functions vs. Requirements Mapping

	Cryptographic support	User data protection	Identification and authentication	Security management	Privacy	Protection of the TSF	Trusted path/channels
FCS_CKM_EXT.1	X						
FCS_CKM.1/AK	X						
FCS_CKM.2	X						
FCS_RBG_EXT.1	X						
FCS_STO_EXT.1	X						
FDP_DAR_EXT.1		X					
FDP_NET_EXT.1		X					
FDP_DEC_EXT.1		X					
FIA_X509_EXT.1			X				
FIA_X509_EXT.2			X				
FMT_CFG_EXT.1				X			
FMT_MEC_EXT.1				X			
FMT_SMF.1				X			
FPR_ANO_EXT.1					X		
FPT_AEX_EXT.1						X	
FPT_API_EXT.1						X	
FPT_IDV_EXT.1						X	

	Cryptographic support	User data protection	Identification and authentication	Security management	Privacy	Protection of the TSF	Trusted path/channels
FPT_LIB_EXT.1						X	
FPT_TUD_EXT.1						X	
FPT_TUD_EXT.1						X	
FTP_DIT_EXT.1							X

9. Appendix: Android APIs

The Hypori Halo Client uses the following Android APIs:

1. android.Manifest
2. android.accounts.AbstractAccountAuthenticator
3. android.accounts.Account
4. android.accounts.AccountAuthenticatorResponse
5. android.accounts.AccountManager
6. android.accounts.AccountManagerCallback
7. android.accounts.AccountManagerFuture
8. android.accounts.AccountsException
9. android.accounts.NetworkErrorException
10. android.accounts.OnAccountsUpdateListener
11. android.accounts.OperationCanceledException
12. android.animation.Animator
13. android.animation.AnimatorListenerAdapter
14. android.animation.ValueAnimator
15. android.annotation.SuppressLint
16. android.annotation.TargetApi
17. android.app.ActionBar
18. android.app.Activity
19. android.app.ActivityManager.RunningTaskInfo
20. android.app.ActivityManager
21. android.app.AlertDialog
22. android.app.Application
23. android.app.Application.ActivityLifecycleCallbacks
24. android.app.Application
25. android.app.Dialog
26. android.app.DialogFragment
27. android.app.IntentService
28. android.app.KeyguardManager
29. android.app.ListActivity
30. android.app.Notification
31. android.app.NotificationChannel
32. android.app.NotificationChannelGroup
33. android.app.NotificationManager
34. android.app.PendingIntent
35. android.app.ProgressDialog

36. android.app.SearchManager
37. android.app.SearchableInfo
38. android.app.Service
39. android.app.admin.DevicePolicyManager
40. android.bluetooth.BluetoothAdapter
41. android.bluetooth.BluetoothClass
42. android.bluetooth.BluetoothDevice
43. android.bluetooth.BluetoothGatt
44. android.bluetooth.BluetoothGattCallback
45. android.bluetooth.BluetoothGattCharacteristic
46. android.bluetooth.BluetoothGattDescriptor
47. android.bluetooth.BluetoothGattService
48. android.bluetooth.BluetoothHeadset
49. android.bluetooth.BluetoothManager
50. android.bluetooth.BluetoothProfile
51. android.bluetooth.BluetoothServerSocket
52. android.bluetooth.BluetoothSocket
53. android.content.AbstractThreadedSyncAdapter
54. android.content.ActivityNotFoundException
55. android.content.BroadcastReceiver
56. android.content.ComponentName
57. android.content.ContentProvider
58. android.content.ContentProviderClient
59. android.content.ContentResolver
60. android.content.ContentUris
61. android.content.ContentValues
62. android.content.Context
63. android.content.DialogInterface.OnClickListener
64. android.content.DialogInterface
65. android.content.Intent.ShortcutIconResource
66. android.content.Intent
67. android.content.IntentFilter
68. android.content.RestrictionsManager
69. android.content.ServiceConnection
70. android.content.SharedPreferences.Editor
71. android.content.SharedPreferences.OnSharedPreferenceChangeListener
72. android.content.SharedPreferences

