Hypori Halo Client (Windows) 4.3 Security Target

Version 1.0
March 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

Hypori and the Hypori logo are registered trademarks of Hypori, Inc. All other trademarks are the property of their respective owners. Hypori provides no warranty with regard to this manual, the software, or other information contained herein, and hereby expressly disclaims any implied warranties of merchantability or fitness for any particular purpose with regard to this manual, the software, or such other information, in no event shall Hypori be liable for any incidental, consequential, or special damages, whether based on tort, contract, or otherwise, arising out of or in connection with this manual, the software, or other information contained herein or the use thereof.

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 Halo Client (Windows) 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: Windows APIs (Section 9).

1.1 Security Target, TOE and CC Identification

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

ST Version – Version 1.0

ST Date – March 15, 2024

TOE Identification – Hypori Halo Client (Windows) 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.

- TD0815: Addition of Conditional TSS Activity for FPT_AEX_EXT.1.5
- TD0780: FIA_X509_EXT.1 Test 4 Clarification
- TD0756: Update for platform-provided full disk encryption
- 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.

- TD0747: Configuration Storage Option for Android
  - The TOE is not installed on an Android platform.
- 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 number in parentheses placed at the end of the component. For example, FDP_ACC.1(1) and FDP_ACC.1(2) indicate that the ST includes two iterations of the FDP_ACC.1 requirement, (1) and (2).
  - 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
- 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.

Table 1 Abbreviations

API	Application Programming Interface
App	Software application
ASLR	Address Space Layout Randomization
CC	Common Criteria
CEM	Common Evaluation Methodology
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
UWP	Universal Windows Platform

2. TOE Description

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

2.1 Product Overview

The TOE is the Hypori Halo Client (Windows) 4.3. The TOE is a software program as specified in the [PP_APP_v1.4] which is a Windows-based application that installs on an end user's desktop, Surface tablet, or laptop that runs Windows 10 or 11 and communicates only with the Hypori Virtual Device on the server through platform provided secure TLS 1.2 encrypted protocols.

In the Hypori platform, end users running a Hypori Halo Client (Windows) on their desktop, Surface tablet, or laptop 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 Halo Client (Windows). The Hypori Halo Client (Windows) application provides secure access to the remote Android virtual device and brokers access between the user's desktop, Surface tablet, or laptop 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 Windows executing on the physical device.

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

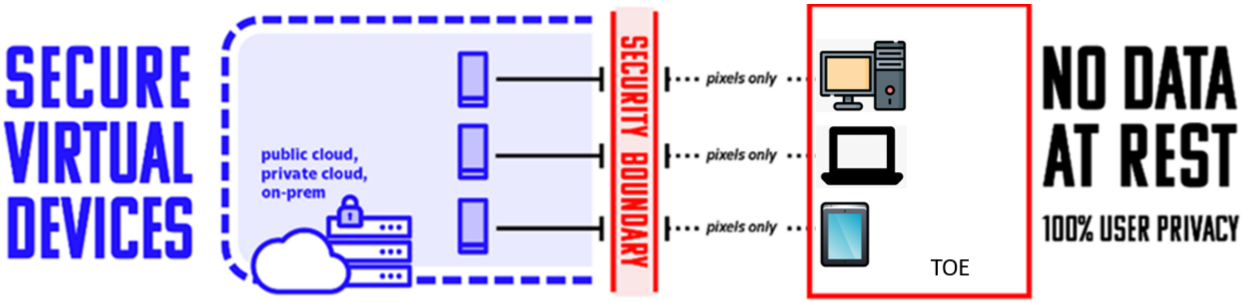

Figure 1 Hypori Halo Client (Windows) User Data Flow

The user's physical device is a "window" to their virtualized device residing in the cloud or on-premises. The Hypori Halo application captures touch and sensor data from any Windows desktop, Surface tablet, or laptop. 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 applications and data via a separate, secure virtual device from a Windows desktop, Surface tablet, or laptop. 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.

