VMware Horizon Client 8 2209 (Horizon 8.7)

Security Target

Version 1.0

04 April 2023

Prepared for:

VMware, Inc.
3401 Hillview Avenue
Palo Alto, CA 94304

Prepared by:

Accredited Testing and Evaluation Labs
6841 Benjamin Franklin Drive
Columbia, MD 21046

Contents

Tables

1 Security Target Introduction

The Security Target (ST) contains the following additional sections:

- Product and TOE Description (Section 2)
- Security Problem Definition (Section 3)
- Security Objectives (Section 4)
- IT Security Requirements (Section 5)
-

- TOE Summary Specification (Section 0)
- Protection Profile Claims (Section 7)
- Rationale (Section 8)
- A TOE Usage of Third-Party Components (Appendix 0)

1.1 Security Target, TOE and CC Identification

ST Title – VMware Horizon Client 8 2209 (Horizon 8.7) Security Target

ST Version – Version 1.0

ST Date – 04 April 2023

TOE Identification – VMware Horizon Client 8 version 2209 (Horizon 8.7)

TOE Developer – VMware, Inc.

Evaluation Sponsor – VMware, Inc.

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

1.2 Conformance Claims

This ST and the TOE it describes are conformant to the following CC specifications:

- *Protection Profile for Application Software, Version 1.4, 07 October 2021* (App PP) with the following optional and selection-based SFRs:
  - FCS_CKM.1/AK
  - FCS_CKM.1/SK
  - FCS_CKM.2
  - FCS_COP.1/SKC
  - FCS_COP.1/Hash
  - FCS_COP.1/Sig
  - FCS_COP.1/KeyedHash
  - FCS_HTTPS_EXT.1/Client
  - FCS_RBG_EXT.2
  - FIA_X509_EXT.1
  - FIA_X509_EXT.2
- *Functional Package for Transport Layer Security (TLS), Version 1.1, February 12, 2019* (TLS Package) with the following optional and selection-based SFRs:
  - FCS_TLSC_EXT.1
  - FCS_TLSC_EXT.2
  - FCS_TLSC_EXT.3
  - FCS_TLSC_EXT.5
- The following NIAP Technical Decisions apply to the TOE and have been accounted for in the ST development and the conduct of the evaluation, or were considered to be non-applicable:

  TD0442: Updated TLS Ciphersuites for TLS Package

- This TD is applicable to the TOE.

TD0469: Modification of test activity for FCS_TLSS_EXT.1.1 test 4.1

- This TD is not applicable to the TOE because it applies to an SFR the TOE does not claim.

TD0499: Testing with pinned certificates

- This TD is applicable to the TOE because it affects an SFR that the TOE claims. However, the TOE does not support certificate pinning so the TD's modification to the testing for this does not affect the claims made for the TSF.

TD0513: CA Certificate loading

- This TD is applicable to the TOE but applies specifically to test activities so the ST itself is unaffected.

TD0588: Session Resumption Support in TLS package

- This TD is not applicable to the TOE because it applies to an SFR that the TOE does not claim.

TD0624: Addition of DataStore for Storing and Setting Configuration Options

- This TD is applicable to the TOE.

TD0628: Addition of Container Image to Package Format

- This TD is not applicable to the TOE because it is not packaged in a container format.

TD0650: Conformance claim sections updated to allow for MOD_VPNC_V2.3 and 2.4

- This TD is not applicable to the TOE because it does not claim VPN client functionality.

TD0655: Mutual authentication in FTP_DIT_EXT.1 for SW App

- This TD is applicable to the TOE.

TD0664: Testing activity for FPT_TUD_EXT.2.2

- This TD is applicable to the TOE.

TD0669: FIA_X509_EXT.1 Test 4 Interpretation

- This TD is applicable to the TOE.

TD0709: Number of elements for iterations of FCS_HTTPS_EXT.1

- This TD is not applicable to the TOE; it only affects FCS_HTTPS_EXT.1/Server, which the TOE does not claim.

TD0717: Format changes for PP_APP_V1.4

- This TD is applicable to the TOE.

TD0719: ECD for PP APP V1.3 and 1.4

  - This TD is not applicable to the TOE; this TD updates the App PP to include a formal ECD which is needed for the PP itself to conform to CC Part 3. This does not change the ST or how the evaluation of the TOE is conducted.

  TD0726: Corrections to (D)TLSS SFRs in TLS 1.1 FP

  - This TD is not applicable to the TOE. The ST does not claim the SFRs that are affected by the TD.

- 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. An iterated SFR is indicated by a slash followed by a descriptor for the purpose of the iteration. For example, FCS_HTTPS_EXT.1/Client indicates that the FCS_HTTPS_EXT.1 requirement applies specifically to HTTPS client functionality.
  - Assignment: allows the specification of an identified parameter. Assignments are indicated using italics and are surrounded by brackets (e.g., [*assignment item*]). Note that an assignment within a selection would be identified in both italics and underline, with the brackets themselves underlined since they are explicitly part of the selection text, unlike the brackets around the selection itself (e.g., [<u>selection item, [*assignment item inside selection*]</u>]).
  - Selection: allows the specification of one or more elements from a list. Selections are indicated using underlines and are surrounded by brackets (e.g., [<u>selection item</u>]).
  - Refinement: allows technical changes to a requirement to make it more restrictive and allows non-technical changes to grammar and formatting. Refinements are indicated using bold, for additions, and strike-through, for deletions (e.g., "… **all** objects …" or "… ~~some~~ **big** things …"). Note that minor grammatical changes that do not involve the addition or removal of entire words (e.g., for consistency of quantity such as changing "meets" to "meet") do not have formatting applied.
- Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions.
- The ST does not show selection/assignment operations that have been completed by the PP authors, though it does preserve brackets to show where such operations have been made.
- The ST does not show refinement operations that have been completed by the PP authors; refinements are used only to show where the ST author has refined text from the PP.

1.3.1 Terminology

The following terms and abbreviations are used in this ST:

Table 1: Terms and Definitions

Term	Definition
Agent	A Horizon component that acts as an endpoint on a protected resource and serves content on that resource (individual applications or an interactive desktop session) to an authorized Horizon Client.
Blast	A communications protocol that is used to transmit interactive desktop and application sessions (user inputs and audio/visual outputs).
Client	A Horizon component that resides on an end user device that the user can run to access enterprise computing resources via the virtual desktop.
Cloud Pod	A self-contained Horizon deployment on a particular network. Multiple cloud pods can be federated, allowing a client on one pod to access resources on another.
Connection Server	A Horizon component that is responsible for determining the authorizations of a Horizon Client user and facilitating the establishment of Agent communications so that authorized resources can be served to that user.
Horizon	A collection of products that are used to allow an organizational user to access shared enterprise resources in a protected network from a single client application.
Unified Access Gateway	A network device that acts as a proxy between a Horizon Client on an unprotected network and other Horizon components on a protected internal network. The Unified Access Gateway is responsible for authenticating Horizon Client users and passing their validated identity to a Connection Server via SAML assertion. It is also responsible for establishing Horizon Agent connectivity on behalf of the client.
Virtual Desktop	The virtual desktop is the set of enterprise computing resources that are served to a user within an interactive Horizon Client session. For the purpose of the TSF, the important consideration is that all virtual desktop content is transmitted over TLS.

1.3.2 Acronyms

Table 2: Acronyms

Term	Definition
ASLR	Address Space Layout Randomization
CDR	Client Drive Redirection
DRBG	Deterministic Random Bit Generator
ECC	Elliptic Curve Cryptography
MMR	Multimedia Redirection
PGP	Pretty Good Privacy
PII	Personally Identifiable Information
PP	Protection Profile

RDP	Remote Desktop Protocol
SAML	Security Assertion Markup Language
SAN	Subject Alternative Name
SoC	System-on-Chip
ST	Security Target
SWID	Software Identification (standard)
TOE	Target of Evaluation
TSF	TOE Security Functionality
UAG	Unified Access Gateway
VPN	Virtual Private Network
WAN	Wide-Area Network

2 Product and TOE Description

2.1 Introduction

VMware Horizon is a collection of applications that work together to deliver centralized enterprise resources to end users. This is done by providing users with a "virtual desktop" that consolidates their authorized enterprise computing environments and applications into a single view that is presented to them through a client application.

For this Security Target, the Target of Evaluation (TOE) is the VMware Horizon Client 8 application, specifically version 2209 or 8.7. This is the application that resides on the end user device that is used to access these enterprise resources.

The TOE conforms to the App PP and TLS Package. As such, the security-relevant functionality of the product is limited to the claimed requirements in those standards. The security-relevant functionality is described in sections 2.3 and 2.4. The product overview in section 2.2 below is intended to provide the reader with an overall summary of the entire product so that its intended usage is clear. The subset of the product functionality that is within the evaluation scope is subsequently described in the sections that follow it.

2.2 Product Overview

VMware Horizon is a suite of applications that establish a virtualization environment within an organization. The Horizon applications collectively allow users to access virtualized desktops or enterprise resources from their end user device. These resources are made available with granular security controls that allow users to access only the capabilities for which they are authorized.

VMware Horizon as a suite consists of several components:

- Horizon Clients are applications that are installed on end user devices. A user accesses their virtual desktop through the Horizon Client.
- Horizon Agents are applications that run on virtual servers in the enterprise environment. These agents facilitate remote access to the desktop of a virtual server or to specific applications running on that server that may be served directly to the virtual desktop.
- The Horizon Connection Server is responsible for brokering connections between Horizon Clients and Horizon Agents to authenticate users and serve appropriate resources to a particular user based on enterprise permissions.

A VMware Horizon deployment typically includes one or more instances of the VMware Unified Access Gateway (UAG) as well. The purpose of the UAG is to enforce separation of internal and external networks. This allows the Horizon Client to act as a TLS VPN to access services within the protected network when the end user device is in an external setting such as an untrusted mobile Wi-Fi network.