73. android.content.SyncResult
74. android.content.UriMatcher
75. android.content.pm.ActivityInfo
76. android.content.pm.ActivityInfo
77. android.content.pm.PackageInfo
78. android.content.pm.PackageManager
79. android.content.pm.PackageManager.NameNotFoundException
80. android.content.pm.PackageManager
81. android.content.pm.ResolveInfo
82. android.content.res.AssetManager
83. android.content.res.Configuration
84. android.content.res.Resources
85. android.content.res.TypedArray
86. android.database.ContentObserver
87. android.database.Cursor
88. android.database.DataSetObserver
89. android.database.MatrixCursor
90. android.database.sqlite.SQLiteDatabase
91. android.database.sqlite.SQLiteOpenHelper
92. android.graphics.Bitmap.CompressFormat
93. android.graphics.Bitmap
94. android.graphics.BitmapFactory
95. android.graphics.Canvas
96. android.graphics.Color
97. android.graphics.ImageFormat
98. android.graphics.Matrix
99. android.graphics.Paint
100. android.graphics.Path
101. android.graphics.PixelFormat
102. android.graphics.Point
103. android.graphics.PointF
104. android.graphics.PorterDuff
105. android.graphics.Rect
106. android.graphics.RectF
107. android.graphics.SurfaceTexture.OnFrameAvailableListener
108. android.graphics.SurfaceTexture
109. android.graphics.YuvImage

110. android.graphics.drawable.BitmapDrawable
111. android.graphics.drawable.ColorDrawable
112. android.graphics.drawable.Drawable
113. android.graphics.drawable.GradientDrawable
114. android.graphics.drawable.LayerDrawable
115. android.graphics.drawable.TransitionDrawable
116. android.hardware.Camera.Area
117. android.hardware.Camera.CameraInfo
118. android.hardware.Camera.Face
119. android.hardware.Camera.FaceDetectionListener
120. android.hardware.Camera.Parameters
121. android.hardware.Camera.PictureCallback
122. android.hardware.Camera.Size
123. android.hardware.Camera
124. android.hardware.Sensor
125. android.hardware.SensorEvent
126. android.hardware.SensorEventListener
127. android.hardware.SensorManager
128. android.location.Location
129. android.location.LocationManager
130. android.location.LocationProvider
131. android.location.OnNmeaMessageListener
132. android.media.AudioDeviceInfo
133. android.media.AudioFormat
134. android.media.AudioManager
135. android.media.AudioRecord
136. android.media.AudioTrack
137. android.media.CamcorderProfile
138. android.media.CameraProfile
139. android.media.MediaActionSound
140. android.media.MediaCodec.BufferInfo
141. android.media.MediaCodec
142. android.media.MediaCodecInfo
143. android.media.MediaCodecList
144. android.media.MediaFormat
145. android.media.MediaMetadataRetriever
146. android.media.MediaRecorder

147. android.media.ThumbnailUtils
148. android.media.audiofx.AcousticEchoCanceler
149. android.media.audiofx.AutomaticGainControl
150. android.media.audiofx.NoiseSuppressor
151. android.net.ConnectivityManager
152. android.net.LinkAddress
153. android.net.LinkProperties
154. android.net.Network
155. android.net.NetworkCapabilities
156. android.net.NetworkInfo
157. android.net.NetworkRequest
158. android.net.RouteInfo
159. android.net.SSLCertificateSocketFactory
160. android.net.Uri.Builder
161. android.net.Uri
162. android.net.wifi.WifiInfo
163. android.net.wifi.WifiManager.WifiLock
164. android.net.wifi.WifiManager
165. android.opengl.EGL14
166. android.opengl.EGLConfig
167. android.opengl.EGLContext
168. android.opengl.EGLDisplay
169. android.opengl.EGLExt
170. android.opengl.EGLSurface
171. android.opengl.GLES11Ext
172. android.opengl.GLES20
173. android.opengl.GLSurfaceView.Renderer
174. android.opengl.GLSurfaceView
175. android.opengl.GLUtils
176. android.opengl.Matrix
177. android.os.AsyncTask
178. android.os.BatteryManager
179. android.os.Binder
180. android.os.Build
181. android.os.Bundle
182. android.os.CountDownTimer
183. android.os.Environment