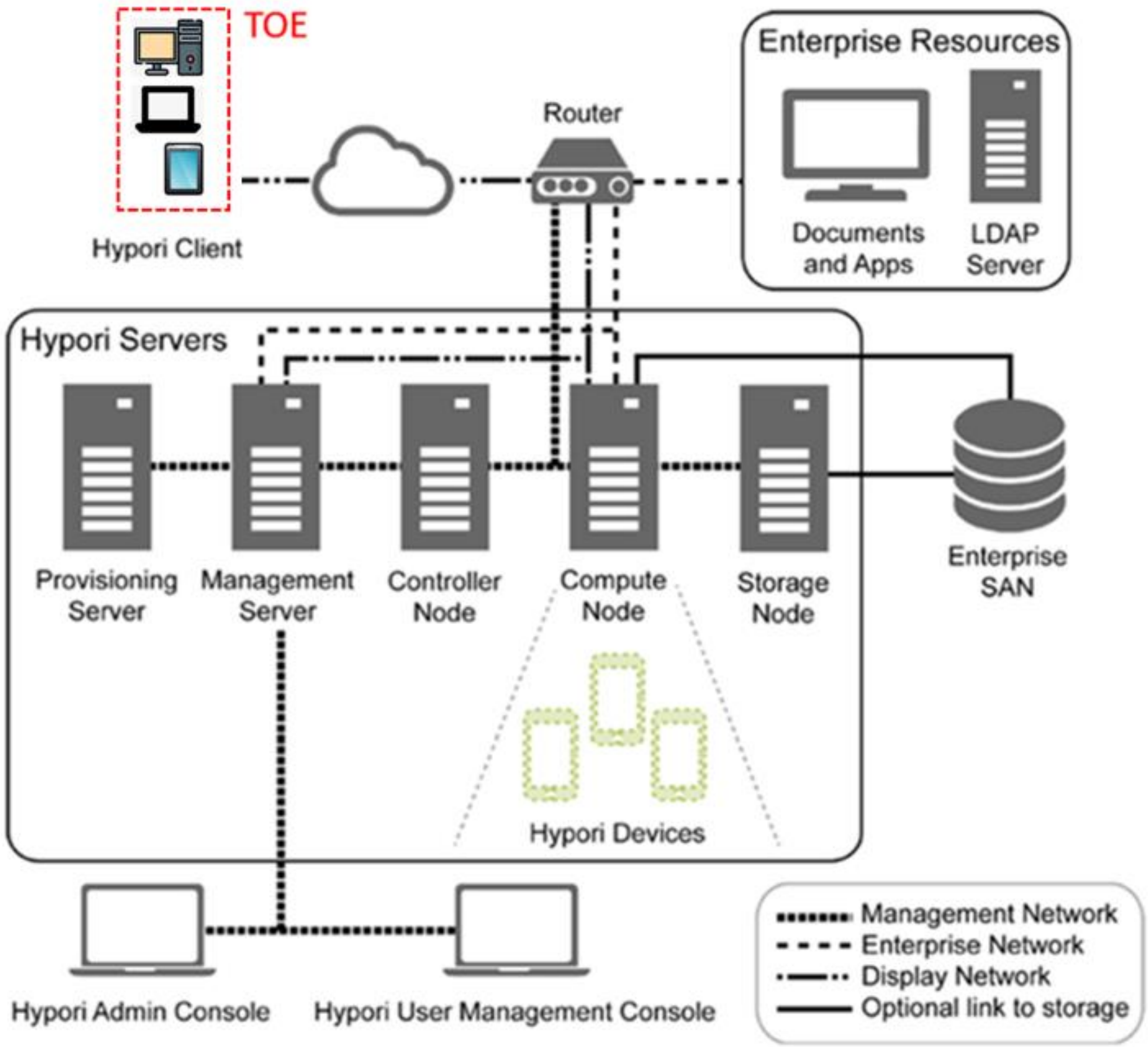

Figure 2 Hypori Solution

The Hypori VMI platform includes the following components:

- **Hypori Halo Client:** This is a Windows-based application that installs on the end user's Windows desktop, Surface tablet, or laptop 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 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 Windows-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.

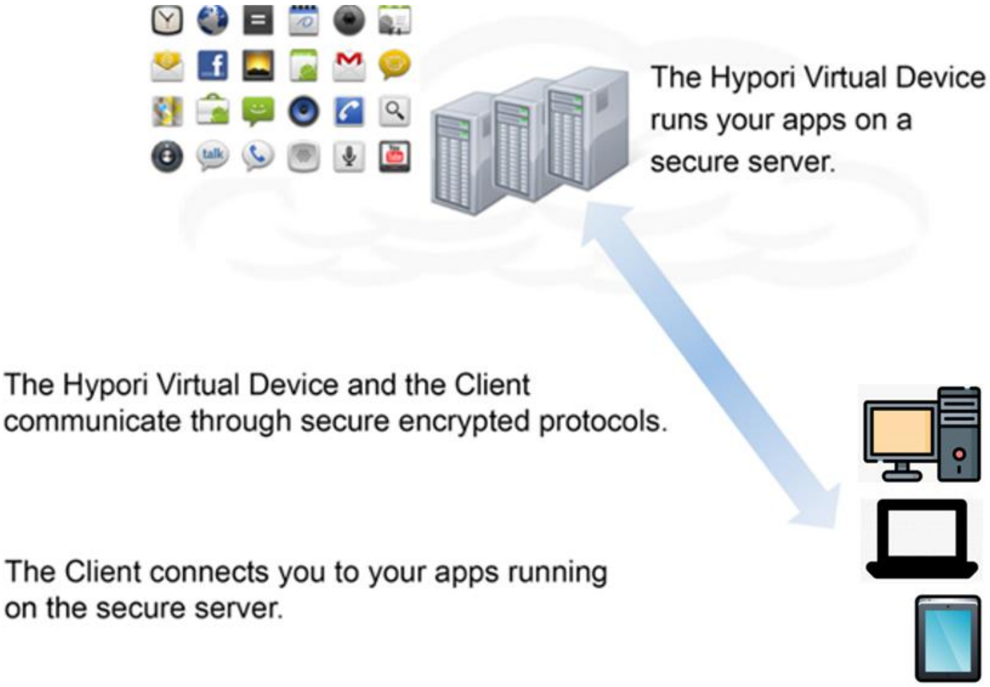
The Hypori Virtual Device runs your apps on a secure server.
The Hypori Virtual Device and the Client communicate through secure encrypted protocols.
The Client connects you to your apps running on the secure server.