In cases where a Horizon deployment needs to give users access to resources that span multiple physical data centers or are maintained by multiple branches within an organization, Horizon also supports a cloud pod architecture. This allows for multiple Connection Servers to be federated so that access to Horizon Agent resources on disparate WANs can be served through a single Horizon Client session. Within a single pod, a single Connection Server can also be replicated to ensure availability.

In the evaluated configuration, users interact with resources on their virtual desktop using the VMware Blast Extreme protocol ("Blast protocol"), which is a VMware proprietary encoding protocol for real-time streaming of video data from a remote device. In the evaluated configuration, the Blast protocol is configured to be transmitted over TCP.

2.3 TOE Overview

The Target of Evaluation (TOE) is VMware Horizon Client 8 application. The specific evaluated version of the application is version 2209 or 8.7; these are synonymous. All references to "Horizon Client" throughout the ST refer to this specific version. The TOE includes the Windows and Android platform versions of this application. The user-facing functionality for both platform versions is fundamentally the same.

With respect to the security functionality of the TOE, the TSF is limited to the relevant functionality that is defined in the claimed PP and package. The logical boundary of the TOE is summarized in section 2.4.2. However, the following general capabilities are considered to be within the scope of the TOE:

- **Protection of sensitive data at rest:** the TOE leverages secure platform storage mechanisms to protect sensitive credential data at rest.
- **Protection of data in transit:** the TOE secures data in transit between itself and its operational environment using TLS/HTTPS.
- **Trusted updates:** the TOE provides visibility into its current running version and the vendor distributes updates to it that are digitally signed so that administrators can securely maintain up-to-date software.
- **Cryptographic services:** the TOE includes an implementation of OpenSSL with CAVP validated algorithm services that it uses to secure data in transit.
- **Secure interaction with operating system:** the TOE is designed to interact with its underlying host operating system platform in such a way that the TOE cannot be used as an attack vector to compromise an operating system. The specific mechanisms to achieve this differ between platform versions but the same underlying security threats are mitigated in both cases.

Notably, there are no standardized security requirements in the claimed PP (or any other published PP at the time of this ST's publication) for application layer authentication and authorization. Therefore, the security of these interfaces is only assessed with respect to the ability of the TOE to protect these communications from unauthorized modification or disclosure.

2.4 TOE Architecture

The Horizon Client TOE consists of the Horizon Client application. The TOE has both Windows and Android platform versions. Both platform versions consist of C and C++ code; the Android platform version also has Java components and the Windows platform version also has C++/CLI and C# components. All third-party components used by the TOE (see Appendix A.2) are linked into the TOE binaries; the sole executable process is the TOE itself.

2.4.1 Physical Boundary

The TOE consists of the following component, as shown in Figure 1 below:

- VMware Horizon Client 8 2209 (Horizon 8.7)

Figure 1 shows the TOE in a sample deployment with other VMware Horizon applications in its operational environment. Note the following:

- This figure also includes a UAG and therefore assumes that the TOE platform is located outside of the protected network in which the other Horizon components reside. The UAG acts as a reverse proxy to handle all inbound TLS/HTTPS connections from the Horizon Client.
- Firewalls are not shown between internal and external networks but it is assumed that the UAG is deployed in a DMZ between them.
- Multiple UAGs may be deployed in a load balancer configuration to ensure resource availability. As the claimed PP and package do not have availability requirements, only one UAG is deployed in the tested configuration.
- The second Horizon Connection Server that is depicted on the diagram has its own associated Horizon Agents and other external interfaces. These are omitted for simplicity.
- Horizon Agents are deployed on multiple virtual systems. Horizon Agents support multiple hypervisors but the TOE's evaluated configuration assumes the use of VMware ESXi for this. For simplicity's sake, the Horizon Agent is represented by a single logical component on the figure. The associated physical device and hypervisor are not shown on the figure as the TSF only interfaces logically with the specific application component of those systems. The UAG and Connection Server are similarly deployed in virtualized environments and the figure does not show their dependent components.
- The Connection Server network may include multiple virtual systems on the same physical host that are networked virtually. Specifically, the same physical host may include separate VMs running Horizon Connection Server, Horizon Agent, and vCenter components all as part of the same managed VM infrastructure.
- The environment assumes that all components have access to the organization's Certification Authority for issuance and validation of X.509 certificates.
- The 'Database' component refers to the optional Event Database (SQL Server or Postgres) that is used by the Connection Server if configured.
- The operational environment includes two CRL distribution points: one for certificate revocation checking for components on the internal protected network, and one for the Horizon Client and UAG on the external network. The internal network's CRL distribution point is non-interfering with respect to the security of the TSF.

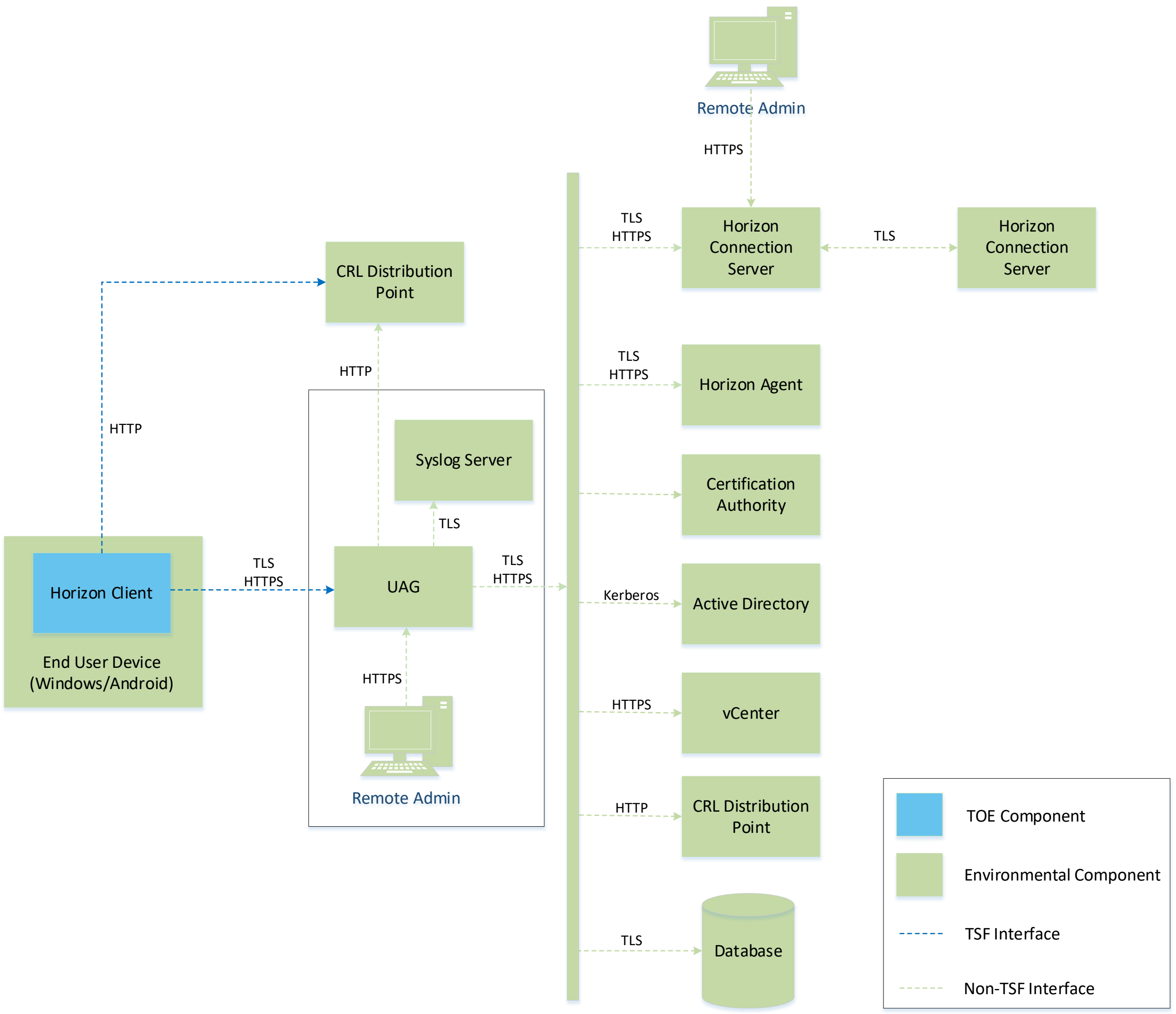

Figure 1 - TOE Boundary

The TOE interfaces directly with the UAG in its operational environment as a reverse proxy to interact with the Connection Server and Horizon Agent(s). Following authentication by the UAG, the initial connection is made to the Connection Server to identify the resources that the user has access to, based on their organizational role or other permissions. The Connection Server then brokers connections to the relevant Horizon Agents.

Connectivity to the UAG uses mutually-authenticated TLS, where the user's smartcard certificate is used to authenticate the TLS client.

The TOE has the following system requirements for its host platform:

- x86-based processor with 800MHz or higher clock speed (Windows), x86-based or ARM-based processor (Android) – note that the tested Windows platform used 64-bit x86 and the tested Android platform used arm64
- 1 GB RAM
- Windows 10 or Android 11, depending on platform version

- Platform must be configured into FIPS-compliant mode of operation (Windows)
- .NET Framework version 4.5 or later (Windows)
- H.264 codec support

The following network ports must be open for the TOE to function:

- TCP/443 (for initial connection to Connection Server via UAG and Blast protocol connectivity to Horizon Agent via UAG)

The TOE's operational environment includes the following:

- Other VMware Horizon components (at least one each of Horizon Connection Server and Horizon Agent).
- Network access to other VMware Horizon components mediated through at least one VMware UAG.
- Platform (hardware and software) on which the TOE is hosted.
    - The TOE is capable of running on a general-purpose Windows or Android operating system on standard consumer-grade hardware. For the evaluated configuration, the TOE was tested on the following environments:
        - Windows 10: Intel Core i7-10850H (10th Gen, Comet Lake) processor on Dell Precision 5550
        - Android 11: Samsung Exynos 9820 (M4, Cortex-A75, Cortex-A55) processor on Samsung Galaxy S10 5G Module/SKU SM-G977N
- Access to a Certification Authority and corresponding revocation checking mechanism is needed to validate presented X.509 certificates.