184. android.os.Handler
185. android.os.HandlerThread
186. android.os.IBinder
187. android.os.Looper
188. android.os.Message
189. android.os.Parcel
190. android.os.ParcelFileDescriptor.AutoCloseOutputStream
191. android.os.ParcelFileDescriptor
192. android.os.Parcelable
193. android.os.PowerManager
194. android.os.StatFs
195. android.os.SystemClock
196. android.preference.PreferenceManager
197. android.provider.MediaStore.Images.ImageColumns
198. android.provider.MediaStore.Images
199. android.provider.MediaStore.MediaColumns
200. android.provider.MediaStore.Video.VideoColumns
201. android.provider.MediaStore.Video
202. android.provider.MediaStore
203. android.provider.Settings.System
204. android.provider.Settings
205. android.renderscript.ScriptGroup
206. android.security.KeyChain
207. android.security.KeyChainAliasCallback
208. android.security.KeyChainException
209. android.security.keystore.KeyGenParameterSpec
210. android.security.keystore.KeyInfo
211. android.security.keystore.KeyProperties
212. android.service.notification.StatusBarNotification
213. android.system.ErrnoException
214. android.telephony.PhoneStateListener
215. android.telephony.ServiceState
216. android.telephony.SignalStrength
217. android.telephony.TelephonyManager
218. android.text.Html
219. android.text.InputFilter
220. android.text.InputFilter

221. android.text.Spanned
222. android.text.TextUtils
223. android.text.TextUtils
224. android.text.format.DateFormat
225. android.text.format.Time
226. android.text.method.LinkMovementMethod
227. android.util.AttributeSet
228. android.util.Base64
229. android.util.Base64OutputStream
230. android.util.DisplayMetrics
231. android.util.Log
232. android.util.Range
233. android.util.SparseArray
234. android.util.TypedValue
235. android.util.Xml
236. android.view.Display
237. android.view.DisplayCutout
238. android.view.GestureDetector.SimpleOnGestureListener
239. android.view.GestureDetector
240. android.view.Gravity
241. android.view.InflateException
242. android.view.InputDevice
243. android.view.KeyEvent
244. android.view.LayoutInflater
245. android.view.Menu
246. android.view.MenuInflater
247. android.view.MenuItem.OnMenuItemClickListener
248. android.view.MenuItem
249. android.view.MotionEvent
250. android.view.OrientationEventListener
251. android.view.SoundEffectConstants
252. android.view.Surface
253. android.view.SurfaceHolder
254. android.view.SurfaceView
255. android.view.VelocityTracker
256. android.view.View.OnClickListener
257. android.view.View

258. android.view.ViewConfiguration
259. android.view.ViewGroup.LayoutParams
260. android.view.ViewGroup
261. android.view.ViewTreeObserver
262. android.view.Window
263. android.view.WindowInsets
264. android.view.WindowInsetsController
265. android.view.WindowManager
266. android.view.accessibility.AccessibilityEvent
267. android.view.animation.AlphaAnimation
268. android.view.animation.Animation.AnimationListener
269. android.view.animation.Animation
270. android.view.animation.AnimationUtils
271. android.view.inputmethod.EditorInfo
272. android.view.inputmethod.InputConnection
273. android.view.inputmethod.InputMethodManager
274. android.webkit.WebSettings
275. android.webkit.WebView
276. android.webkit.WebViewClient
277. android.widget.AbsListView
278. android.widget.Adapter
279. android.widget.AdapterView.OnItemClickListener
280. android.widget.AdapterView.OnItemLongClickListener
281. android.widget.AdapterView
282. android.widget.ArrayAdapter
283. android.widget.BaseAdapter
284. android.widget.Button
285. android.widget.CheckBox
286. android.widget.CompoundButton.OnCheckedChangeListener
287. android.widget.CompoundButton
288. android.widget.CursorAdapter
289. android.widget.EditText
290. android.widget.Filter
291. android.widget.Filterable
292. android.widget.FrameLayout
293. android.widget.GridView
294. android.widget.HorizontalScrollView