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.

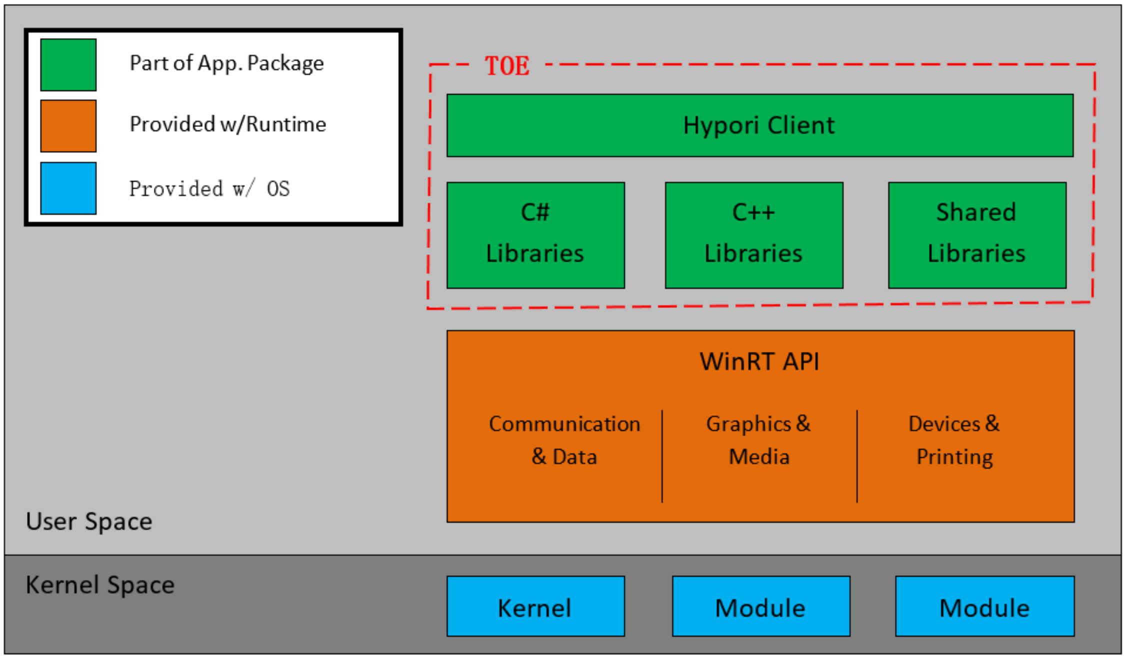

Figure 4 TOE Boundary for Windows Devices

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 Windows Store. The Hypori Halo Client is a Windows-based application that only communicates with the Hypori Virtual Device on the Hypori Server. The Hypori Virtual Device, applications running on the Hypori server, the Windows desktop, Surface tablet, or laptop hardware, and any functions not specified in this security target are outside the scope of the TOE.

2.3.1.1 Software Requirements

The TOE runs on Microsoft Windows 10 and 11 operating systems.

2.3.1.2 Hardware Requirements

The TOE imposes no hardware requirements beyond the Microsoft Windows 10 and 11 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 Hypori server using TLS and cryptographic services provided by the Windows platform. The TOE stores certificates for mutual authentication in the Windows Certificate Store.

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 Hypori server using the TOE. In general, sensitive data resides on the Hypori server and not the Hypori Halo 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, 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 Halo Client configuration options and applying configuration policies from the Hypori Server. The TOE uses the platform's mechanisms for storing the configuration settings.

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 a Windows application running on a desktop, Surface tablet, or laptop. 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 TOE (Windows).

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, October 7, 2021 [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 SFRs. Since these SFRs 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
- FPT_IDV_EXT.1 Software Identification and Versions
- FTP_DIT_EXT.1 Protection of Data in Transit

5.2 TOE Security Functional Requirements

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

Table 2: 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 cryptographic 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 meet the following: RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 8017, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1",***
- ***[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,***
- ***Bluetooth GATT,***
- ***Bluetooth RFComm,***
- ***[Graphics Capture***
- ***Private Network usage]***

].

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]***

].

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 and 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 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 extendedKeyUsage 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 extendedKeyUsage 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 extendedKeyUsage 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 extendedKeyUsage 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 extendedKeyUsage field.[4]

FIA_X509_EXT.1.2 The application shall only treat a certificate as a CA certificate 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***].

[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 Security Management (FMT)

5.2.4.1 Secure by Default Configuration (FMT_CFG_EXT.1)

FMT_CFG_EXT.1.1 The application shall only provide 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.