The TOE has the following logical exclusions, in addition to any functionality that is not directly related to any of the SFR claims made in this ST:

- Tunnel Channel – The Horizon Client has a separate tunnel channel that allows for communications of Microsoft RDP and Windows Media MMR through HTTPS. This channel also allows a USB device connected to the end user workstation to be accessible on the virtual desktop as if it was plugged in to the remote device (USB redirection), and it allows for the Horizon Client's local file system to be similarly accessible on the virtual desktop (Client Drive Redirection, or CDR). In the evaluated configuration, the communications that use the tunnel channel are configured to use Blast instead.
- PCoIP – The Horizon Client supports PC over IP (PCoIP) for remote communications with Horizon Agents. In the evaluated configuration, this is disabled on the server side of the connection and Blast is used instead.
- Direct Connection Server Interface – Depending on network architecture, a Horizon Client may be configured to communicate directly with a Connection Server as part of establishing a connection to a Horizon Agent. In the evaluated configuration, this particular deployment is not used and all communications from the Horizon Client are routed through a UAG.

2.4.2 Logical Boundary

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

2.4.2.1 Cryptographic Support

The TOE implements cryptography to protect data in transit. For data in transit, the TOE implements TLS/HTTPS as a client. The TOE supports mutual authentication for its TLS connections.

The TOE implements all cryptography used for these functions using its own implementations of OpenSSL with NIST-approved algorithms. The TOE's DRBG is seeded using entropy from the underlying OS platform.

For data at rest, the TOE relies on its operational environment to control access to stored credential data.

2.4.2.2 User Data Protection

The TOE relies on credential storage mechanisms to protect sensitive data at rest.

The TOE relies on the network connectivity of its host OS platform. The TOE can also access the system clipboard (depending on administrative configuration), audio/video capture devices, and attached USB storage devices and file system resources.

2.4.2.3 Identification and Authentication

The TOE supports X.509 certificate validation as part of establishing TLS/HTTPS connections. The TOE relies on platform-provided functionality to support various certificate validity checking methods, including the checking of certificate revocation status using CRL. If the validity status of a certificate cannot be determined, the certificate will be accepted or rejected based on administrative configuration. All other cases where a certificate is found to be invalid will result in rejection without an administrative override.

2.4.2.4 Security Management

The TOE itself and the configuration settings it uses are stored in locations recommended by the platform vendor. The TOE is launched by an authenticated OS user and runs in the session context of that user; there is no interface for a non-administrator to act as an administrator through separate authentication. When in its evaluated configuration, the TOE does not have any security-relevant management functions as all security-relevant configuration is done as part of the initial setup of the TOE.

2.4.2.5 Privacy

The TOE does not have an interface to request or transmit requested PII from a user; PII is only transmitted over the network if initiated by the user.

2.4.2.6 Protection of the TSF

The TOE enforces various mechanisms to prevent itself from being used as an attack vector to its host OS platform. The TOE implements address space layout randomization (ASLR), does not allocate any memory with both write and execute permissions, does not write user-modifiable files to directories that contain executable files, is compiled using stack overflow protection, and is compatible with the security features of its host OS platform.

The TOE contains libraries and invokes system APIs that are well-known and explicitly identified.

The TOE has a mechanism to determine its current software version. Software updates to the TOE can be acquired through the application itself or by leveraging its OS platform, depending on the platform version of the TOE. All updates are digitally signed to guarantee their authenticity and integrity.

2.4.2.7 Trusted Path/Channels

The TOE encrypts sensitive data in transit between itself and its operational environment using TLS/HTTPS. These interfaces are used to secure all data in transit between the TOE and its operational environment.

2.5 TOE Documentation

VMware provides the following product documentation in support of the installation and secure use of the TOE:

- VMware Horizon 2209 Horizon Administration (https://docs.vmware.com/en/VMware-Horizon/2209/horizon-console-administration.pdf), 2022
- VMware Horizon 2209 Horizon Installation and Upgrade (https://docs.vmware.com/en/VMware-Horizon/2209/horizon-installation.pdf), 2022
- VMware Horizon 2209 Horizon Security (https://docs.vmware.com/en/VMware-Horizon/2209/horizon-security.pdf), 2022
- VMware Horizon 2209 Horizon Overview and Deployment Planning (https://docs.vmware.com/en/VMware-Horizon/2209/horizon-architecture-planning.pdf), 2022
- VMware Horizon Client for Windows 2209 User Guide (https://docs.vmware.com/en/VMware-Horizon-Client-for-Windows/2209/horizon-client-windows-installation.pdf), 2022
- VMware Horizon Client for Android 2209 User Guide (https://docs.vmware.com/en/VMware-Horizon-Client-for-Android/2209/horizon-client-android-installation.pdf), 2022
- VMware Horizon Client 8 2209 (Horizon 8.7) Common Criteria (CC) Evaluated Configuration Guidance, Version 1.0, April 4, 2023

3 Security Problem Definition

This ST includes by reference the Security Problem Definition, composed of threats and assumptions, from the App PP. The Common Criteria also provides for organizational security policies to be part of a security problem definition, but no such policies are defined in the App PP.

As a functional package, the TLS Package does not contain a Security Problem Definition. The TOE's use of TLS is intended to mitigate the T.NETWORK_ATTACK and T.NETWORK_EAVESDROP threats defined by the App PP.

In general, the threat model of the App PP is designed to protect against the following:

- Disclosure of sensitive data at rest or in transit that the user has a reasonable expectation of security for.
- Excessive or poorly-implemented interfaces with the underlying platform that allow an application to be used as an intrusion point to a system.

This threat model is applicable to the TOE because aggregated and analyzed vulnerability scan results could show an attacker what system weaknesses are present in the environment if they were able to obtain this data. It is also applicable because the TOE is a collection of executable binaries that an attacker could attempt to use to compromise the underlying OS platform if it was designed in such a manner that this exploitation was possible.

4 Security Objectives

Like the Security Problem Definition, this ST includes by reference the security objectives defined in the App PP. This includes security objectives for the TOE (used to mitigate threats) and for its operational environment (used to satisfy assumptions).

As a functional package, the TLS Package does not contain a Security Problem Definition. The TOE's use of TLS is intended to satisfy the O.PROTECTED_COMMS objective of the App PP by implementing a specific method by which network communications are protected.

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 SFRs have all been drawn from the following Protection Profiles (PP) and Functional Packages:

- *Protection Profile for Application Software*, Version 1.4, October 7, 2021
- *Functional Packages for Transport Layer Security (TLS),* Version 1.1, February 12, 2019

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.

5.1 Extended Requirements

All of the extended requirements in this ST have been drawn from the App PP and TLS Package. These documents define the following extended SAR and extended SFRs; since they have not been redefined in this ST, the App PP and TLS Package should be consulted for more information regarding these extensions to CC Parts 2 and 3.

Defined in App PP:

- ALC_TSU_EXT.1 Timely Security Updates
- FCS_CKM_EXT.1 Cryptographic Key Generation Services
- FCS_HTTPS_EXT.1/Client HTTPS Protocol
- FCS_RBG_EXT.1 Random Bit Generation Services
- FCS_RBG_EXT.2 Random Bit Generation from Application
- FCS_STO_EXT.1 Storage of Credentials
- FDP_DAR_EXT.1 Encryption of Sensitive Application Data
- FDP_DEC_EXT.1 Access to Platform Resources
- FDP_NET_EXT.1 Network Communications
- FIA_X509_EXT.1 X.509 Certificate Validation
- FIA_X509_EXT.2 X.509 Certificate Authentication
- FMT_CFG_EXT.1 Secure by Default Configuration
- FMT_MEC_EXT.1 Supported Configuration Mechanism
- 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

Defined in TLS Package:

- FCS_TLS_EXT.1 TLS Protocol

- FCS_TLSC_EXT.1 TLS Client Protocol
- FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication
- FCS_TLSC_EXT.3 TLS Client Support for Signature Algorithms Extension
- FCS_TLSC_EXT.5 TLS Client Support for Supported Groups Extension

5.2 TOE Security Functional Requirements

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

Table 3: TOE Security Functional Components

Requirement Class	Requirement Component
FCS: Cryptographic Support	FCS_CKM_EXT.1 Cryptographic Key Generation Services
	FCS_CKM.1/AK Cryptographic Asymmetric Key Generation
	FCS_CKM.1/SK Cryptographic Symmetric Key Generation
	FCS_CKM.2 Cryptographic Key Establishment
	FCS_COP.1/Hash Cryptographic Operation – Hashing
	FCS_COP.1/KeyedHash Cryptographic Operation – Keyed-Hash Message Authentication
	FCS_COP.1/Sig Cryptographic Operation – Signing
	FCS_COP.1/SKC Cryptographic Operation – Encryption/Decryption
	FCS_HTTPS_EXT.1/Client HTTPS Protocol
	FCS_RBG_EXT.1 Random Bit Generation Services
	FCS_RBG_EXT.2 Random Bit Generation from Application
	FCS_STO_EXT.1 Storage of Credentials
	FCS_TLS_EXT.1 TLS Protocol (TLS Package)
	FCS_TLSC_EXT.1 TLS Client Protocol (TLS Package)
	FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication (TLS Package)
	FCS_TLSC_EXT.3 TLS Client Support for Signature Algorithms Extension (TLS Package)
	FCS_TLSC_EXT.5 TLS Client Support for Supported Groups Extension (TLS Package)
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

Requirement Class	Requirement Component
	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 FCS_CKM_EXT.1 Cryptographic Key Generation Services[1]

FCS_CKM_EXT.1.1 The application shall [

- Implement asymmetric key generation

].