295. android.widget.ImageButton
296. android.widget.ImageView.ScaleType
297. android.widget.ImageView
298. android.widget.LinearLayout
299. android.widget.ListView
300. android.widget.PopupMenu
301. android.widget.PopupWindow
302. android.widget.ProgressBar
303. android.widget.RelativeLayout
304. android.widget.SearchView
305. android.widget.SimpleAdapter
306. android.widget.Switch
307. android.widget.TextView
308. android.widget.Toast
309. androidx.activity.OnBackPressedCallback
310. androidx.activity.compose.setContent
311. androidx.activity.result.contract.ActivityResultContracts
312. androidx.activity.viewModels
313. androidx.annotation.CallSuper
314. androidx.annotation.Keep
315. androidx.annotation.MainThread
316. androidx.annotation.NonNull
317. androidx.annotation.Nullable
318. androidx.annotation.PluralsRes
319. androidx.annotation.RequiresApi
320. androidx.annotation.StringRes
321. androidx.annotation.StyleRes
322. androidx.annotation.VisibleForTesting
323. androidx.appcompat.app.ActionBar
324. androidx.appcompat.app.AppCompatActivity
325. androidx.appcompat.view.ContextThemeWrapper
326. androidx.biometric.BiometricManager
327. androidx.biometric.BiometricPrompt
328. androidx.compose.foundation.Image
329. androidx.compose.foundation.background
330. androidx.compose.foundation.clickable
331. androidx.compose.foundation.layout.BoxWithConstraintsScope

332. androidx.compose.foundation.rememberScrollState
333. androidx.compose.foundation.verticalScroll
334. androidx.compose.material.MaterialTheme
335. androidx.compose.material.RadioButton
336. androidx.compose.material.Text
337. androidx.compose.runtime.Composable
338. androidx.compose.runtime.CompositionLocalProvider
339. androidx.compose.runtime.Immutable
340. androidx.compose.runtime.mutableStateOf
341. androidx.compose.runtime.remember
342. androidx.compose.ui.Alignment
343. androidx.compose.ui.Modifier
344. androidx.compose.ui.geometry.Size
345. androidx.compose.ui.graphics.toAndroidRect
346. androidx.compose.ui.graphics.toComposeRect
347. androidx.compose.ui.layout.ContentScale
348. androidx.compose.ui.platform.LocalConfiguration
349. androidx.compose.ui.platform.LocalContext
350. androidx.compose.ui.platform.LocalDensity
351. androidx.compose.ui.res.painterResource
352. androidx.compose.ui.res.stringResource
353. androidx.compose.ui.text.style.TextAlign
354. androidx.compose.ui.tooling.preview.Devices
355. androidx.compose.ui.tooling.preview.Preview
356. androidx.compose.ui.unit.Density
357. androidx.compose.ui.unit.DpSize
358. androidx.compose.ui.unit.dp
359. androidx.core.app.ActivityCompat
360. androidx.core.app.NotificationCompat
361. androidx.core.content.ContextCompat
362. androidx.core.content.res.ResourcesCompat
363. androidx.core.location.GnssStatusCompat
364. androidx.core.location.LocationListenerCompat
365. androidx.core.location.LocationManagerCompat
366. androidx.core.view.MotionEventCompat
367. androidx.core.view.ViewConfigurationCompat
368. androidx.core.view.setPadding

369. androidx.core.widget.doAfterTextChanged
370. androidx.drawerlayout.widget.DrawerLayout
371. androidx.fragment.app.DialogFragment
372. androidx.fragment.app.Fragment
373. androidx.fragment.app.FragmentActivity
374. androidx.fragment.app.FragmentManager
375. androidx.fragment.app.FragmentStatePagerAdapter
376. androidx.fragment.app.FragmentTransaction
377. androidx.fragment.app.activityViewModels
378. androidx.lifecycle.AndroidViewModel
379. androidx.lifecycle.LifecycleOwner
380. androidx.lifecycle.LifecycleService
381. androidx.lifecycle.LiveData
382. androidx.lifecycle.MutableLiveData
383. androidx.lifecycle.Observer
384. androidx.lifecycle.ProcessLifecycleOwner
385. androidx.lifecycle.ViewModel
386. androidx.lifecycle.ViewModelProvider
387. androidx.lifecycle.lifecycleScope
388. androidx.lifecycle.viewModelScope
389. androidx.localbroadcastmanager.content.LocalBroadcastManager
390. androidx.navigation.NavController
391. androidx.navigation.Navigation
392. androidx.navigation.fragment.findNavController
393. androidx.navigation.fragment.navArgs
394. androidx.navigation.ui.AppBarConfiguration
395. androidx.navigation.ui.NavigationUI
396. androidx.preference.Preference;
397. androidx.preference.PreferenceDataStore
398. androidx.preference.PreferenceViewHolder
399. androidx.recyclerview.widget.ItemTouchHelper
400. androidx.recyclerview.widget.LinearLayoutManager
401. androidx.recyclerview.widget.LinearLayoutManager
402. androidx.recyclerview.widget.RecyclerView
403. androidx.security.crypto.EncryptedSharedPreferences
404. androidx.security.crypto.MasterKeys
405. androidx.viewbinding.ViewBinding