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***].

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

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 Use of Supported Services and APIs (FPT_API_EXT.1)

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

5.2.6.2 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.

5.2.6.3 Software Identification and Versions (FPT_IDV_EXT.1)

FPT_IDV_EXT.1.1 The application shall be versioned with [*[**Microsoft's standards for packaging version numbering (Major, Minor, Maintenance Release, and a fourth number controlled by the Microsoft Store***]].

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 [

- **Microsoft .NET Framework v4.8.03752**
- **Universal Windows Apps v15.0.28307.1000**
- **Google Protobuf v3.21.12**
- **gRPC v1.51.1**
- **OpenSSL 3.1.1.0**
- **Microsoft.UI.XAML v2.3.191211002**
- **Microsoft.Toolkit.Uwp v5.1.1**
- **ZXing.Net v0.16.5**
- **Microsoft.ApplicationInsights v1.2.3**
- **Newtonsoft.Json v12.0.2**
- **TimeZoneConverter v3.2.0**
- **Opus-Windows v1.1.5**
- **MetroLog v1.0.1**
- **Microsoft.NETCore.UniversalPlatform v6.2.9**
- **libyuv-windows v1.0.1671**
- **ANGLE.WindowsStore v2.1.14**

].

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 The application installation package and its updates shall be digitally signed such that its 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 [

[5] Modified per TD0743.

- 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 3 are included in this ST by reference from the [PP_APP_v1.4].

Table 3: 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
	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
- Security management
- Privacy
- Protection of the TSF
- Trusted path/channels

6.1 Cryptographic support

The Hypori Halo 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 the RSA or Elliptic Curve key pairs 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 Windows 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 platforms:

- Surface Laptop Go 2 Core i5-1135G7 with Windows 11 Pro
- Surface Pro 7+ Core i5-1135G7 (Wi-Fi) with Windows 10 Enterprise Version 21H2

Surface Laptop Go 2 Core i5-1135G7 with Windows 11 Pro

The Common Criteria evaluated version of Windows 10 and Windows 11 is identified on the NIAP Product Compliant List with a reference number 2022-21-INF-3955- v1.

https://www.niap-ccevs.org/Product/CompliantCC.cfm?CCID=2023.1010

Table 4: Windows 11 Platform

Device Name	Chipset Vendor	Processor	Architecture
Surface Go 2	Intel	Core i5-1135G7	Microsoft Windows 11 version (64-bit)

CAVP Certificate A2645

Table 5: Windows 11 CAVP Certificates

SFR	Algorithm	NIST Standard	CAVP Certificate
FTP_DIT_EXT.1	AES CBC encrypt, decrypt 128-bit, 256-bit	FIPS 197 SP800-38A	A2645
FTP_DIT_EXT.1	AES GCM Encrypt, decrypt 128-bit, 256-bit	FIPS 197 SP 800-38D GCM mode	A2645

SFR	Algorithm	NIST Standard	CAVP Certificate
FCS_CKM.1/AK FTP_DIT_EXT.1	ECDSA KeyGen P-256, P-384, P-521	FIPS186-4	A2645
FCS_CKM.1/AK FTP_DIT_EXT.1	RSA KeyGen Modulo 2048, 3072	FIPS 186-4	A2645
FTP_DIT_EXT.1	ECDSA SigVer P-256, P-384, P-521	FIPS186-4	A2645
FTP_DIT_EXT.1	HMAC-SHA-1 MAC: 80-160 Increment 8 Key Length: 8-2048 Increment 8	FIPS 198	A2645
FTP_DIT_EXT.1	HMAC-SHA2-256 MAC: 80-160 Increment 8 Key Length: 8-2048 Increment 8	FIPS 198	A2645
FTP_DIT_EXT.1	HMAC-SHA2-384 MAC: 80-384 Increment 8 Key Length: 8-2048 Increment 8	FIPS 198	A2645
FTP_DIT_EXT.1	HMAC-SHA2-512 MAC: 80-512 Increment 8 Key Length: 8-2048 Increment 8	FIPS 198	A2645
FCS_CKM.2 FTP_DIT_EXT.1	KAS-ECC P-256	SP 800-56A	A2645
FCS_CKM.2 FTP_DIT_EXT.1	RSA key establishment	RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 8017	None
FTP_DIT_EXT.1	RSA SigVer Modulo 2048, 3072	FIPS 186-4	A2645
FTP_DIT_EXT.1	SHA-1 Message Length: 0-51200 Increment 8	FIPS 180-4	A2645
FTP_DIT_EXT.1	SHA-256 Message Length: 0-51200 Increment 8	FIPS 180-4	A2645
FTP_DIT_EXT.1	SHA-384 Message Length: 0-65536 Increment 8	FIPS 180-4	A2645

SFR	Algorithm	NIST Standard	CAVP Certificate
FTP_DIT_EXT.1	SHA-512 Message Length: 0-65536 Increment 8	FIPS 180-4	A2645
FTP_DIT_EXT.1	Counter DRBG AES-256	FIPS 197 SP 800-80A	A2645