5.2.1.2 FCS_CKM.1/AK Cryptographic Asymmetric Key Generation[2]

FCS_CKM.1.1/AK The application shall [

- implement functionality

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

- [ECC schemes] using [“NIST curves” P-384 and [no other curves]] that meet the following: [FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4]

].

5.2.1.3 FCS_CKM.1/SK Cryptographic Symmetric Key Generation

FCS_CKM.1.1/SK The application shall generate symmetric cryptographic keys using a Random Bit Generator as specified in FCS_RBG_EXT.1 and specified cryptographic key sizes [

- 256 bit

].

[1] Modified by TD0717

[2] Modified by TD0717

5.2.1.4 FCS_CKM.2 Cryptographic Key Establishment

FCS_CKM.2.1 The application shall [implement functionality] to perform cryptographic key establishment in accordance with a specified cryptographic key establishment method: [

- [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"]

].

5.2.1.5 FCS_COP.1/Hash Cryptographic Operation – Hashing[3]

FCS_COP.1.1/Hash The application shall perform [cryptographic hashing services] in accordance with a specified cryptographic algorithm [

- SHA-384

] and message digest sizes [

- 384

] bits that meet the following: [FIPS Pub 180-4].

5.2.1.6 FCS_COP.1/KeyedHash Cryptographic Operation – Keyed-Hash Message Authentication[4]

FCS_COP.1.1/KeyedHash The application shall perform [keyed-hash message authentication] in accordance with a specified cryptographic algorithm [

- HMAC-SHA-384

] and [

- no other algorithms

] with key sizes [*384 bits*] and message digest sizes [384] and [no other size] bits that meet the following: [FIPS Pub 198-1, 'The Keyed-Hash Message Authentication Code' and FIPS Pub 180-4 'Secure Hash Standard'].

5.2.1.7 FCS_COP.1/Sig Cryptographic Operation – Signing[5]

FCS_COP.1.1/Sig The application shall perform [cryptographic signature services (generation and verification)] in accordance with a specified cryptographic algorithm [

[3] Modified by TD0717

[4] Modified by TD0717

[5] Modified by TD0717

- RSA schemes using cryptographic key sizes of [*2048-bit or greater*] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5

].

5.2.1.8 FCS_COP.1/SKC Cryptographic Operation – Encryption/Decryption[6]

FCS_COP.1.1/SKC The application shall perform [encryption/decryption] in accordance with a specified cryptographic algorithm [

- AES-GCM (as defined in NIST SP 800-38D) mode

] and cryptographic key sizes [256-bit].

5.2.1.9 FCS_HTTPS_EXT.1/Client HTTPS Protocol

FCS_HTTPS_EXT.1.1/Client The application shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2/Client The application shall implement HTTPS using TLS as defined in the Functional Package for TLS.

FCS_HTTPS_EXT.1.3/Client The application shall [notify the user and not establish the user-initiated connection] if the peer certificate is deemed invalid.

5.2.1.10 FCS_RBG_EXT.1 Random Bit Generation Services

FCS_RBG_EXT.1.1 The application shall [

- implement DRBG functionality

] for its cryptographic operations.

5.2.1.11 FCS_RBG_EXT.2 Random Bit Generation from Application

FCS_RBG_EXT.2.1 The application shall perform all deterministic random bit generation (DRBG) services in accordance with NIST Special Publication 800-90A using [CTR_DRBG (AES)].

FCS_RBG_EXT.2.2 The deterministic RBG shall be seeded by an entropy source that accumulates entropy from a platform-based DRBG and [

- no other noise source

] with a minimum of [

- 256 bits

[6] Modified by TD0717

] of entropy at least equal to the greatest security strength (according to NIST SP 800-57) of the keys and hashes that it will generate.

5.2.1.12 FCS_STO_EXT.1 Storage of Credentials

FCS_STO_EXT.1.1 The application shall [

- invoke the functionality provided by the platform to securely store [*TLS client certificate private key*]

] to non-volatile memory.

5.2.1.13 FCS_TLS_EXT.1 TLS Protocol (TLS Package)

FCS_TLS_EXT.1.1 The product shall implement [

- TLS as a client

].

5.2.1.14 FCS_TLSC_EXT.1 TLS Client Protocol (TLS Package)[7]

FCS_TLSC_EXT.1.1 The product shall implement TLS 1.2 (RFC 5246) and [no earlier TLS versions] as a client that supports the cipher suites [

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289

] and also supports functionality for [

- mutual authentication

].

FCS_TLSC_EXT.1.2 The product shall verify that the presented identifier matches the reference identifier according to RFC 6125.

FCS_TLSC_EXT.1.3 The product shall not establish a trusted channel if the server certificate is invalid [

- with no exceptions

].

5.2.1.15 FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication (TLS Package)

FCS_TLSC_EXT.2.1 The product shall support mutual authentication using X.509v3 certificates.

[7] Modified by TD0442

5.2.1.16 FCS_TLSC_EXT.3 TLS Client Support for Signature Algorithms Extension (TLS Package)

FCS_TLSC_EXT.3.1 The product shall present the signature_algorithms extension in the Client Hello with the supported_signature_algorithms value containing the following hash algorithms: [SHA384] and no other hash algorithms.

5.2.1.17 FCS_TLSC_EXT.5 TLS Client Support for Supported Groups Extension (TLS Package)

FCS_TLSC_EXT.5.1 The product shall present the Supported Groups Extension in the Client Hello with the supported groups [

- secp384r1

].

5.2.2 User Data Protection (FDP)

5.2.2.1 FDP_DAR_EXT.1 Encryption of Sensitive Application Data

FDP_DAR_EXT.1.1 The application shall [

- protect sensitive data in accordance with FCS_STO_EXT.1

] in non-volatile memory.

5.2.2.2 FDP_DEC_EXT.1 Access to Platform Resources

FDP_DEC_EXT.1.1 The application shall restrict its access to [

- network connectivity,
- camera,
- microphone,
- location services,
- USB,
- [*smartcard,*
- *scanner,*
- *serial port devices,*
- *printer,*
- *speaker,*
- *input devices (keyboard, mouse, etc.),*
- *monitor*]

].

FDP_DEC_EXT.1.2 The application shall restrict its access to [

- [*file system,*
- *clipboard,*
- *system information*]

].

5.2.2.3 FDP_NET_EXT.1 Network Communications

FDP_NET_EXT.1.1 The application shall restrict network communication to [

- user-initiated communication for [*initial connectivity to UAG*],
- [*application-initiated communication Blast data path*]

].

5.2.3 Identification and Authentication (FIA)

5.2.3.1 FIA_X509_EXT.1 X.509 Certificate Validation

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 [CRL as specified in RFC 5280 Section 6.3, CRL as specified in RFC 8603].
- 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.
  - 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.

Application Note: *There are no cases where the TOE is presented with a code signing, TLS client, S/MIME, OCSP, or EST certificate in its evaluated configuration so certificates designated for these purposes will never be accepted by the TSF.*

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 FIA_X509_EXT.2 X.509 Certificate Authentication

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

FIA_X509_EXT.2.2 When the application cannot establish a connection to determine the validity of a certificate, the application shall [allow the administrator to choose whether to accept the certificate in these cases].

5.2.4 Security Management (FMT)

5.2.4.1 FMT_CFG_EXT.1 Secure by Default Configuration

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.

5.2.4.2 FMT_MEC_EXT.1 Supported Configuration Mechanism

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 FMT_SMF.1 Specification of Management Functions

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

- no management functions

].

5.2.5 Privacy (FPR)

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

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 FPT_AEX_EXT.1 Anti-Exploitation Capabilities

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 FPT_API_EXT.1 Use of Supported Services and APIs

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

5.2.6.3 FPT_IDV_EXT.1 Software Identification and Versions

FPT_IDV_EXT.1.1 The application shall be versioned with [[*date-based versioning, major/minor release versioning*]].

5.2.6.4 FPT_LIB_EXT.1 Use of Third Party Libraries

FPT_LIB_EXT.1.1 The application shall be packaged with only [*third-party libraries listed in Appendix A.2*].

Application Note: *The TOE uses a substantial number of third-party libraries so this information has been provided in an Appendix for readability purposes.*

5.2.6.5 FPT_TUD_EXT.1 Integrity for Installation and Update

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

Application Note: *The Windows platform version of the TOE checks for software updates through the TOE itself. The Android platform version of the TOE relies on the Google Play store.*

FPT_TUD_EXT.1.2 The application shall [provide the ability, leverage the platform] 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 FPT_TUD_EXT.2 Integrity for Installation and Update

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 FTP_DIT_EXT.1 Protection of Data in Transit[8]

FTP_DIT_EXT.1.1 The application shall [

- encrypt all transmitted [data] with [HTTPS as a client in accordance with FCS_HTTPS_EXT.1/Client, TLS as a client as defined in the Functional Package for TLS]

] between itself and another trusted IT product.

5.3 TOE Security Assurance Requirements

The security assurance requirements for the TOE are included by reference to the App PP.