406. androidx.viewpager.widget.ViewPager
407. androidx.viewpager2.adapter.FragmentStateAdapter
408. androidx.window.layout.WindowMetrics
409. android.windows.layout.WindowMetricsCalculator
410. com.google.android.gms.common.ConnectionResult
411. com.google.android.gms.common.GoogleApiAvailability
412. com.google.android.material.button.MaterialButton
413. com.google.android.material.composethemeadapter.MdcTheme
414. com.google.android.material.snackbar.Snackbar
415. com.google.android.material.tabs.TabLayoutMediator
416. com.google.android.material.textfield.TextInputLayout
417. com.google.firebase.FirebaseApp
418. com.google.firebase.FirebaseOptions
419. com.google.firebase.iid.FirebaseInstanceId
420. com.google.firebase.messaging.FirebaseMessaging
421. com.google.firebase.messaging.FirebaseMessagingService
422. com.google.firebase.messaging.RemoteMessage
423. org.xmlpull.v1.XmlPullParser
424. org.xmlpull.v1.XmlPullParserException
425. java.beans.PropertyChangeEvent
426. java.beans.PropertyChangeListener
427. java.io.BufferedInputStream
428. java.io.BufferedOutputStream
429. java.io.BufferedReader
430. java.io.BufferedWriter
431. java.io.ByteArrayInputStream
432. java.io.ByteArrayOutputStream
433. java.io.Closeable
434. java.io.DataInputStream
435. java.io.DataOutputStream
436. java.io.File
437. java.io.FileDescriptor
438. java.io.FileInputStream
439. java.io.FileNotFoundException
440. java.io.FileOutputStream
441. java.io.FileReader
442. java.io.FileWriter