The Common Criteria evaluated version of Windows 10 and Windows 11 is identified on the NIAP Product Compliant List with a reference number 2022-21-INF-3955- v1.

https://www.niap-ccevs.org/Product/CompliantCC.cfm?CCID=2023.1010

Surface Pro 7+ Core i5-1135G7 (Wi-Fi) with Windows 10 Enterprise Version 21H2

Table 6: Windows 10 Platform

Device Name	Chipset Vendor	Processor	Architecture
Surface Pro 7+	Intel	Core i5-1135G7	Microsoft Windows 10 version (64-bit)

CAVP Certificate 2677

Table 7: Windows 10 CAVP Certificates

SFR	Algorithm	NIST Standard	CAVP Certificate
FTP_DIT_EXT.1	AES CBC encrypt, decrypt 128-bit, 256-bit	FIPS 197 SP800-38A	A2677
FTP_DIT_EXT.1	AES GCM Encrypt, decrypt 128-bit, 256-bit	FIPS 197 SP 800-38D GCM mode	A2677
FCS_CKM.1/AK FTP_DIT_EXT.1	ECDSA KeyGen P-256, P-384, P-521	FIPS186-4	A2677
FCS_CKM.1/AK FTP_DIT_EXT.1	RSA KeyGen Modulo 2048, 3072	FIPS 186-4	A2677
FTP_DIT_EXT.1	ECDSA SigVer P-256, P-384, P-521	FIPS186-4	A2677
FTP_DIT_EXT.1	HMAC-SHA-1 MAC: 80-160 Increment 8 Key Length: 8-2048 Increment 8	FIPS 198	A2677
FTP_DIT_EXT.1	HMAC-SHA2-256 MAC: 80-160 Increment 8 Key Length: 8-2048 Increment 8	FIPS 198	A2677

SFR	Algorithm	NIST Standard	CAVP Certificate
FTP_DIT_EXT.1	HMAC-SHA2-384 MAC: 80-384 Increment 8 Key Length: 8-2048 Increment 8	FIPS 198	A2677
FTP_DIT_EXT.1	HMAC-SHA2-512 MAC: 80-512 Increment 8 Key Length: 8-2048 Increment 8	FIPS 198	A2677
FCS_CKM.2 FTP_DIT_EXT.1	KAS-ECC P-256	SP 800-56A	A2677
FCS_CKM.2 FTP_DIT_EXT.1	RSA key establishment	RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 8017	None
FTP_DIT_EXT.1	RSA SigVer Modulo 2048, 3072	FIPS 186-4	A2677
FTP_DIT_EXT.1	SHA-1 Message Length: 0-51200 Increment 8	FIPS 180-4	A2677
FTP_DIT_EXT.1	SHA-256 Message Length: 0-51200 Increment 8	FIPS 180-4	A2677
FTP_DIT_EXT.1	SHA-384 Message Length: 0-65536 Increment 8	FIPS 180-4	A2677
FTP_DIT_EXT.1	SHA-512 Message Length: 0-65536 Increment 8	FIPS 180-4	A2677
FTP_DIT_EXT.1	Counter DRBG AES-256	FIPS 197 SP 800-80A	A2677

For elliptic curve cipher suites, the Hypori Halo Client relies on the platform for elliptic curves. The Windows platform supports NIST curves secp256r1, secp384r1, and secp521r1 and Supported Elliptic Curves Extension for TLS. No configuration is required by a Hypori Halo Client user.

6.1.1 FCS_CKM_EXT.1

The Hypori Halo Client invokes the platform to generate cryptographic keys. The Hypori Halo Client relies on the platform for TLS support. The platform generates all ephemeral TLS keys without direct Hypori Halo Client action.

6.1.2 FCS_CKM.1/AK

The TOE invokes the platform to generate asymmetric cryptographic keys for the secure communication to the Hypori Virtual Device on the Hypori Server. The Windows platform generates the P-256, P-384, P-521 Elliptic Curve keys and the RSA 2048-bit and 3072-bit key sizes.

The Windows platforms call the Windows SymCrypt Cryptographic library for the platform to create the ECC and RSA keys. Description of the Windows SymCrypt Cryptographic library and API calls can be found at: https://github.com/microsoft/SymCrypt.

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 Hypori 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 Halo Client itself uses no DRBG functions.

6.1.5 FCS_STO_EXT.1

Table 8 lists each Hypori 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	The RSA/Elliptic Curve key pairs and the X509 certificate are used for TLS mutual authentication (client side of TLS exchange) that is implemented by the platform.	Windows Certificate Store

6.2 User data protection

The Hypori Halo Client uses the platform's permission mechanisms to inform the user of hardware and software resources the client accesses. The client presents the required permissions to the user for approval during installation. A user initiates network connections to the Hypori Virtual Device on the Hypori server. In general, sensitive data resides on the Hypori server and is not stored on the Hypori Halo Client. Sensitive data on the Hypori Halo Client is limited to credentials, which the client stores as described in section 6.1. The client does not maintain Personally Identifiable Information (PII).