Table 4: Assurance Components

Requirement Class	Requirement Component
ADV: Development	ADV_FSP.1 Basic Functional Specification
AGD: Guidance Documentation	AGD_OPE.1 Operational User Guidance
	AGD_PRE.1 Preparative Procedures
ALC: Life-cycle Support	ALC_CMC.1 Labeling 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

[8] Modified by TD0655

As a functional package, the TLS Package does not define its own SARs. The expectation is that all SARs required by the App PP will apply to the entire TOE, including the portions addressed by the TLS Package. Consequently, the evaluation activities specified in the App PP apply to the entire TOE evaluation, including any changes made to them by subsequent NIAP Technical Decisions as summarized in section 1.2 above.

The TLS Package does contain evaluation activities for how to evaluate its SFR claims as part of the evaluation of ASE_TSS.1, AGD_OPE.1, AGD_PRE.1, and ATE_IND.1. All Security Functional Requirements specified by the TLS Package will be evaluated in the manner specified in that package.

6 TOE Summary Specification

This chapter describes the security functions of the TOE:

- Timely Security Updates
- Cryptographic Support
- User Data Protection
- Identification and Authentication
- Security Management
- Privacy
- Protection of the TSF
- Trusted Path/Channels

6.1 Timely Security Updates

VMware uses an internal classification system to categorize product security flaws by severity level. The classifications and their respective service-level agreements for mitigation are as follows:

- Critical:
  - Vulnerabilities that can be exploited by an unauthenticated attacker from the Internet or those that break the guest/host Operating System isolation. The exploitation results in the complete compromise of confidentiality, integrity, and availability of user data and/or processing resources without user interaction. Exploitation could be leveraged to propagate an Internet worm or execute arbitrary code between Virtual Machines and/or the Host Operating System.
  - A fix or corrective action is begun immediately and will be made available in the shortest commercially reasonable time.
- Important:
  - Vulnerabilities that are not rated critical but whose exploitation results in the complete compromise of confidentiality and/or integrity of user data and/or processing resources through user assistance or by authenticated attackers. This rating also applies to those vulnerabilities which could lead to the complete compromise of availability when exploitation is by a remote unauthenticated attacker from the Internet or through a breach of virtual machine isolation.
  - A fix will be delivered as part of the next planned maintenance release of the product and will be released as a patch if appropriate to do so.
- Moderate:
  - Vulnerabilities where the ability to exploit is mitigated to a significant degree by configuration or difficulty of exploitation, but in certain deployment scenarios could still lead to the compromise of confidentiality, integrity, or availability of user data and/or processing resources.
  - A fix will be delivered with the next planned major or minor release of the product.
- Low:
  - All other issues that have a security impact. Vulnerabilities where exploitation is believed to be extremely difficult, or where successful exploitation would have minimal impact.
  - A fix will be delivered with the next planned major or minor release of the product.

The standard release cycle for VMware products is quarterly, so all Moderate and Low findings are typically resolved within a maximum of 90 days, while more significant findings are generally resolved in less time. Both quarterly releases and mid-cycle patches can be obtained for the Windows Client from https://customerconnect.vmware.com, while the Android Client uses the Google Play store.

VMware provides an email address (security@vmware.com) that is used for the reporting of potential security findings. VMware encourages the use of Pretty Good Privacy (PGP) to encrypt any communications sent to this email address and provides a copy of their PGP public key at https://kb.vmware.com/s/article/1055.

VMware staff identifies potential vulnerabilities through third-party researchers reporting potential flaws via email, reports from field personnel, reports from customers, and monitoring of public vulnerability sites. When a report is received, VMware attempts to reproduce the finding and determine its severity. If a finding is discovered for which there is no current fix, VMware will publish a Knowledge Base article about the finding as well as any potential workarounds that may be used until an updated version of the product can be delivered.

6.2 Cryptographic Support

The TOE uses cryptography to secure data in transit between itself and its operational environment.

TSF cryptographic services are implemented by the OpenSSL cryptographic library included within the TOE boundary. Both platform versions of the TOE use VMware's OpenSSL FIPS Object Module 2.0.20-vmw. The cryptographic algorithms supplied by the TOE are CAVP validated. The following table identifies the cryptographic algorithms used by the TSF, the associated standards to which they conform, and the NIST certificates that demonstrate that the claimed conformance has been met.

Table 5: Cryptographic Algorithm Claims

Functions	Libraries	Standards	Certificates
FCS_CKM.1/AK Cryptographic Asymmetric Key Generation			
ECC key pair generation (NIST curve P-384)	OpenSSL	FIPS PUB 186-4	A1292
FCS_CKM.2 Cryptographic Key Establishment			
Elliptic curve-based key establishment	OpenSSL	NIST SP 800-56A	A1292
FCS_COP.1/Hash Cryptographic Operation – Hashing			
SHA-384 (digest size 384 bits)	OpenSSL	FIPS PUB 180-4	A1292
FCS_COP.1/KeyedHash Cryptographic Operation – Keyed Hash Message Authentication			
HMAC-SHA-384	OpenSSL	FIPS PUB 198-1 FIPS PUB 180-4	A1292
FCS_COP.1/Sig Cryptographic Operation – Signing			
RSA (2048, 3072-bit)	OpenSSL	FIPS PUB 186-4, Section 5	A1292
FCS_COP.1/SKC Cryptographic Operation – Encryption/Decryption			
AES-GCM (256 bits)	OpenSSL	GCM as defined in NIST SP 800-38D	A1292
FCS_RBG_EXT.2 Random Bit Generation from Application			

Functions	Libraries	Standards	Certificates
AES-CTR_DRBG (256 bits)	OpenSSL	NIST SP 800-90A NIST SP 800-57	A1292

The TOE generates asymmetric keys in support of trusted communications. The TSF generates ECC keys using P-384. The TOE generates these keys in support of the ECDHE key establishment schemes used for TLS/HTTPS communications. To ensure sufficient key strength, the TOE also implements DRBG functionality for key generation, using the AES-CTR_DRBG. The proprietary Entropy Analysis Report (EAR) describes how the TSF extracts random data from software-based sources to ensure that an amount of entropy that is at least equal to the strength of the generated keys is present (i.e., at least 256 bits when the largest supported keys are generated) when seeding the DRBG for key generation purposes. The Windows platform version of the TOE relies on a third-party entropy source provided by the platform vendor. The Android platform version of the TOE relies on the OS platform entropy source. Specifically, random numbers are obtained from the following platform APIs, depending on the platform used:

- Windows: BCryptGenRandom
- Android: invocation of /dev/random pseudo-device

In both cases, it is assumed that these platforms provide at least 256 bits of entropy.

The TOE uses TLS 1.2 as part of HTTPS for client communications. The TOE's implementation of TLS conforms to RFC 5246 and its implementation of HTTPS conforms to RFC 2818. The specific TOE network interfaces are documented below in section 6.3. The TLS client offers the following cipher suite in its evaluated configuration:

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289

The supported ciphersuite uses elliptic curve ephemeral Diffie-Hellman as the method of key establishment. The TSF presents secp384r1 in the Supported Groups extension as the parameter used for key establishment. The TSF also presents SHA-384 in the signature_algorithms extension as the hash parameter used for digital signatures. If a presented server certificate is invalid, the TSF will automatically reject it and notify the user when in its evaluated configuration. The evaluated configuration of the TOE is to support appropriate certificates using the RSA+SHA384 configuration string that is applied during initial configuration. Digital signature generation and verification is supported for RSA 2048-bit and 3072-bit certificates.

As part of certificate validation in the establishment of TLS connectivity, the TOE will validate the reference identifier of a presented server certificate. This is done through comparison of the DNS name presented in the Subject Alternative Name (SAN) certificate field to the hostname of the server. IP addresses are not supported. Wildcards are only supported for the left-most label immediately preceding the public suffix. Certificate pinning is not supported.

The TOE supports TLS client functionality for connectivity to a UAG in its operational environment. Initial connectivity with the UAG is used to obtain an authorization token for virtual desktop access. Subsequent connectivity to access Horizon Agent resources using the Blast protocol is also established through the UAG. These are performed using TLS mutual authentication. For the TLS client certificate used for mutual authentication, the Windows application uses a certificate linked to a physical smartcard connected to the host OS platform. The Android application uses a certificate linked to a virtual smartcard that is derived from the user credentials. The certificate resides in Android KeyStore.

The TOE relies on platform-provided storage mechanisms for credential data. Both the Windows and Android platform versions store the private key for the TLS client certificate. The Windows platform version uses the Windows Certificate Store. The Android platform version uses the OS keychain.

The Cryptographic Support security function is designed to satisfy the following security functional requirements:

- FCS_CKM_EXT.1 – The TOE implements its own cryptographic functionality.
- FCS_CKM.1/AK – The TOE uses a CAVP validated implementation to generate asymmetric keys in support of TLS/HTTPS communications.
- FCS_CKM.1/SK – The TOE uses its DRBG to generate symmetric keys used for AES.
- FCS_CKM.2 – The TOE performs CAVP validated key establishment in support of TLS/HTTPS communications.
- FCS_COP.1/Hash – The TOE uses a CAVP validated implementation to perform cryptographic hashing in support of TLS/HTTPS communications.
- FCS_COP.1/KeyedHash – The TOE uses a CAVP validated implementation to perform HMAC functions in support of TLS/HTTPS communications.
- FCS_COP.1/Sig – The TOE uses a CAVP validated implementation to generate and verify RSA digital signatures in support of TLS/HTTPS communications.
- FCS_COP.1/SKC – The TOE uses a CAVP validated implementation to perform AES encryption and decryption in support of TLS/HTTPS communications.
- FCS_HTTPS_EXT.1/Client – The TOE implements HTTPS as a client to secure data in transit.
- FCS_RBG_EXT.1 – The TOE implements its own random bit generation services.
- FCS_RBG_EXT.2 – The TOE uses a CAVP validated implementation to generate pseudo-random bits and this implementation is seeded with sufficiently strong entropy collected from the operational environment.
- FCS_STO_EXT.1 – The TOE uses platform-provided mechanisms to secure credential data at rest.
- FCS_TLS_EXT.1 – The TOE implements TLS to secure data in transit.
- FCS_TLSC_EXT.1 – The TOE implements TLS as a client.
- FCS_TLSC_EXT.2 – The TOE’s TLS client implementation supports mutual authentication for some TLS functions.
- FCS_TLSC_EXT.3 – The TOE’s TLS client implementation presents supported hash algorithms to the server in the signature_algorithms extension.
- FCS_TLSC_EXT.5 – The TOE’s TLS client implementation presents supported elliptic curves to the server in the Supported Groups extension.