443.java.io.FilenameFilter

444.java.io.IOException

445.java.io.InputStream

446.java.io.InputStreamReader

447.java.io.ObjectInputStream

448.java.io.ObjectOutputStream

449.java.io.OutputStream

450.java.io.OutputStreamWriter

451.java.io.PrintStream

452.java.io.PrintWriter

453.java.io.RandomAccessFile

454.java.io.Serializable

455.java.io.StringWriter

456.java.io.UnsupportedEncodingException

457.java.io.Writer

458.java.lang.Thread.UncaughtExceptionHandler

459.java.lang.annotation.ElementType

460.java.lang.annotation.Retention

461.java.lang.annotation.RetentionPolicy

462.java.lang.annotation.Target

463.java.lang.ref.WeakReference

464.java.lang.reflect.Array

465.java.lang.reflect.Constructor

466.java.lang.reflect.Field

467.java.lang.reflect.InvocationTargetException

468.java.lang.reflect.Method

469.java.math.BigInteger

470.java.net.ConnectException

471.java.net.HttpURLConnection

472.java.net.InetAddress

473.java.net.MalformedURLException

474.java.net.Socket

475.java.net.SocketException

476.java.net.URL

477.java.net.URLEncoder

478.java.net.UnknownHostException

479.java.net.UnknownServiceException

480.java.nio.BufferOverflowException

481.java.nio.BufferUnderflowException

482.java.nio.ByteBuffer

483.java.nio.ByteOrder

484.java.nio.CharBuffer

485.java.nio.DoubleBuffer

486.java.nio.FloatBuffer

487.java.nio.IntBuffer

488.java.nio.LongBuffer

489.java.nio.ShortBuffer

490.java.nio.charset.StandardCharsets

491.java.security.GeneralSecurityException

492.java.security.InvalidKeyException

493.java.security.InvalidParameterException

494.java.security.Key

495.java.security.KeyFactory

496.java.security.KeyManagementException

497.java.security.KeyPair

498.java.security.KeyPairGenerator

499.java.security.KeyStore

500.java.security.KeyStoreException

501.java.security.NoSuchAlgorithmException

502.java.security.NoSuchProviderException

503.java.security.Principal

504.java.security.PrivateKey

505.java.security.Provider

506.java.security.PublicKey

507.java.security.SecureRandom

508.java.security.SecureRandomSpi

509.java.security.Security

510.java.security.Signature

511.java.security.SignatureException

512.java.security.UnrecoverableKeyException

513.java.security.cert.CertPath

514.java.security.cert.CertPathBuilder

515.java.security.cert.CertPathBuilderException

516.java.security.cert.CertPathValidatorException

517.java.security.cert.CertStore
518.java.security.cert.Certificate
519.java.security.cert.CertificateEncodingException
520.java.security.cert.CertificateException
521.java.security.cert.CertificateExpiredException
522.java.security.cert.CertificateFactory
523.java.security.cert.CertificateNotYetValidException
524.java.security.cert.CertificateParsingException
525.java.security.cert.CertificateRevokedException
526.java.security.cert.CollectionCertStoreParameters
527.java.security.cert.PKIXBuilderParameters
528.java.security.cert.PKIXCertPathBuilderResult
529.java.security.cert.TrustAnchor
530.java.security.cert.X509CertSelector
531.java.security.cert.X509Certificate
532.java.security.interfaces.ECPublicKey
533.java.security.interfaces.RSAPublicKey
534.java.security.spec.AlgorithmParameterSpec
535.java.security.spec.ECParameterSpec
536.java.security.spec.X509EncodedKeySpec
537.java.text.DateFormat
538.java.text.ParseException
539.java.text.SimpleDateFormat
540.java.util.ArrayDeque
541.java.util.ArrayList
542.java.util.Arrays
543.java.util.Calendar
544.java.util.Collection
545.java.util.Collections
546.java.util.Comparator
547.java.util.Date
548.java.util.EmptyStackException
549.java.util.EnumMap
550.java.util.EnumSet
551.java.util.Enumeration
552.java.util.Formatter
553.java.util.HashMap

554.java.util.HashSet

555.java.util.Hashtable

556.java.util.Iterator

557.java.util.LinkedList

558.java.util.List

559.java.util.Locale

560.java.util.Map

561.java.util.Map.Entry

562.java.util.Random

563.java.util.Set

564.java.util.Stack

565.java.util.StringTokenizer

566.java.util.TimeZone

567.java.util.Timer

568.java.util.TimerTask

569.java.util.TreeMap

570.java.util.TreeSet

571.java.util.UUID

572.java.util.WeakHashMap

573.java.util.concurrent.ArrayBlockingQueue

574.java.util.concurrent.Callable

575.java.util.concurrent.CopyOnWriteArrayList

576.java.util.concurrent.CountDownLatch

577.java.util.concurrent.Executor

578.java.util.concurrent.ExecutorService

579.java.util.concurrent.Executors

580.java.util.concurrent.Future

581.java.util.concurrent.LinkedBlockingQueue

582.java.util.concurrent.RejectedExecutionException

583.java.util.concurrent.Semaphore

584.java.util.concurrent.TimeUnit

585.java.util.concurrent.TimeoutException

586.java.util.concurrent.atomic.AtomicBoolean

587.java.util.concurrent.locks.Condition

588.java.util.concurrent.locks.Lock

589.java.util.concurrent.locks.ReentrantLock

590.java.util.regex.Matcher

591.java.util.regex.Pattern

592.javax.crypto.Cipher

593.javax.microedition.khronos.egl.EGLConfig

594.javax.microedition.khronos.opengles.GL10

595.javax.net.ssl.HandshakeCompletedEvent

596.javax.net.ssl.HandshakeCompletedListener

597.javax.net.ssl.HostnameVerifier

598.javax.net.ssl.HttpsURLConnection

599.javax.net.ssl.KeyManager

600.javax.net.ssl.SSLContext

601.javax.net.ssl.SSLException

602.javax.net.ssl.SSLHandshakeException

603.javax.net.ssl.SSLPeerUnverifiedException

604.javax.net.ssl.SSLProtocolException

605.javax.net.ssl.SSLSession

606.javax.net.ssl.SSLSocket

607.javax.net.ssl.TrustManager

608.javax.net.ssl.TrustManagerFactory

609.javax.net.ssl.X509ExtendedKeyManager

610.javax.net.ssl.X509TrustManager

611.javax.security.auth.x500.X500Principal

612.javax.security.cert.CertificateException

613.javax.security.cert.X509Certificate

10. Appendix: Java Library APIs

The Hypori Halo Client uses the following library APIs from the zxing, org.json, and bouncycastle java libraries:

1. Google Protocol Buffers, zxing, yubico, bouncycastle
2. com.google.protobuf.ByteString
3. com.google.protobuf.CodedInputStream
4. com.google.protobuf.CodedOutputStream
5. com.google.protobuf.GeneratedMessageLite
6. com.google.protobuf.InvalidProtocolBufferException
7. com.google.zxing.BarcodeFormat
8. com.google.zxing.BinaryBitmap
9. com.google.zxing.DecodeHintType
10. com.google.zxing.MultiFormatReader
11. com.google.zxing.PlanarYUVLuminanceSource
12. com.google.zxing.ReaderException
13. com.google.zxing.Result
14. com.google.zxing.ResultMetadataType
15. com.google.zxing.ResultPoint
16. com.google.zxing.ResultPointCallback
17. com.google.zxing.client.android.CaptureFragment
18. com.google.zxing.client.android.Contents
19. com.google.zxing.client.android.Intents
20. com.google.zxing.client.android.LocaleManager
21. com.google.zxing.client.android.camera.CameraManager
22. com.google.zxing.client.android.camera.FrontLightMode
23. com.google.zxing.client.android.camera.open.OpenCameraInterface
24. com.google.zxing.client.android.result.ResultHandler
25. com.google.zxing.client.android.result.ResultHandlerFactory
26. com.google.zxing.client.android.wifi.WifiConfigManager
27. com.google.zxing.client.result.AddressBookParsedResult
28. com.google.zxing.client.result.CalendarParsedResult
29. com.google.zxing.client.result.EmailAddressParsedResult
30. com.google.zxing.client.result.ExpandedProductParsedResult
31. com.google.zxing.client.result.GeoParsedResult
32. com.google.zxing.client.result.ISBNParsedResult
33. com.google.zxing.client.result.ParsedResult
34. com.google.zxing.client.result.ParsedResultType
35. com.google.zxing.client.result.ProductParsedResult
36. com.google.zxing.client.result.ResultParser
37. com.google.zxing.client.result.SMSParsedResult
38. com.google.zxing.client.result.TelParsedResult
39. com.google.zxing.client.result.URIParsedResult
40. com.google.zxing.client.result.WifiParsedResult
41. com.google.zxing.common.HybridBinarizer
42. com.yubico.yubikit.YubiKitManager

43. com.yubico.yubikit.apdu.ApduCodeException
44. com.yubico.yubikit.apdu.ApduException
45. com.yubico.yubikit.piv.Algorithm
46. com.yubico.yubikit.piv.InvalidPinException
47. com.yubico.yubikit.piv.PivApplication
48. com.yubico.yubikit.piv.Slot
49. com.yubico.yubikit.transport.usb.UsbConfiguration
50. com.yubico.yubikit.transport.usb.UsbSession
51. com.yubico.yubikit.transport.usb.UsbSessionListener
52. org.bouncycastle.asn1.ASN1InputStream
53. org.bouncycastle.asn1.ASN1ObjectIdentifier
54. org.bouncycastle.asn1.ASN1Primitive
55. org.bouncycastle.asn1.DERIA5String
56. org.bouncycastle.asn1.DEROctetString
57. org.bouncycastle.asn1.x509.AlgorithmIdentifier
58. org.bouncycastle.asn1.x509.DistributionPoint
59. org.bouncycastle.asn1.x509.DistributionPointName
60. org.bouncycastle.asn1.x509.Extension
61. org.bouncycastle.asn1.x509.GeneralName
62. org.bouncycastle.asn1.x509.GeneralNames
63. org.bouncycastle.cert.X509CertificateHolder
64. org.bouncycastle.cert.jcajce.JcaCertStore
65. org.bouncycastle.cms.CMSException
66. org.bouncycastle.cms.CMSProcessableByteArray
67. org.bouncycastle.cms.CMSSignedData
68. org.bouncycastle.cms.CMSSignedDataGenerator
69. org.bouncycastle.cms.CMSTypedData
70. org.bouncycastle.cms.jcajce.JcaSignerInfoGeneratorBuilder
71. org.bouncycastle.operator.ContentSigner
72. org.bouncycastle.operator.DefaultDigestAlgorithmIdentifierFinder
73. org.bouncycastle.operator.DefaultSignatureAlgorithmIdentifierFinder
74. org.bouncycastle.operator.OperatorCreationException
75. org.bouncycastle.operator.jcajce.JcaContentSignerBuilder
76. org.bouncycastle.operator.jcajce.JcaDigestCalculatorProviderBuilder
77. org.bouncycastle.util.Store