6.2.1 FDP_DAR_EXT.1

Hypori Halo Client sensitive data consists of user TLS client private key credentials. FCS_STO_EXT.1 Storage of Secrets specifies the platform's Windows Certificate Store for protecting keys.

6.2.2 FDP_DEC_EXT.1

At first launch, the Hypori Halo Client presents to the user some of the permissions requested by the application that are needed to operate. A user can accept (or reject) the permissions, but rejecting permissions may cause apps on the Hypori Virtual Device to not function properly. Some permissions are requested by the application only as the feature is required on first use (such as microphone input). Table 9 shows the permissions required by the Hypori Halo Client. Those marked with an '*' are prompted for when the Hypori Halo Client is started for the first time.

Table 9 Permissions Required by the Hypori Halo Client

Permission	Description
Internet Connectivity	Open network sockets.
Bluetooth	Connect to paired Bluetooth devices.
Bluetooth GATT	Bluetooth generic attribute profile capabilities.

Permission	Description
Bluetooth RFComm	Bluetooth data transport via serial port, file transfer.
Graphics Capture	Enables the user to take screen captures when connected to the Virtual Device.
Location	Access precise location.
Microphone *	Provides access to the microphone and audio recording capabilities on the Windows desktop, Surface tablet, or laptop hardware to support apps in the Virtual Device that require audio input.
Private Network Usage	Used to access Intranet networks that have an authenticated domain controller, or that the user has designated as either home or work networks.
WiFi Control	Used to access the devices WiFi status, including signal strength.
Camera *	The Hypori Halo Client uses remote access to the device's camera to support multimedia applications that use the camera in the Virtual Device. It can also use the camera when scanning a QR code during account provisioning.

Updates to the Hypori Halo Client may automatically add additional capabilities within each group. 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 Hypori Virtual Device on the Hypori server by starting the Hypori Halo Client and entering account information. After the Hypori Halo Client connects to the Hypori Virtual Device on the Hypori server, the applications the user accesses run on the Hypori Device in the Hypori server, not on the Windows desktop, Surface tablet, or laptop. The Hypori Halo Client does not listen on any ports for inbound connection requests. The Hypori Halo Client interacts only with the Hypori Virtual Device on the Hypori server. When a Hypori Device application needs information from a server (such as a map server), the Hypori Virtual Device – not the Hypori Halo Client – communicates with the server (which may be an internal, enterprise server).

The TOE does not access any sensitive information repositories as defined by the [PP_APP_v1.4].

The Hypori Halo Client does not maintain PII. Hence, it does not transmit PII over any network.

6.2.3 FDP_NET_EXT.1

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

6.3 Identification and authentication

The Windows platform follows RFC 5280 for certificate path validation. The Hypori Halo Client uses Windows certificate validation services to authenticate the X.509 certificate the Hypori server presents as part of the establishing a TLS connection.

6.3.1 FIA_X509_EXT.1

The TOE uses X.509v3 certificates as defined by RFC 5280 to support authentication for TLS connections. All certificate validation is performed by the underlying Windows platform, and certificates are stored in the Windows Certificate Store. Certificate validation is done in conformance to RFC 5280. Certificate validation paths must terminate with a trusted CA certificate that contains the basicConstraints extension and a CA flag that is set to TRUE. ExtendedkeyUsage field validation is also performed. Checking is also done for the basicConstraints extension and the cA flag to determine whether they are present and set to TRUE. If they are not, the certificate is not accepted. The Windows platform validates the revocation status of the certificate using the Online Certificate Status Protocol (OCSP) as specified in RFC 6960. When the platform cannot establish a connection to an OCSP server providing the status to determine certificate validity, the platform will reject the connection.

Certificates must have a valid and established chain of trust by verifying the root certificate and are verified using the Windows.Security.Cryptography.Certificates, System.Security.Cryptography, and System.Security.Cryptography. X509 Certificate Namespaces. The TOE utilizes the Authority Information Access caIssuers field to build certificate paths. This field must be present and it must point to a valid CA Distribution Point for the chain to be successfully verified.

The following URLs provide a description of the Microsoft certificate path validation algorithm:

https://docs.microsoft.com/en-us/uwp/api/windows.security.cryptography.certificates?view=winrt-19041

https://learn.microsoft.com/en-us/dotnet/api/system.security.cryptography?view=net-7.0&viewFallbackFrom=dotnet-uwp-10.0

https://learn.microsoft.com/en-us/dotnet/api/system.security.cryptography.x509certificates?view=net-7.0&viewFallbackFrom=dotnet-uwp-10.0

The Hypori Halo Client relies on the platform for TLS services and package updates. Hence, the platform checks extended key usage for Server Authentication, Client Authentication, and Code Signing purposes. The Hypori Halo Client does not perform email encryption, email signature verification, and Enrollment over Secure Transport. Consequently, no check is made for extended key usage Email Protection or CMC Registration Authority purposes.