6.3 User Data Protection

The App PP defines 'sensitive data' as follows: "Sensitive data may include all user or enterprise data or may be specific application data such as emails, messaging, documents, calendar items, and contacts. Sensitive data must minimally include PII, credentials, and keys. Sensitive data shall be identified in the application's TSS by the ST author."

The TSF relies on platform storage mechanisms identified in FCS_STO_EXT.1 to protect credential data at rest in non-volatile storage. No other data is considered to be sensitive.

The TOE has access to a variety of physical and logical system resources from the host platform. For physical resources, the TOE always requires network connectivity to satisfy its intended purpose. Users may also allow the TOE to access their device's camera, microphone, location services, and various external peripherals (including printer/scanner, serial port devices, speakers, input devices, and monitors) so that applications in the virtual desktop have access to these resources if the user authorizes it. For logical resources, the TOE may be administratively configured to allow the TOE to access the clipboard so that users may copy and paste content within the virtual desktop and between the virtual desktop and applications running natively on the host platform. This is configured on the Horizon Agent so there is no interface on the Horizon Client to control the clipboard behavior. Users may also grant the TOE access to physical and logical storage (i.e. a USB drive or a folder/storage volume on the local hard drive) so that applications on the virtual desktop have access to these repositories. For the Android client, access to storage volumes, camera, microphone, speakers, and location services requires explicit user approval. For the Windows client, explicit user approval is required to access file system, location, and screen capture data. Access to other resources are implicitly granted through the user's informed consent to use the application. Additionally, as part of external connectivity, the TOE collects the following system information that is transmitted to a Horizon Connection Server:

- MAC address
- Device ID
- OS type
- Current user's domain
- OS language
- Time zone
- IP address
- Host domain (Windows client only)
- Device serial number (Windows client only)
- Distinguished Name (Windows client only)
- Number/topology of displays (Windows client only)
- Device model
- Device manufacturer

The TOE interfaces with external components in its operational environment to satisfy its core functionality. The following network interfaces are present in the TSF:

Table 6: TSF Network Usage

Function	Invoked By	Network Port	Secured By

HTTPS connectivity to UAG	User	TCP/443	Mutual TLS/HTTPS (TOE acts as client)
TCP Agent access over Blast	TOE	TCP/443	Mutual TLS/HTTPS (TOE acts as client)

All communications with the virtual desktop (e.g. keyboard/mouse inputs) use Blast, including Windows Media MMR, and USB redirection (i.e. the Horizon Agent's recognition of a physical USB device connected to the TOE system). In the evaluated configuration, the TSF's direct connections are with the UAG; the UAG functions as a reverse proxy and is responsible for handling connections to other Horizon components on the internal network.

The User Data Protection security function is designed to satisfy the following security functional requirements:

- FDP_DAR_EXT.1 – Sensitive data at rest is protected by the TSF's use of platform credential storage repositories.
- FDP_DEC_EXT.1 – The TOE's use of platform services is well understood by users prior to authorizing the TOE activity.
- FDP_NET_EXT.1 – The TOE communicates over the network for well-defined purposes. Depending on the function, the use of network resources is user-initiated directly through the TSF or initiated by the TOE itself.

6.4 Identification and Authentication

The TOE uses X.509 to validate the TLS server certificates of the VMware components that it communicates with (UAG and Horizon Agent).

The TOE invokes platform-provided functionality for the following functional behavior for all uses of X.509 certificates:

- Certificate validation and certificate path validation is performed in accordance with RFC 5280.
- The certificate path is checked to ensure that it terminates with a trusted CA certificate.
- The certificate path is validated 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.
- Any CA certificate is validated by ensuring that the key usage field includes the caSigning purpose.
- Revocation status is checked using CRL. The Windows platform version of the TOE invokes platform services for this in accordance with RFC 8603 while the Android platform version of the TOE invokes services that conform to RFC 5280 section 6.3.
- 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.

In the event that the revocation status of a certificate cannot be verified (i.e. the CRL cannot be reached), administrative configuration determines whether the TOE will accept or reject the certificate. This behavior is configured external to the TSF using a GPO or appconfig setting, depending on which platform version of the client is configured.

Because the TOE's use of the certificate validation function is to validate the authenticity of remote endpoints, the TSF chooses what certificates to use based on what is presented to it as part of establishing the TLS session. The client certificate that the TOE presents to a server when performing TLS mutual authentication is user-configured. For the Windows platform version of the TOE, this is expected to be a certificate stored on a physical smart card and unlocked with a PIN that is associated with the certificate. For the Android platform version of the TOE, the user configures a certificate to be associated with a virtual smart card that is unlocked using a derived credential that is generated from a PIN.

The Identification and Authentication security function is designed to satisfy the following security functional requirements:

- FIA_X509_EXT.1 – X.509 certificates are validated by the TSF when establishing trusted communications.
- FIA_X509_EXT.2 – X.509 certificates are used for TLS. When revocation status of a certificate cannot be determined, the TSF will accept or reject the certificate based on configuration.

6.5 Security Management

The TOE is run locally as an application on the host platform. A user must input a valid credential to gain access to the virtual desktop, but this credential is supplied to the server and is based on an organizational credential (i.e. a user defined in an organization's Active Directory). The TOE itself is launched in the context of the user session on the host OS platform so no separate authentication is required to access the application itself. The only exception to this is when a user configures a virtual smart card on the Android platform version of the TOE. When this is done, the user is prompted to re-authenticate to the device to ensure the user session is valid. This does not apply to the Windows platform version of the TOE because client certificates are configured by a system administrator on the platform.

The Windows platform version of the TOE is installed by default to %ProgramFiles%\VMware\VMware Horizon View Client and %ProgramFiles%\Common Files\VMware. These directories are owned by the Administrator account on the host OS platform, who has write access to them. All other users and groups have read-only access. Security-relevant configuration data is stored in the Windows Registry.

The Android platform version of the TOE similarly does not store its binaries or data files with world-writable access, and its configuration data is stored in SharedPreferences (located at /data/data/packages/shared_prefs).

Security-relevant configuration data is defined during initial setup of the TOE and includes the following:

- TLS version
- TLS cipher suites
- TLS supported signature algorithms
- TLS supported groups
- Certificate revocation checking
- Protocol connection certificate verification mode

The product includes a number of management functions, but these relate primarily to usability and compatibility functions such as setting the desired Blast encoding/decoding method, virtual desktop resolution, or proxy server settings. With respect to the TSF, the product provides the capability to configure the allowed TLS versions and cipher suites as well as its behavior when a server certificate is invalid, but these settings are not modified once the TOE is placed into its evaluated configuration. Therefore, the TSF does not have any security-relevant management functions with respect to its operational use.

The Security Management security function is designed to satisfy the following security functional requirements:

- FMT_CFG_EXT.1 – The TOE is protected from direct modification by untrusted users via its host OS platform. In cases where a user can configure a TLS client certificate to use, the host OS platform forces re-authentication of the user before this operation can be completed.
- FMT_MEC_EXT.1 – Configuration settings for the TOE are stored in an appropriate location in its host OS platform.
- FMT_SMF.1 – Once the TOE is placed into its evaluated configuration, the full extent to which users can manage the product does not relate to any security-relevant behavior with respect to the claimed PP.

6.6 Privacy

The TOE's primary function is to facilitate a user's remote access to enterprise computing resources. As part of doing this, it may access system resources (e.g. camera and microphone) that could allow for inadvertent transmission of user PII. Likewise, the virtual desktop is capable of accepting user keyboard/mouse input and accessing files or folders on the user's device. These interfaces may be used to disclose PII, but any mechanism to do this would be user-initiated; the TSF is not designed or intended for the capture of PII. The risk of inadvertent PII disclosure is assumed by the user through consent to allow the TOE to access computing interfaces over which PII could be transmitted if the user initiated an operation to do so (whether intentionally or not). The TOE accepts credentials from the user that are used by the operational environment to validate the user's identity in order to grant them access to their authorized enterprise computing resources, but user account information is not considered to be PII.

- FPR_ANO_EXT.1 – The TOE does not have an interface to request PII from a user; PII is only transmitted over the network if initiated by the user.

6.7 Protection of the TSF

The TOE implements several mechanisms to protect against exploitation. The TOE implements address space layout randomization (ASLR) through the use of the –fPIC compiler flag and relies fully on its underlying host platforms to perform memory mapping. The TOE also does not use both PROT_WRITE and PROT_EXEC on the same memory regions. There is no situation where the TSF maps memory to an explicit address. The TOE is written in C, C++, C++/CLI, and C# (Windows) and C, C++, and Java (Android). It is compiled with stack overflow protection through the use of the /GS (Windows) and –fstack-protector (Android) compiler flags.

The Windows platform version of the TOE is compatible with the security features of Windows Defender Exploit Guard. Android applications cannot disable platform security features so the Android platform version of the TOE's compatibility with its OS platform security features is assured. The TOE uses only documented platform APIs. Appendix A.1 lists the APIs used by each platform version of the TOE. The TOE also makes use of third-party libraries. Appendix A.2 lists the libraries used by each platform version of the TOE. The TOE is versioned using both YYMM date-based versioning and major/minor to correspond to the approximate release of a particular version and major/minor release versioning, e.g. 2209 refers to the TOE version released on or around September of 2022 and is also synonymous with version 8.7; SWID is not used. The TOE is a standalone application that is not natively bundled as part of a host OS.