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 the Windows Certificate Store. The user's credentials are stored 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 Windows devices, the Hypori Halo Client uses Windows platform certificate path validation services with the CA certificate to validate the certificate presented by the Hypori server. The Hypori Halo Client will send a status request to an OCSP responder and receive information if the certificate is valid or revoked. A good response will indicate the certificate is valid and not revoked. A revoked status will indicate the certificate has been revoked. If the OCSP responder fails to respond or there is an error, the Hypori Halo Client will not accept the certificate (invalid) and not establish the connection.

6.4 Security management

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

6.4.1 FMT_CFG_EXT.1

Hypori Halo Client credentials consist of user the TLS private client key. The Hypori Halo Client installer does not include a default client private 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 installing the TLS client certificate and private key.

The binaries are stored in a protected location under %ProgramFiles%\WindowsApps.

The application is configured by default with file permissions which protect the application's binaries and data files from modification by normal unprivileged users. The binaries are stored in a protected location under %ProgramFiles%\WindowsApps. See the following link for additional details. https://learn.microsoft.com/en-us/windows/application-management/overview-windows-apps

6.4.2 FMT_MEC_EXT.1

The Hypori Halo Client invokes the recommended Windows mechanisms for storing account settings files. Accounts are stored using Microsoft's preferred application settings method, Windows.Storage.ApplicationData.Current.LocalSettings class https://docs.microsoft.com/en-us/windows/uwp/design/app-settings/store-and-retrieve-app-data.

The account settings (or options) consists of the Hypori Server hostname (URL), port number of the Hypori Server, Account Name, and the email address.

The namespace used is Windows.Storage. The primary class to manage this data is the ApplicationData.Current.LocalSettings. This class will persist settings (LocalSettings is a dictionary like mechanism using key/value pairs) to a system managed folder created at the time the application is installed. This same data will be removed when the application is uninstalled. The system is responsible for maintaining the physical storage and insures that the data is isolated from other apps and users. The system preserves this data when the app is updated.

Policies are stored on the hard drive in isolated storage (i.e. it's by user and by app). Only users with admin rights to the device may get to this data, or the UWP app itself. UWP apps, defined by system rules, are sandboxed, and can only affect app specific data for the current windows user.

Example location:

C:\Users\<windows user>\AppData\Local\Packages\IntelligentWavesLLC.Hypori-Client_xcxc9e4h3hfva\LocalState\policies

The binaries are stored in a protected location under %ProgramFiles%\WindowsApps.

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

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 Windows .NET UWP applications, and stack-based buffer overflow protection. The client leverages Windows UWP 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

The Hypori Halo Client handles ASLR in its C++ libraries by setting the linker option "/DYNAMICBASE". The C# code is enabled by default, so the client code is ASLR compliant. The application does not allocate any memory region with both write and execute permissions nor does the TOE request to map memory to an explicit address. The TOE does not write user-modifiable files to directories that contain executable files. The application is built with stack-based buffer overflow protection enabled.

The TOE is compatible with security features provided by the platform vendor. The TOE OS platform supports Windows Defender Exploit Guard.

6.6.2 FPT_API_EXT.1

The Hypori Halo Client uses the Windows APIs listed in section 9 Appendix: Windows APIs.

6.6.3 FPT_IDV_EXT.1

The Hypori Halo Client Windows application uses the major.minor.maintenance release format. With the exception that the Windows Store reserves a last integer based number for itself (major.minor.maintenance.internalwindowsuseonly) (ex. 4.3.1.0). The Windows App versioning is integer based, so Hypori doesn't use leading zeros (ex. 4.3.00001).

The build fingerprint displayed on the settings page (ex. xxxxxx-yyyyyyyyyy).

A description of Microsoft's standards for package version numbering by the Microsoft Store is available at:

https://docs.microsoft.com/en-us/windows/uwp/publish/package-version-numbering?redirectedfrom=MSDN.

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 Halo Client Accounts screen, select the ellipses menu and click on 'About'. The About screen will display the version number, build information and copyright.

6.6.4 FPT_LIB_EXT.1

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

- Microsoft .NET Framework v4.8.03752
- Universal Windows Apps v15.0.28307.1000
- Google Protobuf v3.21.12
- gRPC v1.51.1
- OpenSSL 3.1.1.0

- Microsoft.UI.XAML v2.3.191211002
- Microsoft.Toolkit.Uwp v5.1.1
- ZXing.Net v0.16.5
- Microsoft.ApplicationInsights v1.2.3
- Newtonsoft.Json v12.0.2
- TimeZoneConverter v3.2.0
- Opus-Windows v1.1.5
- MetroLog v1.0.1
- Microsoft.NETCore.UniversalPlatform v6.2.9
- libyuv-windows v1.0.1671
- ANGLE.WindowsStore v2.1.14

6.6.5 FPT_TUD_EXT.1, FPT_TUD_EXT.2

Hypori, Inc., distributes the Hypori Halo Client as a Windows standard .appx file for Windows devices.

The TOE relies on the Windows Store to provide application updates. Updates are automatically handled by the Windows Operating System, so notifications will be given to the user about existing application updates. Hypori digitally signs the installation package as well as updates and includes the corresponding public key certificate in the package. Windows 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. The client is signed with a unique certificate. It can be delivered via the Windows Store or the enterprise IT group of the user.

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 Virtual Device on the 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 Virtual Device on the Hypori server, the Hypori Halo Client connects to the server using TLS 1.2 provided by the Windows platform. The Windows platforms call the Windows SymCrypt Cryptographic library for the platform to create the ECC and RSA keys. Description of the Windows SymCrypt Cryptographic library and API calls can be found at: https://github.com/microsoft/SymCrypt.

The Hypori Halo Client will open the socket, designate the host, port, and socket protection level (TLS 1.2). The following Microsoft platform API invokes this functionality:

```
await      _socket.ConnectAsync(new_HostName(_serverAddress),      _serverPort.ToString(),
SocketProtectionLevel.Tls12).
```

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 Microsoft 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 maintins a Support 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 TLS 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.2						X	
FTP_DIT_EXT.1							X

9. Appendix: Windows APIs

The Hypori Halo Client uses the following Windows APIs:

1. Microsoft.Build.Framework
2. Microsoft.Build.Utilities
3. Microsoft.Graphics.Canvas.Brushes
4. Microsoft.Graphics.Canvas.UI.Xaml
5. Microsoft.Graphics.Canvas.UI
6. Microsoft.Graphics.Canvas
7. Microsoft.Toolkit.Uwp.Connectivity
8. Microsoft.VisualStudio.TestPlatform.UnitTestFramework
9. Platform
10. System.Collections.Concurrent
11. System.Collections
12. System.ComponentModel
13. System.Diagnostics
14. System.Drawing.Color
15. System.Globalization
16. System.IO.Compression
17. System.IO
18. System.Linq
19. System.Net.Http
20. System.Net.Security
21. System.Net.Sockets
22. System.Net
23. System.Reflection
24. System.Resources
25. System.Runtime.CompilerServices
26. System.Runtime.InteropServices.WindowsRuntime
27. System.Runtime.InteropServices
28. System.Runtime.Serialization.Formatters
29. System.Runtime.Serialization.Json
30. System.Runtime.Serialization
31. System.Security.Authentication
32. System.Security.Cryptography.X509Certificates
33. System.Security.Cryptography
34. System.Text.RegularExpressions
35. System.Text
36. System.Threading.Tasks
37. System.Threading
38. System.Timers
39. System.Windows.Input
40. System.Xml
41. System
42. Windows.ApplicationModel.Activation

43. Windows.ApplicationModel.Background
44. Windows.ApplicationModel.Core
45. Windows.ApplicationModel.DataTransfer
46. Windows.ApplicationModel.Resources
47. Windows.ApplicationModel
48. Windows.Data.Json
49. Windows.Devices.Geolocation
50. Windows.Devices.Input
51. Windows.Devices.Sensors
52. Windows.Devices.SmartCards
53. Windows.Devices.WiFi
54. Windows.Foundation.Collections
55. Windows.Foundation.Diagnostics
56. Windows.Foundation
57. Windows.Graphics.Display
58. Windows.Media.Audio
59. Windows.Media.Capture.Frames
60. Windows.Media.Capture
61. Windows.Media.Devices
62. Windows.Media.MediaProperties
63. Windows.Media.Render
64. Windows.Media
65. Windows.Networking.Connectivity
66. Windows.Networking.PushNotifications
67. Windows.Networking.Sockets
68. Windows.Networking
69. Windows.Security.Credentials
70. Windows.Security.Cryptography.Certificates
71. Windows.Security.Cryptography.Core
72. Windows.Security.Cryptography.DataProtection
73. Windows.Security.Cryptography
74. Windows.Security.ExchangeActiveSyncProvisioning
75. Windows.Storage.Pickers
76. Windows.Storage.Streams
77. Windows.Storage
78. Windows.System.Power
79. Windows.System.Profile
80. Windows.System.Threading
81. Windows.System
82. Windows.UI.Color
83. Windows.UI.Core
84. Windows.UI.Input
85. Windows.UI.Notifications
86. Windows.UI.Popups

87. Windows.UI.Text.Core
88. Windows.UI.ViewManagement
89. Windows.UI.Xaml.Controls.Primitives
90. Windows.UI.Xaml.Controls
91. Windows.UI.Xaml.Data
92. Windows.UI.Xaml.Input
93. Windows.UI.Xaml.Media.Imaging
94. Windows.UI.Xaml.Media
95. Windows.UI.Xaml.Navigation
96. Windows.UI.Xaml
97. Windows.Web.Http.Headers
98. Windows.Web.Http
99. Windows::Foundation::Collections
100. Windows::Foundation
101. Windows::System::Diagnostics
102. Windows::UI::Xaml::Controls