The user of the Windows platform version of the TOE can check for software updates manually within the application. If an update is available, the user is prompted to download and install it. The Android platform version of the TOE relies on the OS platform to check for software updates through the Google Play store. Both platform versions of the TOE identify the application version in the application itself as well as through standard OS reporting mechanisms (e.g. the Windows application version can be identified through Add/Remove Programs).

The TOE will not download, modify, replace, or update its own binary code. The Windows platform version of the TOE is packaged as an .exe file and the Android platform version of the TOE is packaged as an .apk file. All installation packages are signed by VMware using 2048-bit RSA. Removing (uninstalling) the product will remove all executable code from the host system.

The Protection of the TSF security function is designed to satisfy the following security functional requirements:

- FPT_AEX_EXT.1 – The TOE interacts with its host OS platform in a manner that does not expose the system to memory-related exploitation.
- FPT_API_EXT.1 – The TOE uses only documented platform APIs.
- FPT_IDV_EXT. 1 – The TOE is versioned using YYMM date-based and major/minor versioning.
- FPT_LIB_EXT.1 – The set of third-party libraries used by the TOE is well-defined.
- FPT_TUD_EXT.1 – There is a well-defined method for checking what version of the TOE is currently installed and whether updates to it are available. Signed updates are validated by the host OS platform prior to installation.
- FPT_TUD_EXT.2 – The TOE can be updated through installation packages.

6.8 Trusted Path/Channels

In the evaluated configuration, the TOE uses its own cryptographic implementation to encrypt sensitive data in transit. Listed below are the various external interfaces to the TOE that rely on trusted communications.

- Between TOE and UAG
  - Communications use TLS/HTTPS (TOE is client)
  - The TOE presents a client certificate for mutual TLS authentication
  - TCP port 443
  - Used to authenticate the user to determine their authorizations to access resources managed by Horizon Agents and to establish initial connectivity to them
  - Client is granted a single-use authorization token upon successful authentication
- Between TOE and UAG (Blast TCP, reverse proxy to Horizon Agent)
  - Communications use TLS/HTTPS (TOE is client)
  - The TOE presents a client certificate for mutual TLS authentication
  - TCP port 443

  - Used as the channel for virtual desktop services that are configured to be transmitted via Blast

The Trusted Path/Channels security function is designed to satisfy the following security functional requirements:

- FTP_DIT_EXT.1 – The TOE relies on its own mechanisms to secure all data in transit between itself and its operational environment.

7 Protection Profile Claims

This ST is conformant to the *Protection Profile for Application Software, Version 1.4, October 7, 2021* (App PP) and *Functional Package for Transport Layer Security (TLS), Version 1.1, February 12,* 2019 (TLS Package) along with all applicable errata and interpretations from the certificate issuing scheme.

The TOE consists of a software application that runs on a Windows or Android operating system as its platform.

As explained in section 3, Security Problem Definition, the Security Problem Definition of the App PP has been included by reference into this ST.

As explained in section 4, Security Objectives, the Security Objectives of the App PP has been included by reference into this ST.

All claimed SFRs are defined in the App PP and TLS Package. All mandatory SFRs are claimed. Some optional or objective SFRs are claimed. Selection-based SFR claims are consistent with the selections made in the mandatory SFRs that prompt their inclusion.

8 Rationale

This Security Target includes by reference the App PP Security Problem Definition, Security Objectives, and Security Assurance Requirements. The Security Target does not add, remove, or modify any of these items. Security Functional Requirements have been reproduced with the Protection Profile operations completed. All selections, assignments, and refinements made on the claimed Security Functional Requirements have been performed in a manner that is consistent with what is permitted by the App PP and TLS Package. The proper set of selection-based requirements have been claimed based on the selections made in the mandatory requirements. Consequently, the claims made by this Security Target are sufficient to address the TOE's security problem. Rationale for the sufficiency of the TOE Summary Specification is provided below.

8.1 TOE Summary Specification Rationale

This section in conjunction with Section 0, the

TOE Summary Specification, provides evidence that the security functions meet the TOE security requirements. Each description includes rationale indicating which requirements the corresponding security functions satisfy. The combined security functions work together to satisfy all of the security requirements. The security functions described in Section 6 are necessary for the TSF to enforce the required security functionality. Table 5 demonstrates the relationship between security requirements and functions.

Table 5: 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.1/SK	X						
FCS_CKM.2	X						
FCS_COP.1/Hash	X						
FCS_COP.1/KeyedHash	X						
FCS_COP.1/Sig	X						
FCS_COP.1/SKC	X						
FCS_HTTPS_EXT.1/Client	X						
FCS_RBG_EXT.1	X						
FCS_RBG_EXT.2	X						
FCS_STO_EXT.1	X						
FCS_TLS_EXT.1	X						
FCS_TLSC_EXT.1	X						
FCS_TLSC_EXT.2	X						
FCS_TLSC_EXT.3	X						
FCS_TLSC_EXT.5	X						
FDP_DAR_EXT.1		X					
FDP_DEC_EXT.1		X					
FDP_NET_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			

	Cryptographic Support	User Data Protection	Identification and Authentication	Security Management	Privacy	Protection of the TSF	Trusted Path/Channels
FPR_ANO_EXT.1					X		
FPT_AEX_EXT.1						X	
FPT_API_EXT.1						X	
FPT_IDV_EXT.1						X	
FPT_LIB_EXT.1						X	
FPT_TUD_EXT.1						X	
FPT_TUD_EXT.2						X	
FTP_DIT_EXT.1							X

A TOE Usage of Third-Party Components

This Appendix lists the platform APIs and third-party libraries that are used by the TOE.

A.1 Platform APIs

Listed below are the platform APIs used by the Horizon Client product.

A.1.1 Windows Platform

ADVAPI32.dll
AVRT.dll
COMCTL32.dll
CRYPT32.dll
CRYPTUI.dll
EVR.dll
FONTSUB.dll
GDI32.dll
IMM32.dll
KERNEL32.dll
LIBEAY32.dll
MF.dll
MPR.dll
MSIMG32.dll
MSVCP140.dll
NETAPI32.dll
Normaliz.dll
OLEAUT32.dll
PROPSYS.dll
RPCRT4.dll
SETUPAPI.dll
SHELL32.dll
SHLWAPI.dll
SSLEAY32.dll
Secur32.dll
SspiCli.dll
USER32.dll
USERENV.dll
VCRUNTIME140.dll
VCRUNTIME140_1.dll
VERSION.dll
WINHTTP.dll
WININET.dll
WINMM.dll
WINTRUST.dll
WLDAP32.dll
WS2_32.dll
WSOCK32.dll
WTSAPI32.dll

WinSDCard.dll
bcrypt.dll
d2d1.dll
d3d11.dll
d3d9.dll
dbghelp.dll
dcomp.dll
dwmapi.dll
dxgi.dll
dxva2.dll
gdiplus.dll
mscoree.dll
msdmo.dll
msi.dll
msvcrt.dll
ncrypt.dll
ntdll.dll
ole32.dll
prntvpt.dll
pthreadVC2.dll

A.1.2 Android Platform

android.Manifest.permission
android.annotation.TargetApi
android.app.ActivityManager.MemoryInfo
android.app.AlarmManager
android.app.DialogFragment
android.app.NotificationChannel
android.app.Presentation
android.app.admin.DeviceAdminReceiver
android.bluetooth.BluetoothAdapter
android.bluetooth.BluetoothDevice
android.content.ClipData
android.content.ClipboardManager.OnPrimaryClipChangedListener
android.content.ClipboardManager
android.content.ComponentName
android.content.ContentProvider
android.content.Context
android.content.DialogInterface.OnDismissListener
android.content.RestrictionsManager
android.content.SharedPreferences
android.content.pm.ApplicationInfo
android.content.pm.FeatureInfo
android.content.pm.ShortcutInfo
android.content.pm.ShortcutManager
android.content.pm.Signature
android.content.res.AssetManager
android.content.res.AssetManager

android.content.res.TypedArray
android.content.res.XmlResourceParser
android.database.DataSetObserver
android.database.SQLException
android.database.sqlite.SQLiteOpenHelper
android.graphics.PointF
android.graphics.PorterDuff
android.graphics.PorterDuffColorFilter
android.graphics.drawable.Icon
android.hardware.Camera
android.hardware.camera2.CameraAccessException
android.hardware.camera2.CameraCaptureSession
android.hardware.camera2.CameraCharacteristics
android.hardware.camera2.CameraDevice
android.hardware.camera2.CameraManager
android.hardware.camera2.CameraMetadata
android.hardware.camera2.CaptureRequest
android.hardware.camera2.params.StreamConfigurationMap
android.location.Location
android.location.LocationListener
android.location.LocationManager
android.media.AudioFormat
android.media.AudioRecord
android.media.Image
android.media.ImageReader
android.media.MediaPlayer
android.media.MediaRecorder.AudioSource
android.media.audiofx.AcousticEchoCanceler
android.media.audiofx.AutomaticGainControl
android.media.audiofx.NoiseSuppressor
android.net.ConnectivityManager
android.net.http.SslCertificate
android.net.wifi.WifiInfo
android.net.wifi.WifiManager
android.opengl.GLES20
android.opengl.GLSurfaceView
android.opengl.Matrix
android.os.AsyncTask
android.os.ConditionVariable
android.os.Looper
android.os.StatFs
android.os.Vibrator
android.preference.CheckBoxPreference
android.preference.ListPreference
android.preference.PreferenceActivity.Header
android.preference.PreferenceActivity
android.preference.PreferenceFragment
android.preference.PreferenceScreen

android.print.PageRange
android.print.PrintAttributes
android.print.PrintDocumentAdapter
android.print.PrintDocumentInfo
android.provider.MediaStore
android.provider.Settings
android.security.keystore.KeyProperties
android.telephony.TelephonyManager
android.test.ActivityInstrumentationTestCase2
android.test.ActivityInstrumentationTestCase
android.text.Editable
android.text.InputFilter
android.text.InputType
android.text.Spannable
android.text.SpannableStringBuilder
android.text.Spanned
android.text.TextWatcher
android.text.TextWatcher
android.text.format.DateFormat
android.text.method.LinkMovementMethod
android.text.method.SingleLineTransformationMethod
android.text.style.ClickableSpan
android.util.Log
android.util.Size
android.util.TypedValue
android.view.ActionMode
android.view.DragAndDropPermissions
android.view.GestureDetector
android.view.InputDevice
android.view.InputEvent
android.view.KeyEvent
android.view.OrientationEventListener
android.view.VelocityTracker
android.view.View.MeasureSpec
android.view.View.OnGenericMotionListener
android.view.View.OnHoverListener
android.view.ViewDebug
android.view.ViewGroup.MarginLayoutParams
android.view.ViewParent
android.view.Window
android.view.WindowManager
android.view.animation.AccelerateInterpolator
android.view.animation.AnimationSet
android.view.animation.LinearInterpolator
android.view.animation.ScaleAnimation
android.view.animation.Transformation
android.widget.Button
android.widget.CompoundButton.OnCheckedChangeListener

android.widget.EditText
android.widget.FrameLayout
android.widget.HorizontalScrollView
android.widget.ImageButton
android.widget.LinearLayout
android.widget.ListAdapter
android.widget.ScrollView
android.widget.Scroller
android.widget.SeekBar.OnSeekBarChangeListener
android.widget.SeekBar
android.widget.SimpleAdapter
androidx.annotation.NonNull
androidx.annotation.RequiresApi
androidx.appcompat.app.ActionBar.LayoutParams
androidx.core.app.ActivityCompat
androidx.core.content.ContextCompat
androidx.core.content.FileProvider
androidx.core.util.Consumer
androidx.window.java.layout.WindowInfoTrackerCallbackAdapter
androidx.window.layout.DisplayFeature
androidx.window.layout.FoldingFeature
androidx.window.layout.WindowInfoTracker
androidx.window.layout.WindowLayoutInfo
java.io.BufferedInputStream
java.io.BufferedWriter
java.io.ByteArrayInputStream
java.io.ByteArrayOutputStream
java.io.FileOutputStream
java.io.FileWriter
java.io.FilenameFilter
java.io.IOException
java.io.IOException
java.io.InputStream
java.io.InputStream
java.io.LineNumberReader
java.io.OutputStream
java.io.OutputStreamWriter
java.io.RandomAccessFile
java.io.StringReader
java.io.UnsupportedEncodingException
java.io.Writer
java.lang.AutoCloseable
java.lang.Double
java.lang.Float
java.lang.System
java.lang.System
java.lang.ref.SoftReference
java.lang.reflect.Field

java.lang.reflect.InvocationHandler
java.lang.reflect.InvocationTargetException
java.lang.reflect.Proxy
java.math.BigDecimal
java.net.HttpURLConnection
java.net.Proxy
java.net.ProxySelector
java.net.URL
java.net.URLEncoder
java.nio.ByteBuffer
java.nio.ByteOrder
java.nio.FloatBuffer
java.nio.charset.Charset
java.nio.charset.StandardCharsets
java.security.InvalidAlgorithmParameterException
java.security.KeyFactory
java.security.PrivateKey
java.security.PublicKey
java.security.SecureRandom
java.security.Signature
java.security.cert.CertPathValidator
java.security.cert.CertPathValidatorException
java.security.cert.Certificate
java.security.cert.CertificateFactory
java.security.cert.PKIXParameters
java.security.cert.PKIXRevocationChecker
java.security.cert.TrustAnchor
java.security.spec.RSAPublicKeySpec
java.text.NumberFormat
java.util.Base64
java.util.Collection
java.util.Comparator
java.util.EnumSet
java.util.Enumeration
java.util.HashMap
java.util.LinkedHashMap
java.util.Map
java.util.Properties
java.util.Scanner
java.util.TimeZone
java.util.Timer
java.util.TimerTask
java.util.Vector
java.util.concurrent.Callable
java.util.concurrent.ConcurrentLinkedQueue
java.util.concurrent.ExecutorService
java.util.concurrent.Executors
java.util.concurrent.FutureTask

java.util.concurrent.LinkedBlockingQueue
java.util.concurrent.Semaphore
java.util.concurrent.atomic.AtomicBoolean
java.util.logging.LogRecord
java.util.zip.ZipEntry
java.util.zip.ZipOutputStream
javax.crypto.spec.DESedeKeySpec
javax.crypto.spec.IvParameterSpec
javax.crypto.spec.SecretKeySpec
javax.microedition.khronos.egl.EGL10
javax.microedition.khronos.egl.EGLConfig
javax.microedition.khronos.egl.EGLContext
javax.microedition.khronos.egl.EGLDisplay
javax.microedition.khronos.egl.EGLSurface
javax.microedition.khronos.opengles.GL10
javax.security.auth.x500.X500Principal
junit.framework.TestSuite
org.json.JSONException
org.webrtc.AudioSource
org.webrtc.Camera2Enumerator
org.webrtc.CameraEnumerator
org.webrtc.CameraVideoCapturer
org.webrtc.DataChannel
org.webrtc.DtmfSender
org.webrtc.EglBase
org.webrtc.HardwareVideoDecoderFactory
org.webrtc.HardwareVideoEncoderFactory
org.webrtc.IceCandidate
org.webrtc.MediaConstraints.KeyValuePair
org.webrtc.PeerConnection.IceConnectionState
org.webrtc.PeerConnection.IceGatheringState
org.webrtc.PeerConnection.PeerConnectionState
org.webrtc.PeerConnection.SignalingState
org.webrtc.PeerConnectionFactory
org.webrtc.RtpReceiver
org.webrtc.RtpSender
org.webrtc.RtpSource
org.webrtc.SdpObserver
org.webrtc.SessionDescription
org.webrtc.StatsObserver
org.webrtc.StatsReport
org.webrtc.SurfaceTextureHelper
org.webrtc.VideoCodecInfo
org.webrtc.VideoDecoderFactory
org.webrtc.VideoEncoderFactory
org.webrtc.VideoFrame.Buffer
org.webrtc.VideoFrame.I420Buffer
org.webrtc.VideoFrame

org.webrtc.VideoSource
org.xmlpull.v1.XmlPullParser
org.xmlpull.v1.XmlPullParser
org.xmlpull.v1.XmlPullParserFactory

A.2 Third-Party Libraries

Listed below are the third-party libraries used by the Horizon Client product.

A.2.1 Windows Platform

Library	Version
Google.Protobuf	3.15.6
Microsoft.AspNetCore.WebUtilities	2.2.0-rtm-35687
Microsoft.Extensions.Localization	6.0.2
Microsoft.Extensions.Localization.Abstractions	6.0.2
Microsoft.Extensions.Logging	2.0.0-rtm-26452
Microsoft.Extensions.Logging.Abstractions	6.0.0
Microsoft.Extensions.Options	6.0.0
Microsoft.Extensions.Primitives	6.0.0
Microsoft.Net.Http.Headers	2.2.0-rtm-35687
Microsoft.Xaml.Behaviors.Wpf	1.1.31
ModernWpfUI	0.9.4
Prism.Core	8.1.97
Prism.Unity	8.1.97
Prism.Wpf	8.1.97
Serilog	2.10.0-master-24b67c6
Serilog.Enrichers.Context	4.2.0
Serilog.Enrichers.Thread	3.1.0
Serilog.Extensions.Logging	3.1.0
Serilog.Sinks.File	5.0.0-main-7eb21bd
System.CommandLine	2.0.0-beta1.20574.7
System.IO.Pipelines	6.0.2
Unity.Abstractions	5.11.7
Unity.Container	5.11.11
WiX	3.11.0.1701
blink	0.1.0
boost	1.67
cef	106.0.5249.61
curl	7.87.0
direct3d	10.0.19041.685
expat	2.5.0

ffmpeg	4.4
freeimage	3.18.0
gettext	0.20.1
glew	2.1.15711
glib	2.71.1
glibmm	2.70.0
gong-wpf-dragdrop	3.1.1
icu	69.1
jansson	2.14
libaom	3.3.0
libiconv	1.15
libidn	1.35
libjpeg	9d
libjpeg-turbo	2.1.0
libpng	1.6.37
libsigc++	3.0.6
libsrtp	2.1.0.0-pre
libtiff	4.1.0
libusb	1.0.24
libvpx	1.9.0-147-g61edec1ef
libxml2	2.10.2
openssl	1.0.2zg
opus	1.3.1
pcre	8.44
pdfium	2500
protobuf	3.18.3
skia	92
speex	1.2rc2
speexdsp	1.2.rc3
sqlite3	3.35.5
theora	1.1
x264	157
zlib	1.2.12

A.2.2 Android Platform

Library	**Version**
chromeos_client_lib	1.0.0
curl	7.87.0
eglib	none
expat	2.5.0

jbcrypt	0.4
libidn	1.35
libjpeg-turbo	2.1.0
libogg	1.3.2
libpng	1.6.37
libxml2	2.10.2
openssl	1.0.2zg
opus	1.3.1
pcsc-lite	1.8.11
rsa-api	2.3.2
samsung_hci_sdk	1.2.0
speex	1.2rc2
theora	1.1.1
libspeexdsp	1.2.rc3
icu4c	69.1
libyuv	r1788
snappy	1.1.7
nlohmann_json	3.10.5